



# Basi Dati

Concetti ed Architetture dei  
Database System

a.a 2020/2021  
Prof.ssa G. Tortora

# Modelli ed astrazione dati

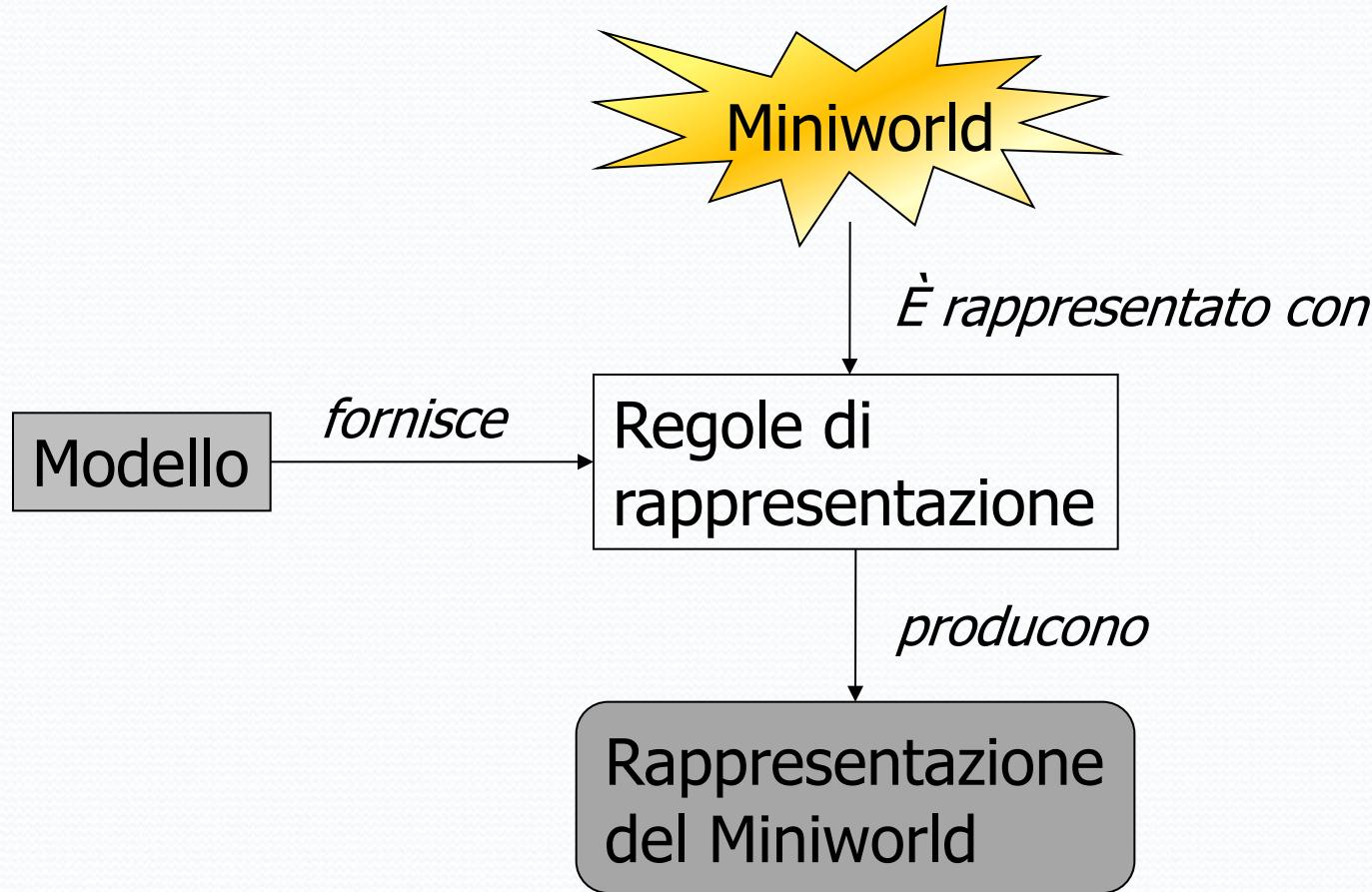
- L'approccio database ha la caratteristica fondamentale di fornire una forte astrazione dei dati, nascondendo tutti i dettagli di memorizzazione non necessari agli utenti.
- L'astrazione dei dati si ottiene per mezzo di un **data model**.
- Un insieme di concetti che possono essere usati per descrivere la struttura di una base di dati.

# Cos'è un modello

*Un modello deve:*

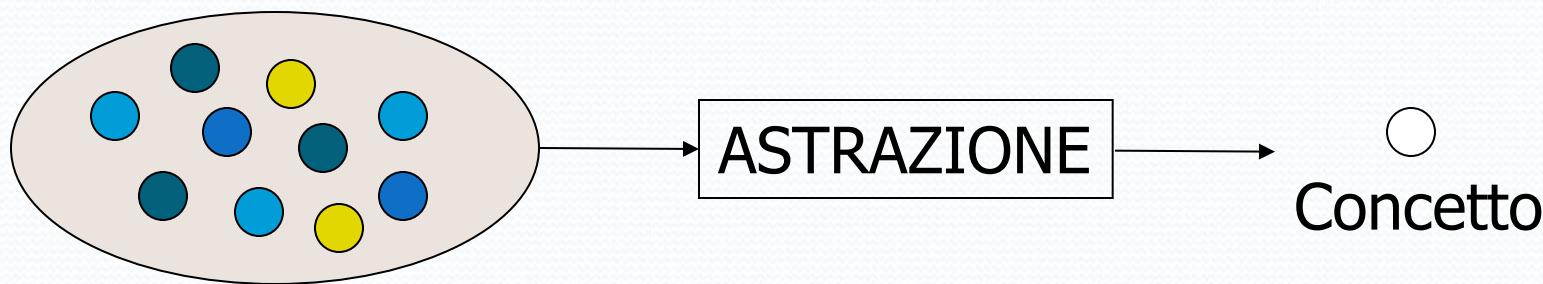
- Rappresentare una certa realtà:
  - *Es:* Una mappa rappresenta una porzione di territorio.  
È quindi un modello del territorio, che rappresenta alcune caratteristiche, nascondendone altre.
- Fornire un insieme di strutture simboliche per descrivere la rappresentazione della realtà:
  - *Es:* Una mappa ha una serie di simbolismi grafici per rappresentare delle entità reali.

# Cos'è un modello (Schema)



# Astrazione

- L'**astrazione** è un procedimento mentale che sostituisce con **un concetto** un insieme di oggetti in base ad alcune loro proprietà:



## *Esempio:*

l'astrazione consente di definire il concetto di “automobile”. Grazie ad esso è possibile descrivere e riconoscere tutte le automobili della realtà.

# Astrazione e modelli

- L'astrazione è importante perché permette di comunicare ed elaborare una rappresentazione del miniworld.
- Per comunicare ed elaborare tale rappresentazione si utilizzano dei modelli concettuali e logici.
- Un **modello concettuale** fornisce i simbolismi per rappresentare concetti astratti in modo indipendente da qualsiasi elaboratore.
- Un **modello logico** traduce le strutture concettuali in strutture logiche processabili da un DBMS.

# Data Model

- Un data model è un **insieme di concetti** usati per descrivere:
  - I tipi di dati
  - Le relazioni tra i dati
  - I vincoli semantici sui dati



# Data Model (2)

*Un data model include anche:*

- Un insieme di operazioni di base per specificare recuperi e aggiornamenti nel database.
  - *Es:* Inserzione, cancellazione, modifica, ritrovamento di un oggetto.
- Dei concetti per specificare aspetti dinamici dell'applicativo database.
  - Tali aspetti sono specificati da un insieme di operazioni valide definite dall'utente.
  - *Es:* Un'operazione COMPUTE\_GPA che calcola la media dei voti e che può essere applicata a un oggetto studente.

# Categorie di data model

*È possibile categorizzare i vari data model proposti secondo i concetti utilizzati per descrivere la struttura del database.*

- I data model di **alto livello** o **concettuali** forniscono concetti che sono vicini al modo di percepire i dati degli utenti.
- I data model **rappresentazionali** o di **implementazione** forniscono concetti comprensibili agli utenti finali, ma che non sono troppo lontani dal modo in cui i dati sono fisicamente organizzati.

Essi nascondono alcuni dettagli della memorizzazione dei dati ma possono essere implementati in maniera diretta.

- I data model di **basso livello** o **fisici** forniscono concetti che descrivono dettagli su come i dati sono memorizzati.

# Data model di alto livello

- I data model di alto livello usano concetti quali **entità**, **attributi** e **relazioni**:
  - Un'entità rappresenta un oggetto o concetto del mondo reale.
    - *Es:* Un impiegato o un progetto.
  - Un attributo rappresenta qualche proprietà importante che descrive ulteriormente un'entità.
    - *Es:* Il nome o lo stipendio di un impiegato.
  - Una relazione tra due o più entità rappresenta un'interazione tra le entità.
    - *Es:* Una relazione “lavora su” tra un impiegato e un progetto.
- Il più comune data model di alto livello è il modello **entità-relazione**.

# Data model rappresentazionali

- I data model rappresentazionali, per la facilità con cui possono essere implementati, comprendono i tre data model più usati dai DBMS commerciali:
  - Il modello relazionale
  - Il modello reticolare
  - Il modello gerarchico
- Tali modelli rappresentano i dati usando strutture di record. Per tale motivo sono chiamati anche modelli **record-based**.
- I modelli **object-oriented**, grazie alla forte astrazione dei dati, costituiscono una nuova famiglia di data model, posti a metà strada tra il rappresentazionale e il concettuale.

# Data model fisici

- I data model fisici descrivono come sono memorizzati i dati nel calcolatore rappresentando informazioni quali formati di record, ordinamenti di record, percorsi di accesso.
  - Un percorso di accesso è una struttura che rende efficiente la ricerca di particolari record di database.

# Schemi e istanze di database

- In qualsiasi data model è necessario distinguere tra la descrizione del database ed il database stesso.
- La differenza è simile a quella tra tipi e variabili nei linguaggi di programmazione.
- Lo **schema** è la struttura logica del database.
- Un'**istanza** è il contenuto del database in un particolare istante di tempo.

# Schemi di database

- La descrizione del database è detta **schema** (o **metadati**). Lo schema del database è specificato in fase di progetto e non ci si aspetta che cambi frequentemente.
  - Per rappresentare uno schema si crea un **diagramma di schema**, secondo opportune convenzioni definite dal data model. Un diagramma di schema visualizza la struttura dei record ma non le reali istanze dei record.
- Il DBMS memorizza lo schema nel catalogo per poterne fare riferimento ogni qualvolta gli occorre.

# SCHEMI DI DATABASE:

## *Esempio DB Università*

STUDENTE		
NOME	MATRICOLA	ANNO
CORSO		
DENOMIN.	SEMESTRE	TITOLARE
PREREQUISITI		
DENOMIN.	PROPEDEUTICITA'	
VOTAZIONE		
NOME	DENOMIN.	VOTO

- Ogni oggetto nello schema è detto **costrutto di schema**.
  - Es: STUDENTE o CORSO sono ciascuno un costrutto di schema.

# Istanze di database

- I dati reali in un db possono cambiare frequentemente.
  - *Es:* Il db UNIVERSITA' cambia ogni volta che si inserisce un nuovo studente o si registra un nuovo esame per uno studente.
- Uno **stato del database** (detto anche istanza o insieme di occorrenze del db) è l'insieme dei dati presenti nel database in un particolare istante di tempo.

# Istanze di database (2)

- Dato un stato del database, ogni costrutto di schema ha un proprio insieme corrente di istanze
  - *Es:* Il costrutto STUDENTE conterrà l'insieme di entità (o record) studenti individuali come sue istanze.
- Per un dato schema di database possono essere costruiti diversi stati di database.  
Ogni volta che si inserisce o cancella un record o si modifica il valore di un data item, si cambia lo stato del database.

# L' importanza dello schema di database

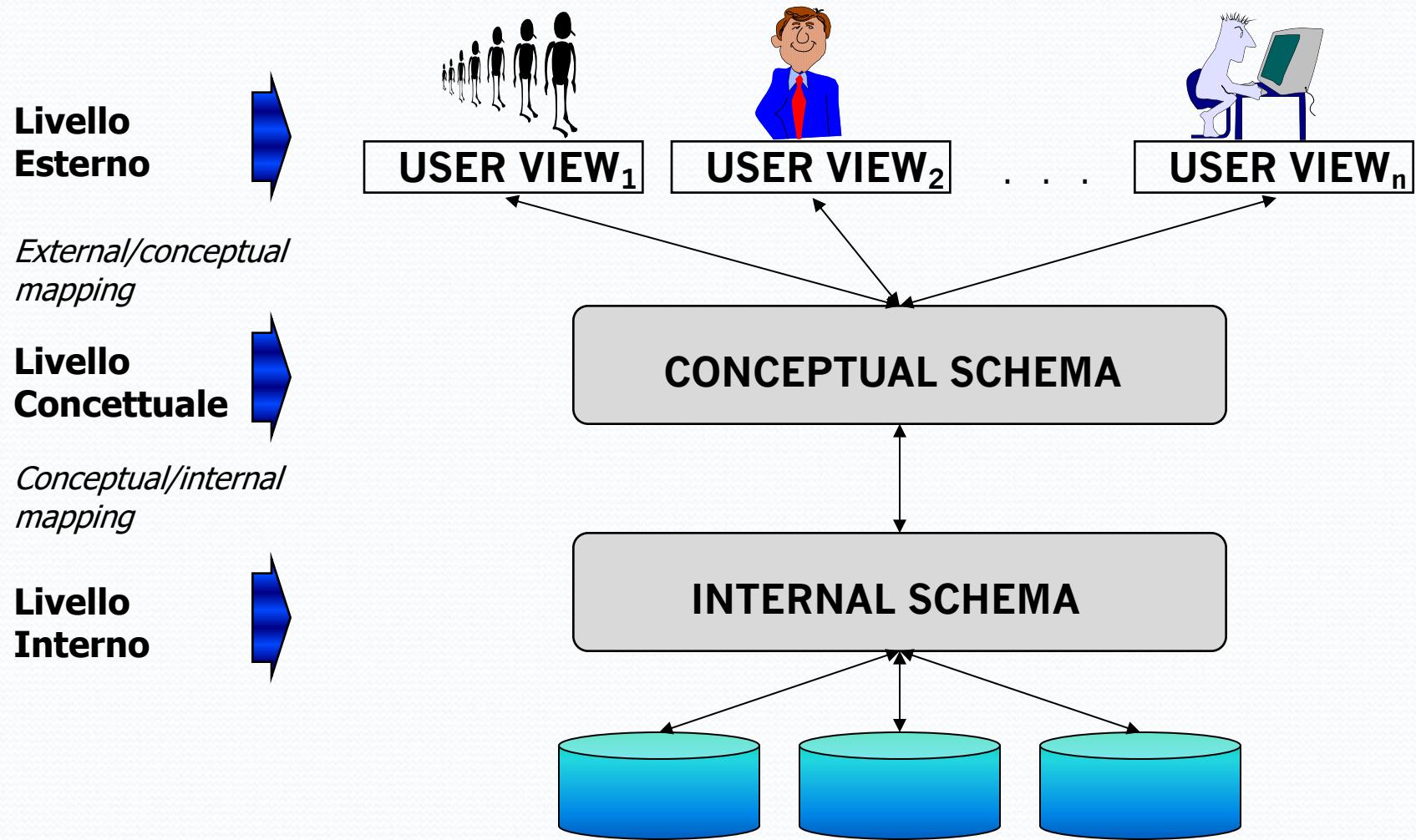
- Quando si crea un nuovo database, si specifica al DBMS lo schema del database:
  - in tale fase il db è nello “**stato vuoto**”, senza dati. Si passa nello “**stato iniziale**”, quando si inseriscono i dati per la prima volta.
- Il DBMS è parzialmente responsabile nel garantire che ogni stato del DB sia valido, cioè che soddisfi la struttura ed i vincoli specificati nello schema.
- Quindi:
  - specificare uno schema corretto è estremamente importante; lo schema deve essere progettato con la massima cura.

# Architettura di un database system

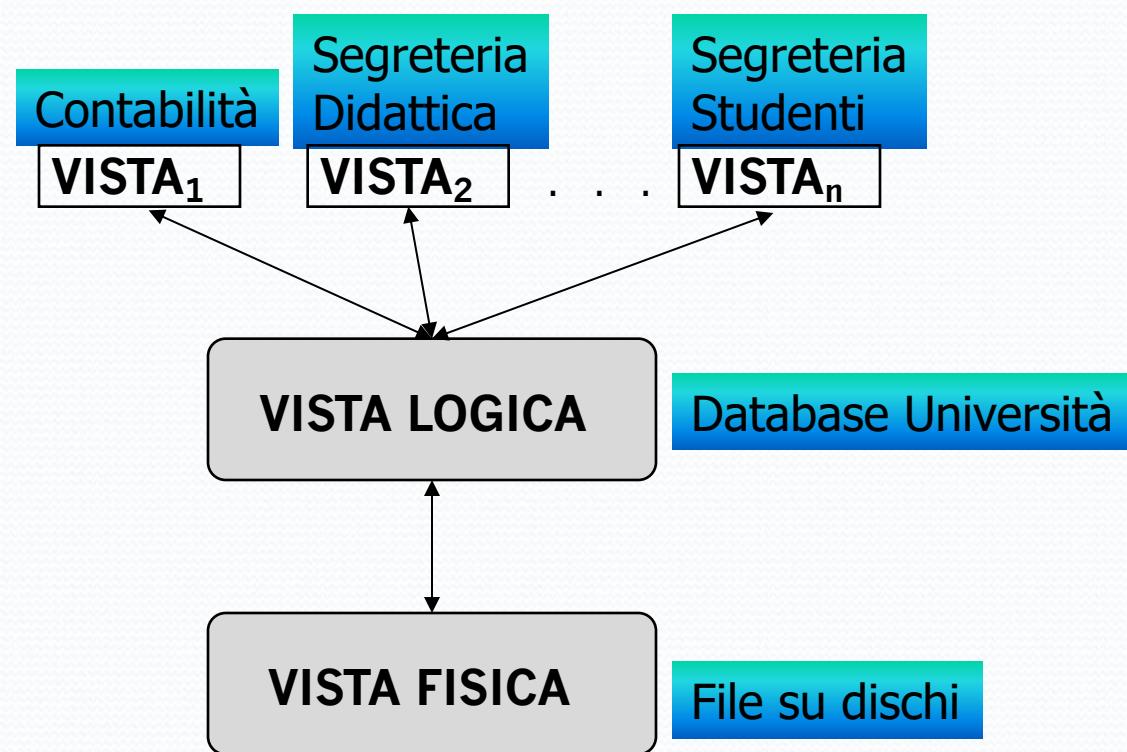
# L'architettura three-schema

- Ricordiamo le caratteristiche principali dell'approccio di database:
  - supporto di viste multiple per gli utenti;
  - uso di un catalogo per memorizzare la descrizione del DB (lo schema);
  - condivisione dei dati ed elaborazione delle transazioni in modalità multiutente.
- L'architettura per database system chiamata **three-schema** aiuta ad ottenere e visualizzare tali caratteristiche, **definendo il database in tre livelli** e separando quindi le applicazioni utente dal database fisico.

# L'architettura three-schema (2)



# L'architettura three-schema: *Esempio*



Gli schemi possono essere specificati a tre livelli:

- a livello esterno (o view level) che include schemi esterni o viste utente;
- a livello concettuale come schema concettuale;
- a livello interno come schema interno.

# LIVELLO ESTERNO

*Livello esterno come schema esterno:*

- Definisce un sottoinsieme del DB per una particolare applicazione.
- Include più schemi esterni o viste utente.
- Usa un data model di alto livello o di implementazione.
- Ogni schema esterno descrive la parte del DB cui è interessato un particolare utente e nasconde il resto del DB a quel gruppo.
- **Esempio:**  
La segreteria didattica ha necessità di vedere tutte le informazioni sugli esami, ma non deve vedere informazioni sugli stipendi.

# LIVELLO CONCETTUALE

*Livello concettuale come schema concettuale:*

- Descrive la struttura del DB per una comunità di utenti.
- Lo schema concettuale nasconde i dettagli delle strutture di storage fisico e si concentra sulla descrizione delle entità, tipi di dati, relazioni, operazioni utente e vincoli.
- può usare un data model ad alto livello o uno di implementazione.
- **Esempio:**  
**type Studente= record**  
    **nome : string;**  
    **matricola : string;**  
    **anno : string;**  
    **end;**

# LIVELLO INTERNO

*Livello interno come schema interno:*

- Descrive la struttura di memorizzazione fisica del DB.
- Lo schema interno usa un data model fisico e descrive i dettagli completi del data storage e gli access paths del DB.
- **Esempio:**  
Dividi i record del file studenti in tre partizioni sui dischi 5, 6 e 7.

# I mapping tra i livelli dell'architettura

- I tre schemi sono solo delle **descrizioni** di dati: gli unici dati che realmente esistono sono a livello fisico.
- In un DBMS basato sull'architettura three-schema, ogni gruppo di utenti utilizza una propria vista esterna.
- Un mapping è un processo di trasformazione delle richieste e dei risultati. Il DBMS trasforma:
  - una richiesta specificata su uno schema esterno...
  - ...in una richiesta secondo schema concettuale e poi...
  - ...in una richiesta sullo schema interno per procedere sul database memorizzato.

# Indipendenza dei dati

L'architettura three-schema è utile per evidenziare il concetto di indipendenza dei dati, ovvero la capacità di cambiare lo schema a un livello del database senza dover cambiare lo schema al livello superiore.

Si definiscono due tipi di indipendenza dei dati:

- Indipendenza logica dei dati:
  - lo schema concettuale può essere cambiato senza dover cambiare gli schemi esterni o i programmi applicativi.
- Indipendenza fisica dei dati:
  - lo schema interno può essere cambiato senza dover cambiare gli schemi concettuali (o esterni).

# Indipendenza logica

- Indipendenza logica dei dati:
  - lo schema concettuale può essere cambiato per espandere oppure per ridurre il database (aggiungendo o rimuovendo, rispettivamente, un tipo di record o data item).
  - Se si elimina un tipo di record, gli schemi esterni che si riferiscono solo ai dati restanti non devono essere alterati.
    - In caso di modifiche, solo le definizioni delle viste ed i mapping devono essere cambiati;
    - i programmi applicativi che fanno riferimento agli schemi esterni sono indifferenti alle modifiche.

# Indipendenza logica: *Esempio*

Lo schema esterno

ESAMI SUPERATI	Nome	Matricola	ESAMI	
			DENOMIN.	VOTO
	Rossi	056/000484	Diritto Privato	24
			Procedura Civile	28
	Verdi	011/120579	Procedura Penale	30
			Diritto Civile	27

non dovrebbe essere alterato cambiando il file votazione da

VOTAZIONE			
NOME	DENOMIN.	VOTO	
Rossi	Diritto Privato	24	
Rossi	Procedura Civile	28	
Verdi	Procedura Penale	30	
...	...	...	

a

VOTAZIONE			
NOME	MATRICOLA	DENOMIN.	VOTO
Rossi	056/000484	Diritto Privato	24
Rossi	056/000484	Procedura Civile	28
Verdi	056/100084	Procedura Penale	30
...	...	...	...

# Indipendenza fisica

- Indipendenza fisica dei dati:
  - lo schema interno può essere cambiato senza dover cambiare gli schemi concettuali (o esterni).
  - Un cambiamento dello schema interno può essere dovuto alla riorganizzazione di qualche file fisico (*es:* creando ulteriori strutture di accesso), per migliorare l'esecuzione del ritrovamento o dell'aggiornamento.
    - Se i dati non subiscono alterazioni, non è necessario cambiare lo schema concettuale.
  - L'indipendenza fisica si ottiene più facilmente di quella logica.

# Indipendenza fisica: *Esempio*

- Fornire un percorso di accesso per migliorare il ritrovamento dei record del file **CORSO** con **Denominaz.** e **Semestre** non dovrebbe richiedere variazioni a una query come  
*“ritrova tutti i corsi tenuti nel secondo semestre”*  
sebbene la query possa essere eseguita in maniera più efficiente.

# L'architettura three-schema: vantaggi e svantaggi

## VANTAGGI

- L'architettura three-schema consente facilmente di ottenere una reale indipendenza dei dati.
- Il database risulta più flessibile e scalabile.

## SVANTAGGI

- Il catalogo del DBMS multilivello deve avere dimensioni maggiori, per includere informazioni su come trasformare le richieste e di dati tra i vari livelli.
- I due livelli di mapping creano un overhead durante la compilazione o esecuzione di una query di un programma, causando inefficienze nel DBMS.

# Linguaggi per i database

# Linguaggi DBMS

- *Un DBMS deve fornire ad ogni categoria di utenti delle interfacce e dei linguaggi adeguati.*
- Alla fase di progetto del DB segue quella di specifica degli schemi concettuale ed interno e di qualsiasi mapping tra i due.
- Nei DBMS dove non c'è una netta separazione tra livelli, progettisti e DBA usano un **data definition language (DDL)** per definire entrambi gli schemi. Il compilatore del DDL (contenuto nel DBMS) ha la funzione di:
  - Trattare gli statement DDL per identificare le descrizioni dei costrutti dello schema.
  - Memorizzare la descrizione dello schema nel catalogo del DBMS.

# Linguaggi DBMS (2)

- Dove c'è una chiara distinzione tra livello concettuale ed interno si usa:
  - uno **storage definition language (SDL)** per specificare lo schema interno,
  - il DDL per specificare soltanto lo schema concettuale.
- I mapping tra i due schemi possono essere specificati in uno qualsiasi dei due linguaggi.
- Nelle architetture three-schema viene usato un **view definition language (VDL)** per specificare le viste utente e i loro mapping sullo schema concettuale.

# Linguaggi DBMS (3)

- Una volta che gli schemi sono compilati ed il DB è riempito di dati:
  - il DBMS fornisce un **data manipulation language (DML)** per consentire agli utenti di manipolare (cioè recuperare, inserire, cancellare e aggiornare) dati.
- Nei DBMS moderni si tende ad utilizzare un unico linguaggio integrato che comprende costrutti:
  - per la definizione di schemi concettuali;
  - per la definizione di viste;
  - per la manipolazione dei DB e
  - per la definizione della memorizzazione.
- **Esempio:**
  - Il linguaggio di database relazionali **structured query language (SQL)** rappresenta una combinazione di DDL, VDL, DML e SDL.

# I DML

*Esistono due tipi di DML:*

- DML di **alto livello** o non procedurali.
  - Consentono da soli di specificare operazioni di database complesse in maniera concisa.
- In molti DBMS gli statement di DML di alto livello
  - Possono essere specificati interattivamente da terminale.
  - Possono essere inglobati (*embedded*) in un linguaggio di programmazione general-purpose (in questo caso gli statement DML sono identificati all'interno del programma in modo che il DBMS possa trattarli (*precompilazione*)).
  - Sono detti **set-at-a-time** o **set-oriented** perché possono specificare e ritrovare molti record in singolo statement DML (**es:** l'SQL).
  - Sono detti dichiarativi perché una query di un DML high-level spesso specifica **quale** dato deve essere ritrovato piuttosto che **come** ritrovarlo.

# I DML (2)

- DML di **basso livello** o procedurale.
  - Deve essere inglobato in un linguaggio general-purpose;
  - È detto **record-at-a-time** perché tipicamente ritrova record individuali nei DB e tratta ciascun record separatamente;  
ha quindi bisogno di usare costrutti di programmazione come l'iterazione per ritrovare e trattare ogni record da un insieme di record.

# SQL, un linguaggio interattivo

- “*Trovare i corsi tenuti in aule a piano terra*”

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

# SQL, un linguaggio interattivo (2)

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

Corso	Aula	Piano
Sistemi	N3	Terra
Reti	N3	Terra

# SQL immerso (embedded) in linguaggio ospite

```
write('nome della citta'?); readln(citta);
EXEC SQL DECLARE P CURSOR FOR
    SELECT NOME, REDDITO
    FROM PERSONE
    WHERE CITTA = :citta ;
EXEC SQL OPEN P ;
EXEC SQL FETCH P INTO :nome, :reddito ;
while SQLCODE = Ø 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 ;
```

# Interfacce per i database

# Interfacce di DBMS

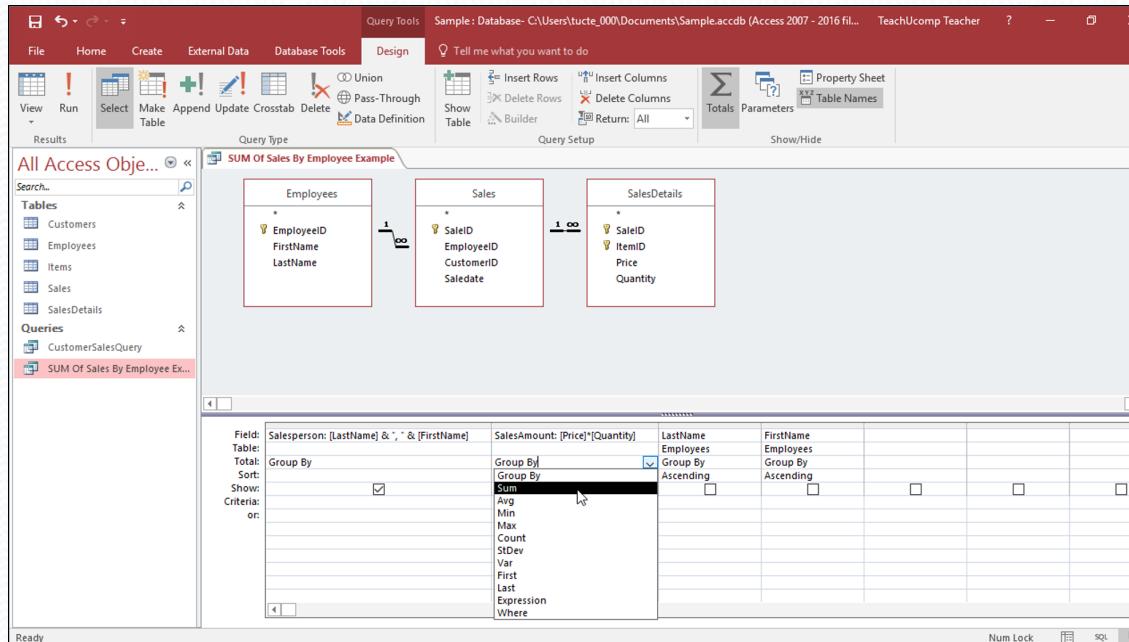
- Interfacce Menu-based:
  - Presentano all'utente liste di opzioni, dette menù, che guidano l'utente nella formulazione di una richiesta.  
La query è composta un passo alla volta scegliendo opzioni da una lista di menu che viene visualizzata dal sistema.
  - Non c'è necessità di memorizzare i comandi specifici e la sintassi di un query language.
  - I *menu pull-down* sono spesso usati nelle interfacce browsing, che consentono all'utente di guardare i contenuti del DB in maniera non strutturata.

# Interfacce di DBMS (2)

- Interfacce Grafiche:

- Tipicamente un'interfaccia grafica dispone su video uno schema in forma diagrammatica.

L'utente specifica una query manipolando il diagramma, tramite dispositivi di puntamento, come il mouse.



# Interfacce di DBMS (3)

- Interfacce Form-based:
  - Un'interfaccia form-based visualizza un form per ciascun utente. Gli utenti possono:
    - Riempire tutte le entrate del form completamente per inserire nuovi dati.
    - Riempire solo alcune entrate, nel qual caso il DBMS ritroverà dati che fanno matching per il resto delle entrate.
  - I form sono in genere progettati e programmati per utenti naive.
  - Il form specification language, che molti DBMS hanno, aiuta i programmatori a specificare i form.
    - *Es:* SQL\*Forms

## Ricerca avanzata

Trova pagine web che contengono...

tutte queste parole:

Per fare questo nella casella di ricerca.

Digita le parole importanti: labrador retriever nero

questa esatta parola o frase:

Racchiudi le parole esatte tra virgolette: "labrador retriever"

una qualunque di queste parole:

Digita OR tra tutte le parole che vuoi: miniatura OR standard

nessuna di queste parole:

Anteponi il segno - (meno) alle parole da escludere:  
-roditore, - "Jack Russell"

numeri da:

a

Inserisci due punti (..) tra i numeri e aggiungi un'unità di misura:  
10..35 kg, € 300..€ 500, 2010..2011

Poi limita i risultati per...

lingua:

Trova le pagine nella lingua selezionata.

area geografica:

Trova le pagine pubblicate in un'area geografica specifica.

ultimo aggiornamento:

Trova le pagine aggiornate nel periodo di tempo specificato.

sito o dominio:

Cerca in un sito (come wikipedia.org ) o visualizza soltanto i risultati relativi a un dominio, come .edu, .org o .gov

termini che compaiono:

Cerca i termini nell'intera pagina, nel titolo della pagina, nell'indirizzo web o nei link che rimandano alla pagina desiderata.

SafeSearch:

Indica a SafeSearch se filtrare i contenuti sessualmente esplicativi.

tipo di file:

Trova le pagine nel formato che preferisci.

diritti di utilizzo:

Trova le pagine che puoi utilizzare liberamente.

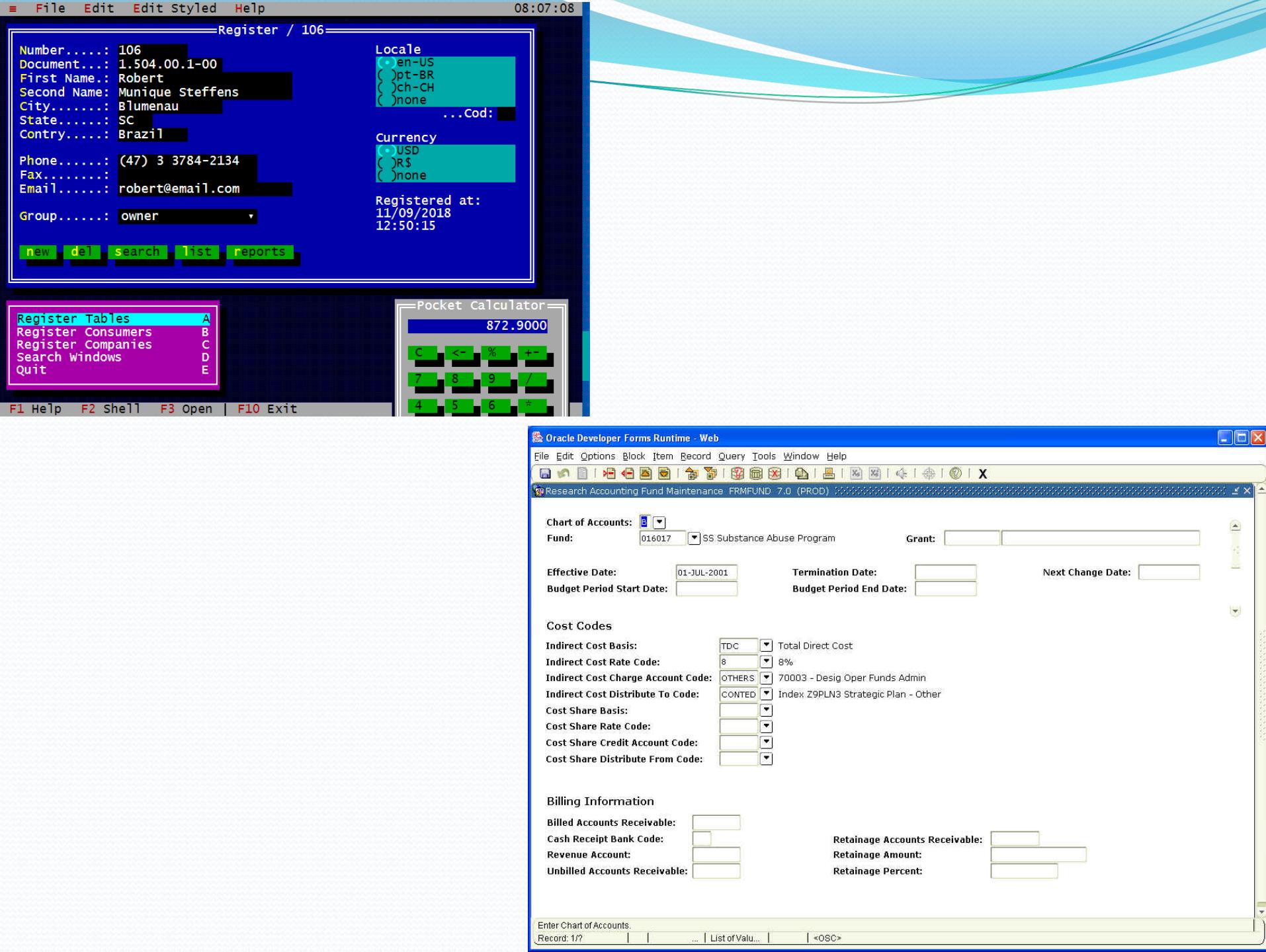
**Ricerca avanzata**

# Interfacce di DBMS (4)

- Interfacce in Linguaggi Naturali:
  - Accettano richieste scritte in linguaggio naturale e cercano di “*capire*”;
  - Hanno un proprio schema specifico, che è simile allo schema concettuale del DB.
  - Nell’interpretare le richieste l’interfaccia fa riferimento alle parole nello schema e a un insieme di parole standard (*dizionario*);
  - se l’interpretazione ha successo, l’interfaccia genera una query high-level che corrisponde alla richiesta in linguaggio naturale e la sottomette al DBMS altrimenti inizia un dialogo con l’utente per chiarire la richiesta.
    - *Es:* Motori di ricerca come Google o AskJeeves.

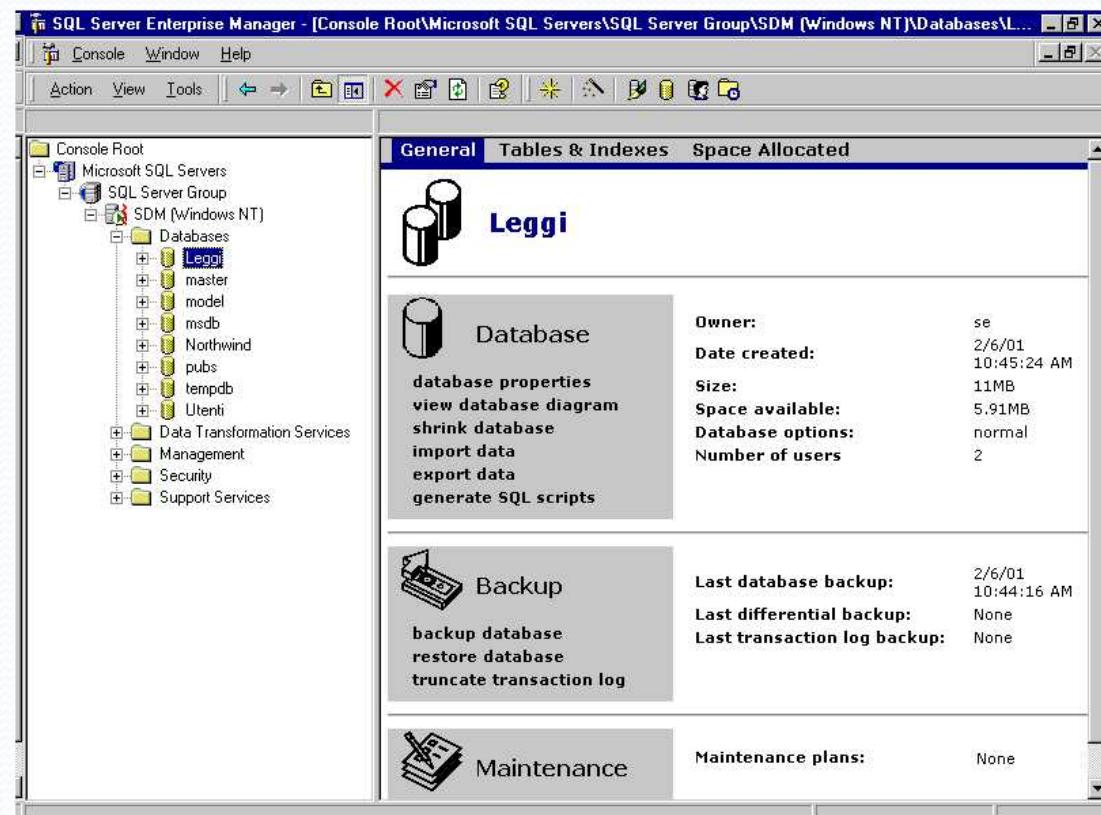
# Interfacce di DBMS (5)

- Interfacce per utenti parametrici:
  - Sono progettate per utenti naive (*es*: impiegati di banca), così da comprendere un piccolo insieme di comandi abbreviati.
  - Riguardano l'esecuzione ripetuta di un piccolo insieme di operazioni.
  - Viene implementata una interfaccia speciale per ciascuna classe nota di utenti non esperti.



# Interfacce di DBMS (6)

- Interfacce per DBA:
  - La maggior parte dei DBMS contengono comandi privilegiati che possono essere usati solo dallo staff del DBA (**es:** creare account, settare dei parametri, cambiare uno schema, riorganizzare la struttura di memorizzazione del DB).



Microsoft SQL Server Manager

MySQL Workbench

remote/server

Management

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

MySQL ENTERPRISE

- Audit Inspector
- Online Backup
- Backup Recovery

SCHEMAS

Filter objects

- chess\_db
- db223859
- mydb\_merge\_table
- sakila
- test
- test\_underscore
- world

Object Info Session

No object selected

filehorse.com

Could not Open WB Admin

Connection Name  
remote/server

Host: printberry  
Socket: /var/run/mysqld/mysqld.sock  
Port: 3306  
Version: 5.5.21-0+wheezy1 (Debian)  
Compiled For: debian-linux-gnu (armv7l)

Available Server Features

Performance Schema: Off SSL Availability: Off  
Thread Pool: n/a PAM Authentication: Off  
Memcached Plugin: n/a Password Validation: n/a  
Semisync Replication Plugin: n/a Audit Log: n/a

Server Directories

Base Directory: /usr  
Data Directory: /var/lib/mysql/  
Disk Space in Data Dir: unable to retrieve  
Plugins Directory: /usr/lib/mysql/plugin/  
Tmp Directory: /tmp  
Error Log: Off  
General Log: Off  
Slow Query Log: Off

Replication Slave

this server is not a slave in a replication setup

Authentication

SHA256 password private key: n/a  
SHA256 password public key: n/a

MySQL Workbench

Query 1 Administration - Server Status

Server Status: Running Load: 0.0 Connections: 9

Traffic: 8.10 KB/s Key Efficiency: 95.1%

Queries per Second: 0 InnoDB Buffer Usage: 8.0%

InnoDB Reads per Second: 0 InnoDB Writes per Second: 0

Administration - Users and Privileges

Local instance 3306

User Accounts

User	From Host
(!) <anonymous>	%
<anonymous>	localhost
<anonymous>	i.telecom
root	localhost
root	i.telecom
root	127.0.0.1
root	::1

Details for account root@127.0.0.1

Login Account Limits Administrative Roles Schema Privileges

Login Name: root You may create multiple accounts with % to connect from different hosts.

Authentication Type: Standard For the standard password and/or host select 'Standard'.

Limit to Hosts Matching: 127.0.0.1 % and \_ wildcards may be used

Password: \*\*\*\*\* Type a password to reset it. Consider using a password with 8 or more characters with mixed case letters, numbers and punctuation marks.

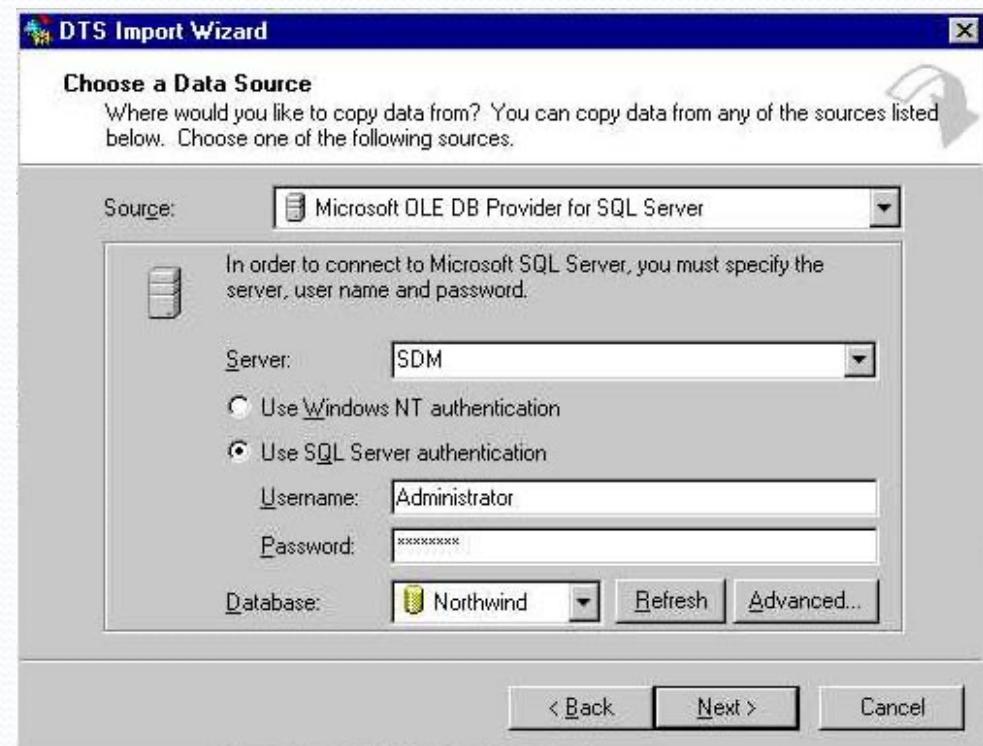
Confirm Password: \*\*\*\*\* Enter password again to confirm.

No password is set for this account.

Add Account Delete Refresh Revoke All Privileges Expire Password Revert Apply

# Utility di un DB System

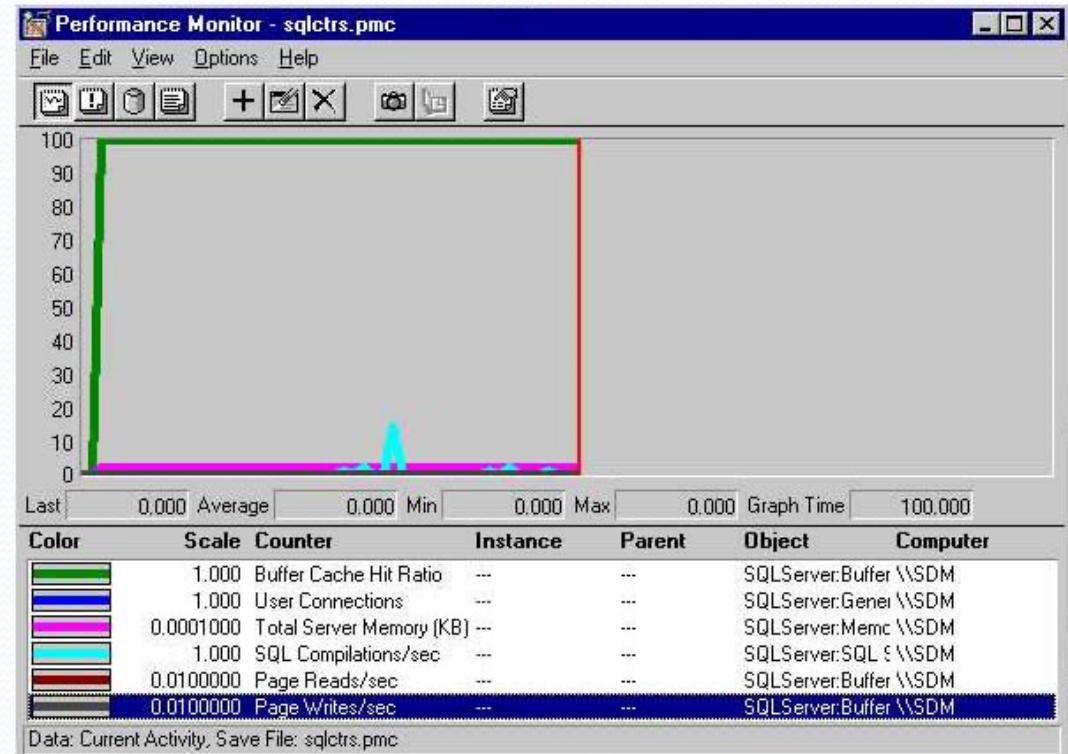
- Loading utility:
  - Per la conversione di dati da formati esterni al formato di specifico del DBMS.



Microsoft SQL Server Import Wizard

# Utility di un DB System (2)

- Performance Monitor utility:
  - Per monitorare il DB e costruire statistiche per il DBA  
*(riorganizzare o meno i file per migliorare le prestazioni).*



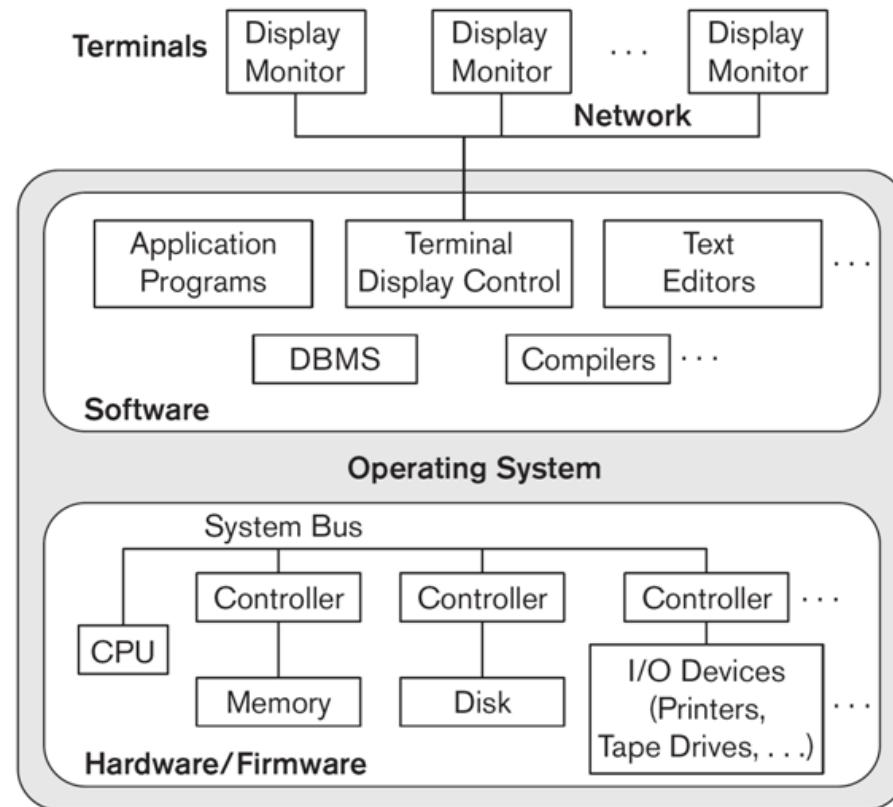
Microsoft SQL Server Performance Monitor

# Utility di un DB System (3)

- Backup utility:
  - Per creare copie di backup del DB su dispositivi di streaming.
  - La copia di backup può essere usata per ripristinare il DB in caso di guasto catastrofico.
  - Spesso si usano *backup incrementali* dove si memorizzano i cambiamenti rispetto al backup precedente (consente di risparmiare spazio).
- File Organization utility:
  - Per riorganizzare i file del database, allo scopo di migliorare le prestazioni.

# Architettura del DBMS centralizzato

- Tutte le funzionalità del DBMS, l'esecuzione del programma applicativo, e l'elaborazione dell'interfaccia utente sono eseguite su una sola macchina.

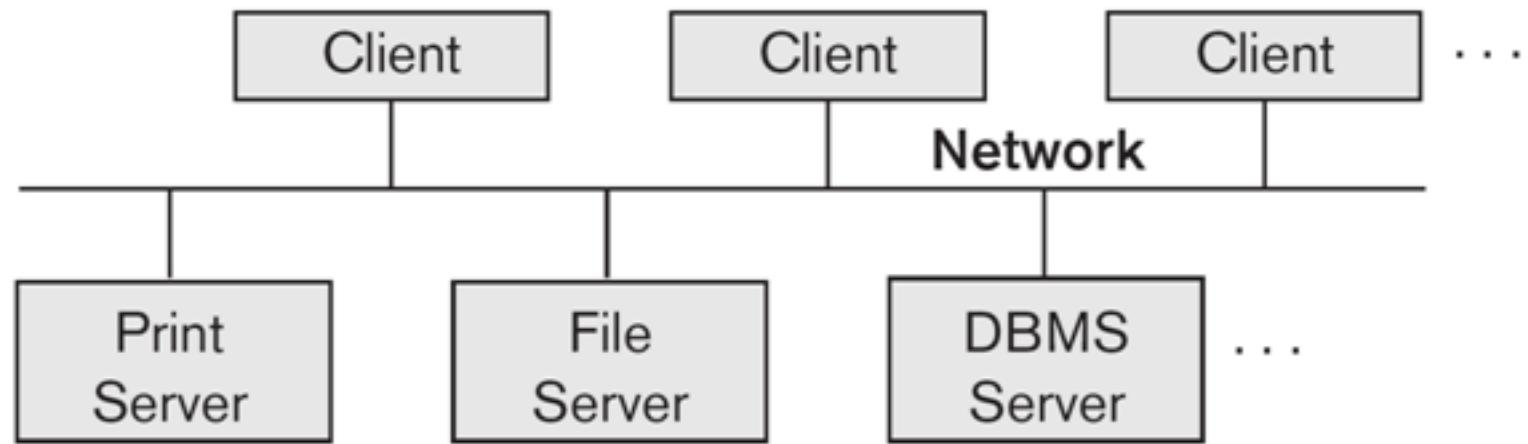


# Architettura Client/Server base

- Sono **Server** con specifiche funzionalità:
  - **File server**
    - Mantiene e gestisce i file delle macchine client.
  - **Printer server**
    - È connesso a diverse stampanti,
    - Tutte le richieste di stampa dalle macchine client sono trasmesse a questa macchina.
  - **Web server o e-mail server**
- Le **macchine Client**:
  - Forniscono l'utente di un interfaccia appropriata per utilizzare i server.
  - Hanno capacità operazionali locali per eseguire applicazioni in locale.

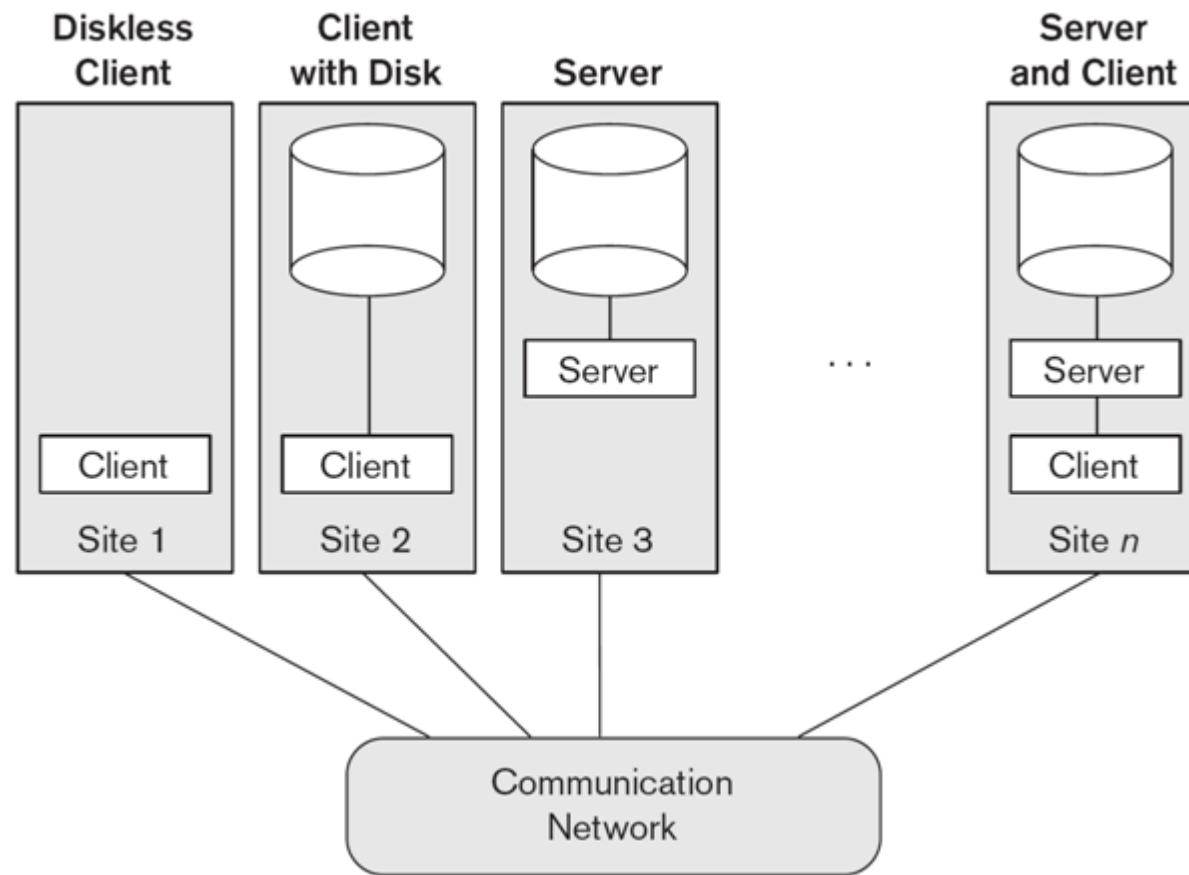
# Architettura client/server Two-Tier

- Architettura logica:



# Architettura client/server Two-Tier (2)

- Architettura fisica:



# Architettura client/server Two-Tier (3)

- Open Database Connectivity (ODBC)
  - Fornisce le Application Programming Interface (API).
  - Permette ad un programma in esecuzione sulla macchina client di comunicare con il DBMS
    - Entrambi le macchine client e server devono avere installato il software necessario.
- Java DataBase Connectivity (JDBC)
  - Permette ai programmi client scritti in **Java** di accedere ad uno o più DBMS attraverso un interfaccia standard.

# Architettura client/server Three-Tier

- **Application server o Web server**

- Aggiunge uno strato intermedio tra il client ed il database server.
- Esegue i programmi applicativi e memorizza le regole di business.

