Contabilità Direzionale
e
Digitalizzazione

Dott. Ing. Alfonso Stuardi

UNIVERSITÀ DEGLI STUDI DI TERAMO

Dati, DMBS
e
Informazioni

Dati e Informazioni

Informazione (def): notizia, o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni e modi d'essere.

Dato (def): elementi di informazione constituiti da simboli che devono essere elaborati.

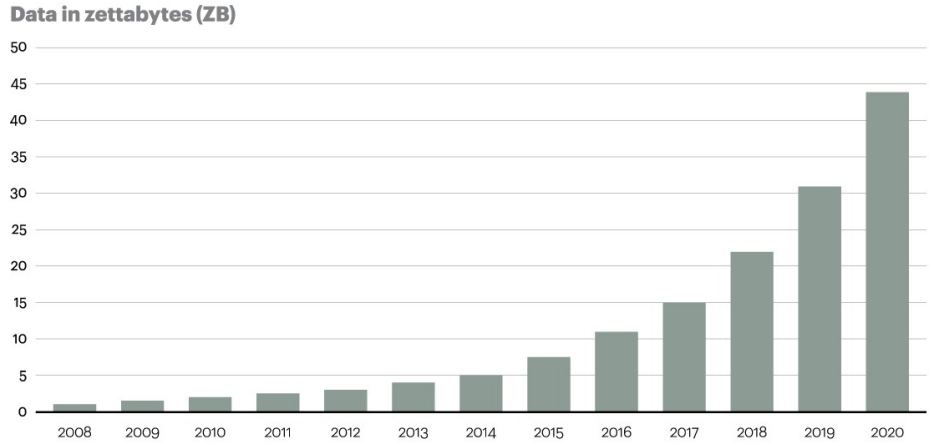
NOTA: Senza interpretazione, il dato non è molto utile!

FERRARI, 4

UNIVERSITÀ DEGLI STUDI DI TERAMO

Dati e Informazioni

Figure 1
Data is growing at a 40 percent compound annual rate, reaching nearly 45 ZB by 2020



Source: Oracle, 2012

Source: http://www.atkearney.it/

UNIVERSITÀ DEGLI STUDI DI TERAMO

Dati e Informazioni

- Worldwide data is expected to hit **175 zettabytes** by 2025, representing a 61% CAGR[1]
    - 51% of the data will be in data centers and 49% will be in the public cloud
    - 90 ZB of this data will be from IoT devices in 2025

- In 2018, **71% of enterprises** reported that unstructured data was growing “somewhat faster” or “much faster” than other business data

- **80% of data** will be unstructured by 2025
    - On top of business documents, video and audio are added new content such as social media, IoT, streaming and geo data

Dati e Informazioni

Chi produce questi dati?

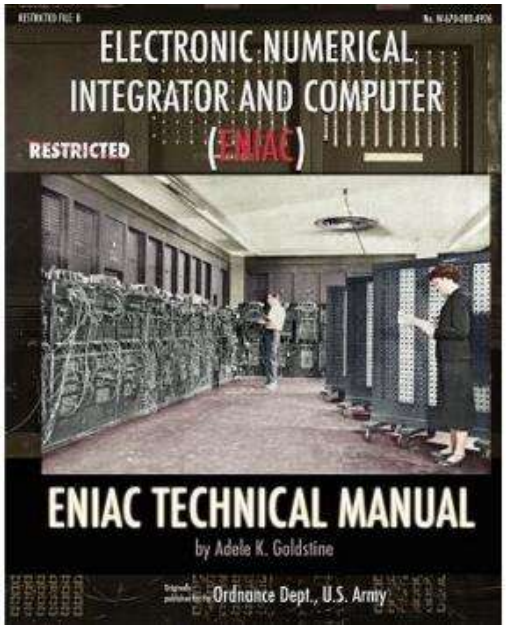
1. Sistemi Informativi aziendali

2. Social media

3. Sorgenti di big-data (es. IoT)

UNIVERSITÀ DEGLI STUDI DI TERAMO

Dati e Informazioni



ENIAC (1946)
L'era dei supercalcolatori

Commodore PET (1977)
L'era dei PC

iPhone (2007)
L'era degli smartphone

UNIVERSITÀ DEGLI STUDI DI TERAMO

Dati e Informazioni

Oggi: Informatica Pervasiva

FITBAND 

 AMAZON Alexa

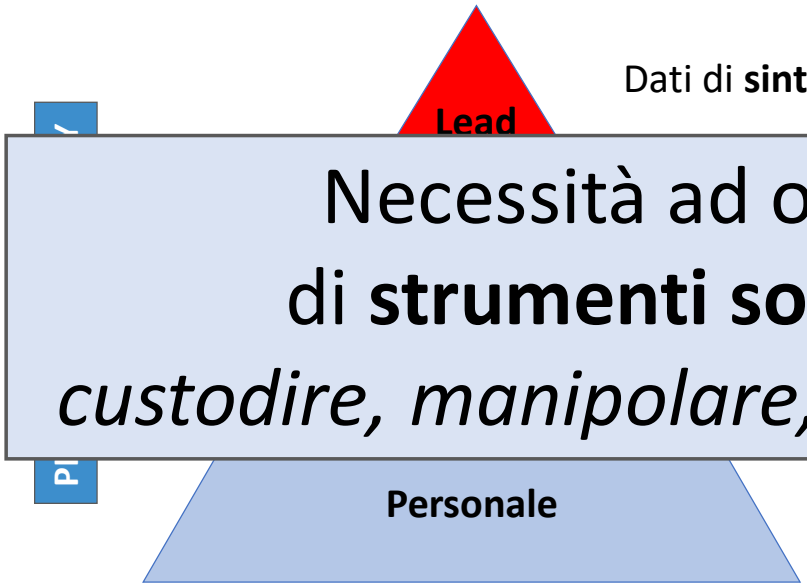
SMARTWATCH 

 TESLA AutoPilot

UNIVERSITÀ DEGLI STUDI DI TERAMO

Importanza dei Dati

R1. DATI come risorsa aziendale, alla stessa maniera del capitale, degli impanti di produzione, delle persone, e dei beni prodotti dall'azienda



Dati di **sintesi** sull'andamento aziendale

Necessità ad ogni livello
di **strumenti software** per
custodire, manipolare, analizzare i dati!

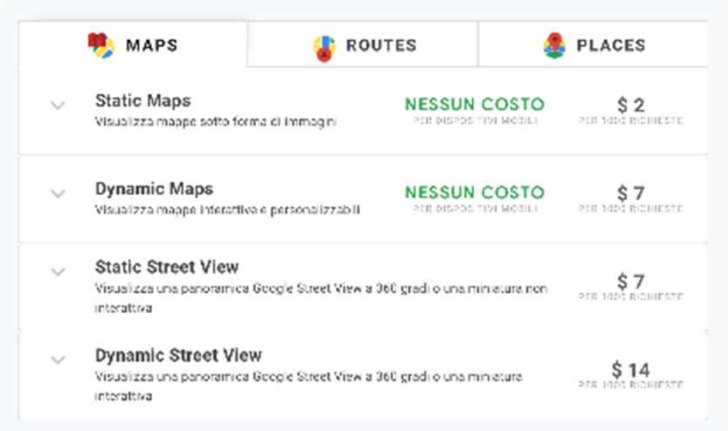
onale

Dati di **dettaglio**: ordini, dati fatturazioni, bilancio, contabilità, etc

UNIVERSITÀ DEGLI STUDI DI TERAMO

Importanza dei Dati

R2. DATI come <u>bene prodotto dall'azienda e fonte di profitto!</u>



https://cloud.google.com/maps-platform/pricing/

http://www.bigdataexchange.com

UNIVERSITÀ DEGLI STUDI DI TERAMO

Importanza dei Dati

R3. DATI = informazione = conoscenza = supporto alle decisioni!

Source: http://www.conbusinessintelligence.com/

BUSINESS INTELLIGENCE (BI)

Insieme di **processi aziendali, metodologie e tool** per raccogliere i dati di un'azienda, ed estrarre **informazioni di supporto alla decisioni strategiche**

UNIVERSITÀ DEGLI STUDI DI TERAMO

Importanza dei Dati

R3. DATI = informazione = conoscenza = supporto alle decisioni!

RECOMMENDATION SYSTEMS

https://www.researchgate.net/figure/Amazoncom-collaborative-based-recommender-system_fig6_221215702

PREDICTIVE MAINTANANCE
(Industry 4.0)

https://www2.deloitte.com/content/dam/Deloitte/us/Documents/process-and-operations/us-cons predictive-maintenance.pdf

UNIVERSITÀ DEGLI STUDI DI TERAMO

Figure Professionali

Il mondo delle basi di dati può essere esplorato da quattro punti di vista:

- Utente → come interagire con un DB (aggiungere/modificare informazioni, recuperare informazioni, etc)
- Progettista → come progettare un DB
- Programmatore → come sviluppare applicazioni Web/stand-alone che si interfaccino con un DB
- Analista → come reperire informazioni da un DB attraverso tecniche di data-mining

Figure Professionali

- Power User in vari ambiti aziendali
- Database Administrator
- Progettista di Database
- Analista/Sviluppatore SQL
- Database Specialist
- Business Intelligence Specialist
- Data Scientist
- …

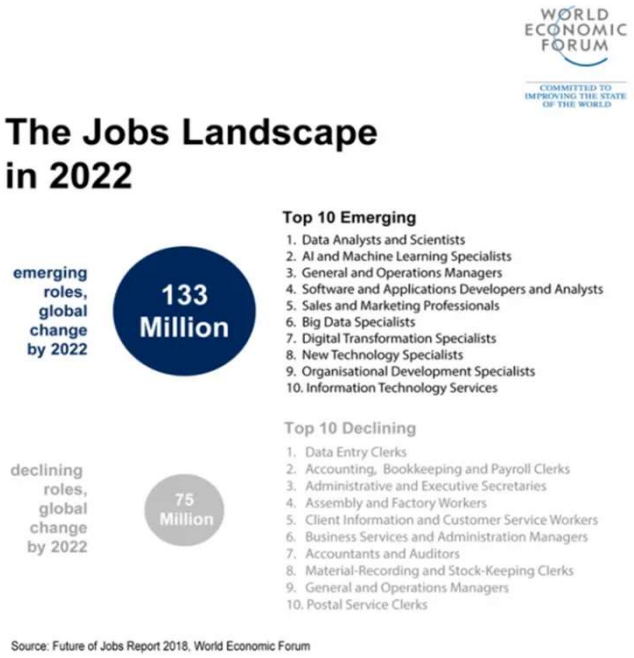
Figure Professionali


NUOVE PROFESSIONI

Perché serve un data scientist per ogni azienda

di Alberto Magnani

https://www.ilsole24ore.com/art/tecnologie/2017-01-13/perche-serve-data-scientist-ogni-azienda-181239.shtml?uuid=ADkcBTYC



UNIVERSITÀ DEGLI STUDI DI TERAMO

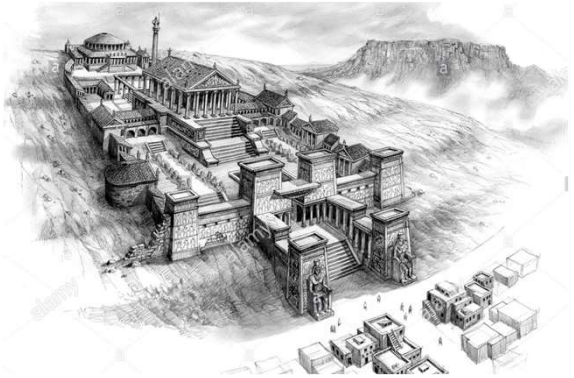
Introduzione ai Database Management Systems (DBMS)

Sistemi Informativi

Un **Sistema Informativo** (**SI**) è una componente di un'organizzazione il cui scopo è quello di **gestire le informazioni** utili ai fini dell'organizzazione stessa

Sistemi Informativi

L’esistenza di un **Sistema Informativo** è indipendente dalla sua automazione



Biblioteca reale di Alessandria d’Egitto
IV-I secolo a.C
400000 rotoli presenti

UNIVERSITÀ DEGLI STUDI DI TERAMO

Sistemi Informativi

L’esistenza di un **Sistema Informativo** è indipendente dalla sua automazione

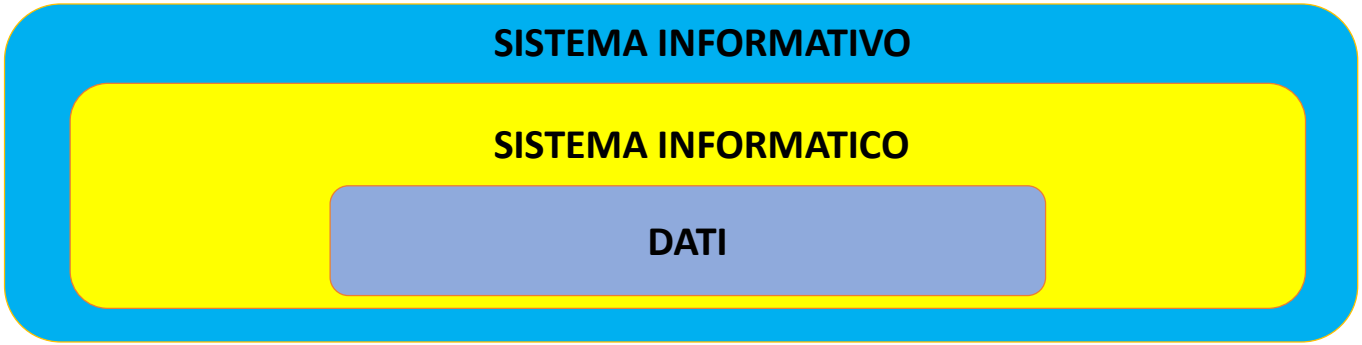
Censimenti e Registro Anagrafe

Nell’Antica Roma, i **censimenti** venivano effettuati dalla fine del IV secolo a.c. Gli **elenchi dei censiti**, distinti secondo il possesso o meno dei diritti civili e politici, la classe patrimoniale e l'età, venivano utilizzati come liste elettorali e per determinare la ruoli per l’esenzione dei tributi e le liste di leva.

UNIVERSITÀ DEGLI STUDI DI TERAMO

Sistemi Informativi

La porzione **automatizzata** di un sistema informativo prende il nome di **Sistema Informatico**



UNIVERSITÀ DEGLI STUDI DI TERAMO

Approcci alla Gestione dei Dati

Gran parte dei sistemi informatici hanno necessità di gestire **dati in maniera persistente**

Persistente: Dati memorizzati su memoria non volatile

APPROCCI di GESTIONE

- Approccio **convenzionale** (basato su files)
- Approccio **strutturato** (basato su software di gestione dei dati)

Approcci alla Gestione dei Dati

Gran parte dei sistemi informatici hanno necessità di gestire **dati in maniera persistente**

Persistente: Dati memorizzati su memoria secondaria

APPROCCI di GESTIONE

- Approccio **convenzionale** (basato su files)
- Approccio **strutturato** (basato su software di gestione dei dati)

Approccio basato su files

Operazioni di **Lettura/Scrittura** su file mediante supporto del Sistema Operativo

- Nessuna chiara distinzione tra **dati ed applicazioni**
- L'applicazione contiene al suo interno la **logica di gestione e memorizzazione** dei dati stessi (es. formato dei dati)
- Il **Sistema Operativo** offre le primitive di base per l'accesso ai files ed i meccanismi di sicurezza del **file-system**

Approccio basato su files

PROBLEMA #1: Gestione di **grandi quantità di dati**

59 Milioni di clienti iscritti
Oltre 42 Terabyte di dati

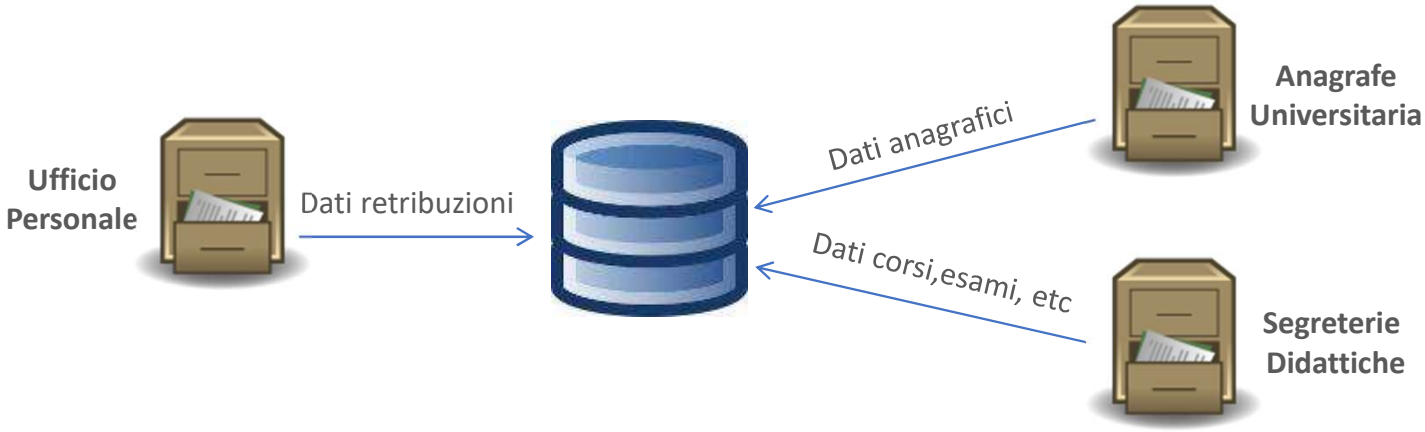
323 Terabyte di dati
1.9 trillioni di record relative a chiamate

Ovvi problemi di **scalabilità** ed **efficienza**

Approccio basato su files

PROBLEMA #2: **Condivisione** ed **accesso** concorrente

In molti scenari, i dati devono essere a disposizione di una moltitudine di utenti/applicazioni per accessi concorrenti.



UNIVERSITÀ DEGLI STUDI DI TERAMO

Approccio basato su files

PROBLEMA #2: **Condivisione** ed **accesso** concorrente

LIMITAZIONI

- Accesso a file condivisi avviene attraverso le politiche di accesso del file-system → Lock a livello di file, **bassa granularità di concorrenza**, prestazioni limitate!
- Applicazioni diverse devono conoscere l'esatta collocazione e formato dei dati → **Aggiornamento del formato dei dati**?
- In alternativa: replica dei dati presso i vari sistemi/utenti che ne fanno utilizzo → **Consistenza delle repliche**?

Approcci alla Gestione dei Dati

Gran parte dei sistemi informatici hanno necessità di gestire **dati in maniera persistente**

Persistente: Dati memorizzati su memoria non volatile

APPROCCI di GESTIONE

- Approccio **convenzionale** (basato su files)
- Approccio strutturato (basato su software di gestione dei dati)

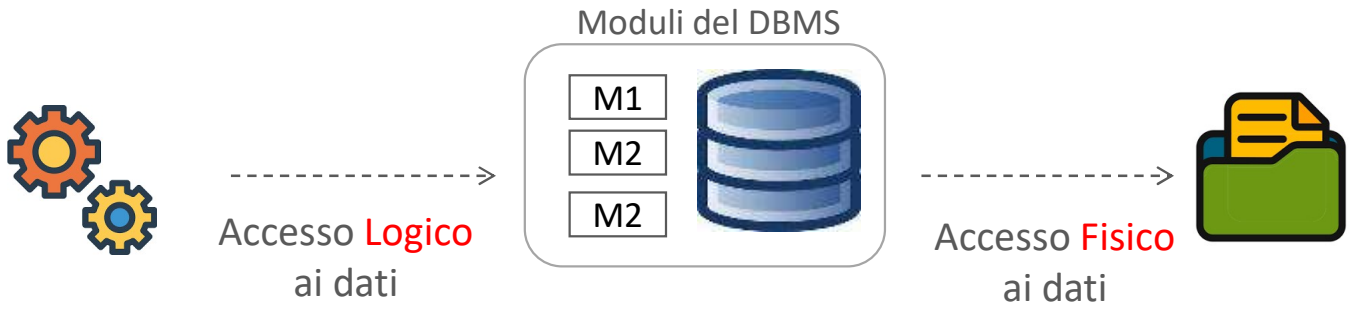
DBMS

Un **DBMS** è un sistema software che è in grado di gestire collezioni di dati *grandi, condivise e persistenti, in maniera efficiente e sicura*

ALCUNE FUNZIONALITA'

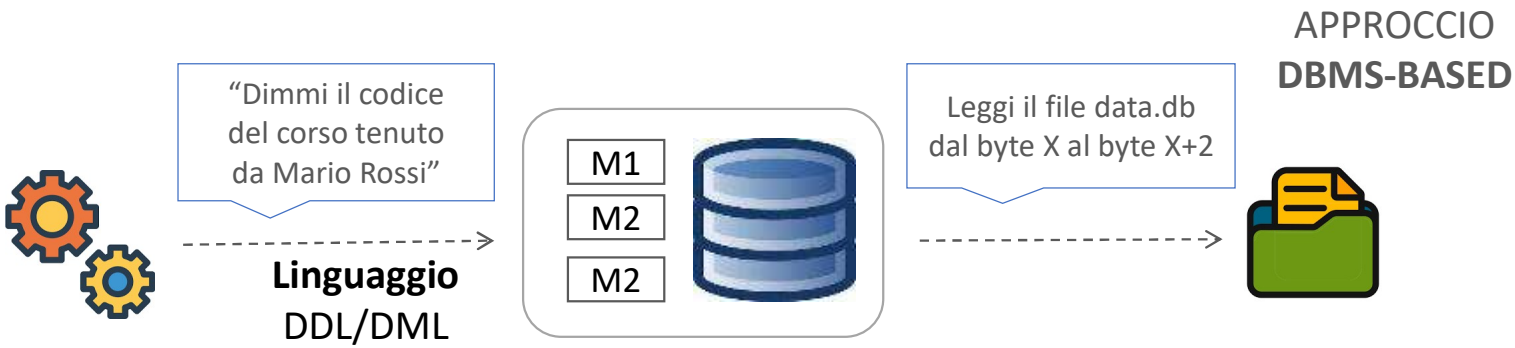
- Creazione di una base di dati e memorizzazione su memoria secondaria
- Accesso in lettura/scrittura ad i dati
- Condivisione di dati tra diversi utenti/applicazioni
- Protezione dei dati da accessi non autorizzati
- Reliability dei dati in caso di guasti (hardware/software)
- …

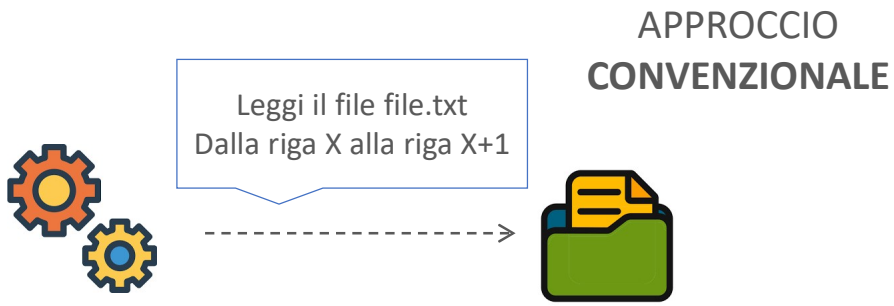
Separazione Dati/Applicazioni



- Tramite i DBMS, è possibile implementare un **paradigma** di **separazione di dati ed applicazioni**
- Le applicazioni non necessitano di conoscere la struttura fisica dei dati (es. come e dove sono memorizzati su disco) ma **solo la struttura logica** (cosa rappresentano)

Separazione Dati/Applicazioni





Lista (Parziale) dei DBMS più Usati

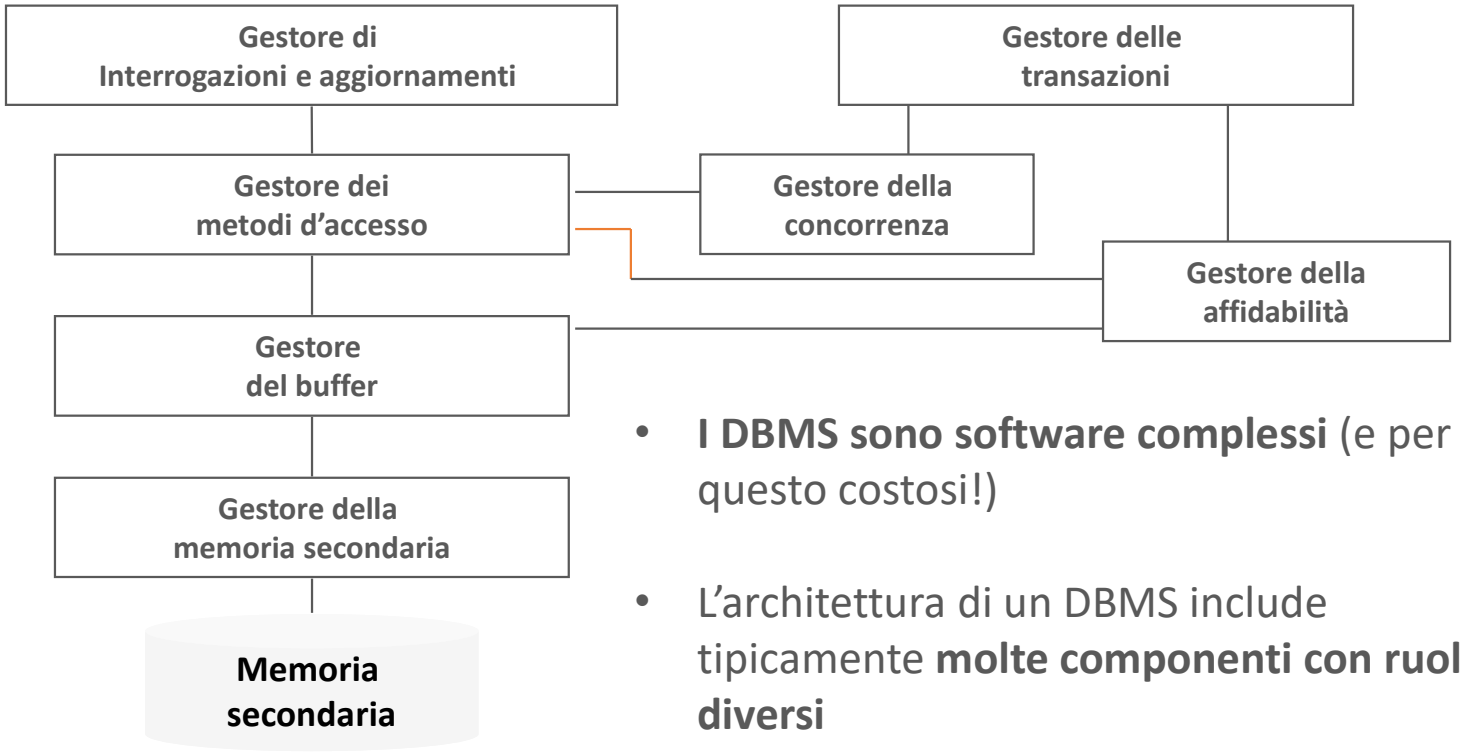
4th Dimension
Adabas D
Alpha Five
Apache Derby
Aster Data
Altibase
BlackRay
CA-Datacom
Clarion
Clustrix
CSQL
CUBRID
Daffodil database
DataEase
Database Management
Dataphor
Java DB
Empress Embedded Database
EnterpriseDB
eXtremeDB
FileMaker Pro
Firebird
Greenplum
GroveSite
HanaDB (SAP)
H2
Helix database
HSQLDB
IBM DB2
IBM Domino Approach
Infobright
Informix
Ingres
InterBase
InterSystems Caché
GT.M
Linter
MariaDB
MaxDB
Microsoft Access
Microsoft Jet Database Engine (part of Microsoft Access)
Microsoft SQL Server

UNIVERSITÀ DEGLI STUDI DI TERAMO

Lista (Parziale) dei DBMS più Usati

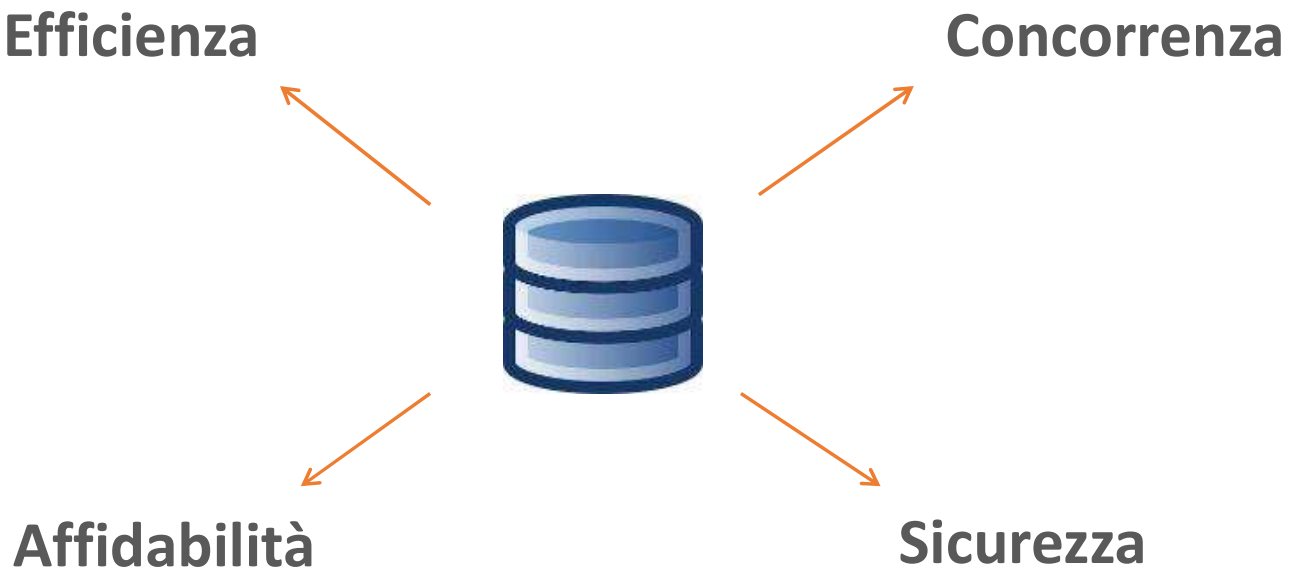
Microsoft SQL Server Express
Microsoft Visual FoxPro
Mimer SQL
MonetDB
mSQL
MySQL
Netezza
NonStop SQL
Openbase
OpenLink Virtuoso
OpenLink Virtuoso Server
OpenOffice.org Base
Oracle
Oracle Rdb for OpenVMS
Panorama
PostgreSQL
Progress Software
RDM Server
SAND CDBMS
Sav Zigzag
ScimoreDB
SmallSQL
SQLBase
SQLite
Sybase SQL AS
Teradata
TimesTen
txtSQL
Unisys RDMS 2200
UniData
UniVerse
Vertica
VMDS
VISTADB

Componenti di un DBMS



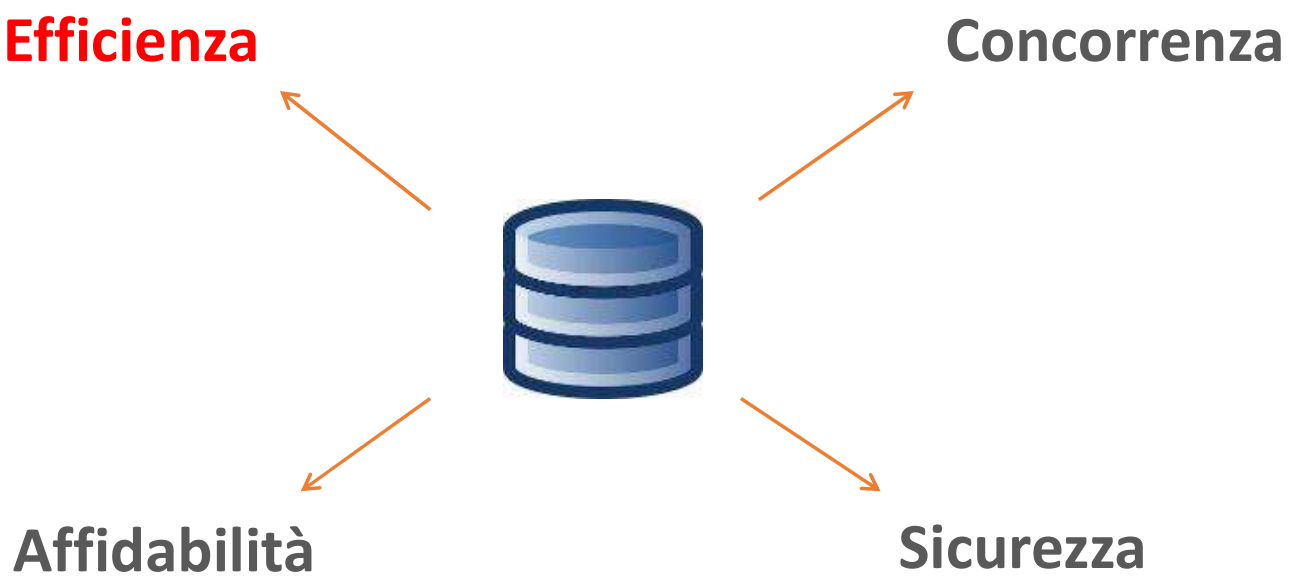
- **I DBMS sono software complessi** (e per questo costosi!)
- L'architettura di un DBMS include tipicamente **molte componenti con ruoli diversi**

Caratteristiche di un DBMS



UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS



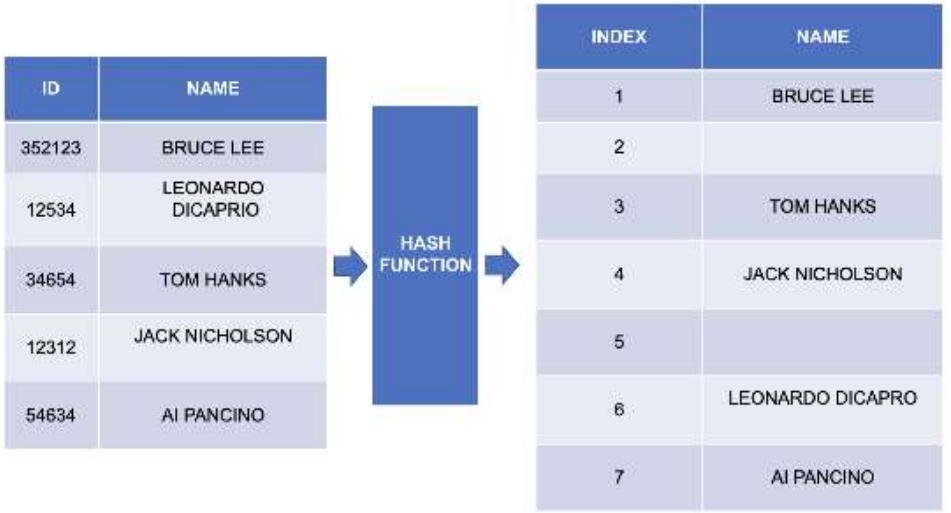
UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS

Efficienza

DBMS forniscono adeguate **strutture dati** per organizzare I dati all'interno dei file, e per supportare le operazioni di ricerca/aggiornamento

In genere, parliamo di **strutture dati ad albero o tabelle hash**



UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS

Efficienza

Indicizzazione:
Creazione di una struttura che contiene informazioni sulla posizione di memorizzazione delle tuple sulla base del valore del campo **chiave**

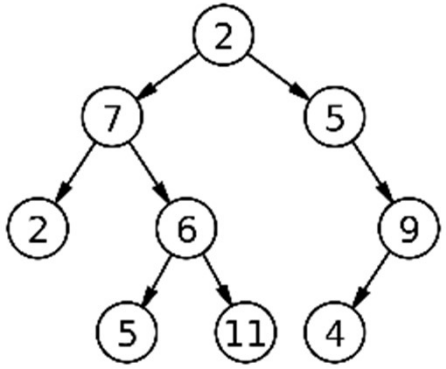
Q. A cosa serve un indice?

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS

Efficienza

Binary-Tree



Ricerca → O(log(n))

Inserimento → O(log(n))

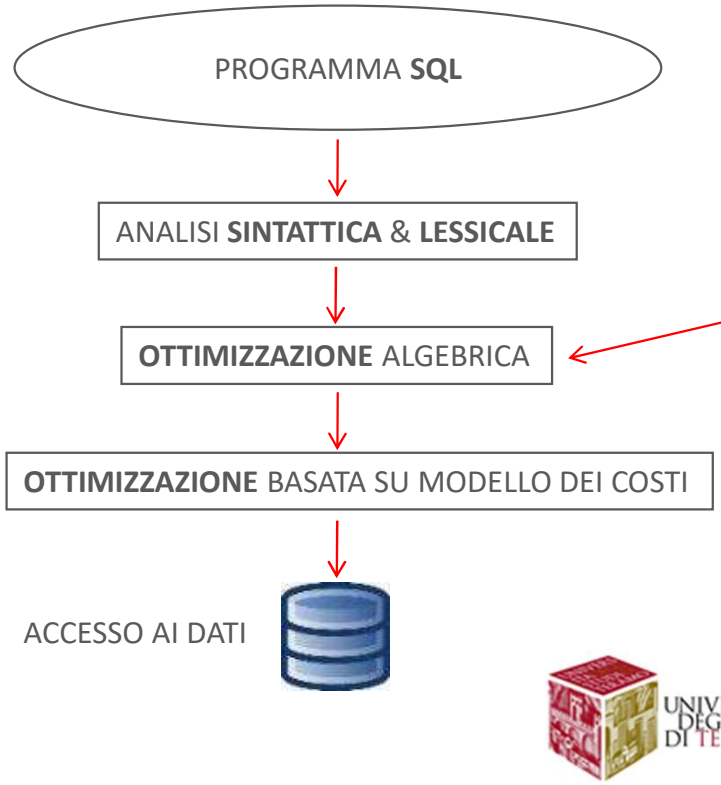
Cancellazione → O(log(n))

Le strutture ad albero dinamiche di tipo B (B-tree) e B+(B+-tree) sono quelle più frequentemente utilizzate per la realizzazione di indici

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS

Efficienza



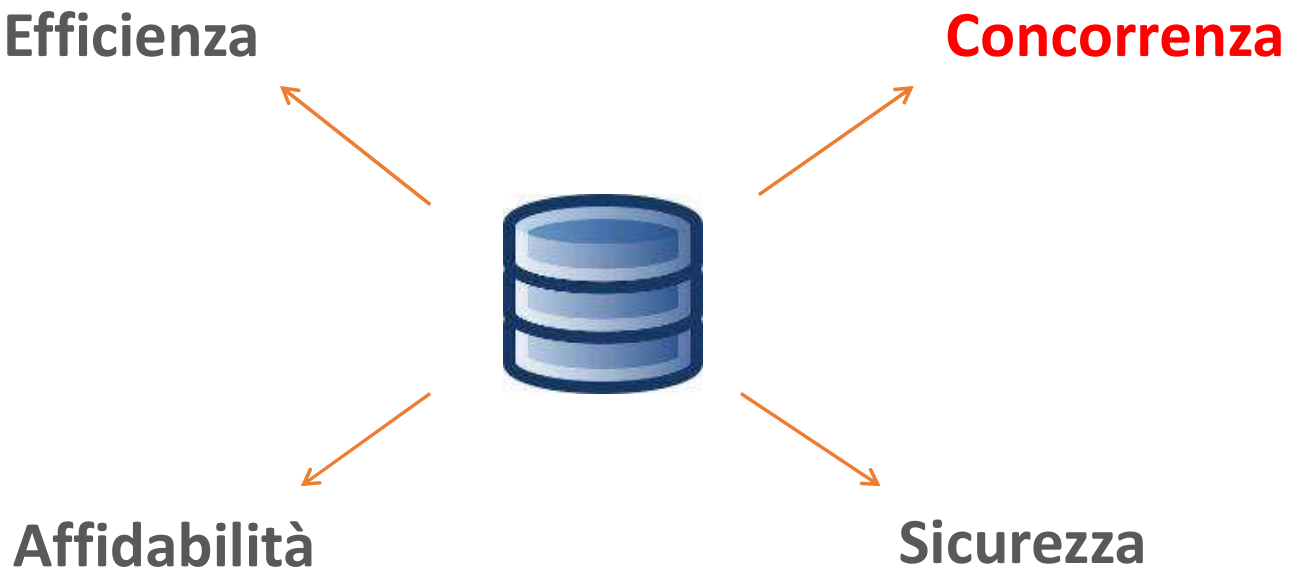
Ottimizzazione di operazioni di ricerca (**interrogazioni**)

La query SQL viene tradotta in una sequenza di operatori algebrici per l'accesso ai dati (**algebra relazionale**).

$$\Pi_{A_1 A_2 .. A_n}(\sigma_{Condizione}(T_1 \triangleright\triangleleft T_2 \triangleright\triangleleft ... \triangleright\triangleleft T_m))$$

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS



UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS
Concorrenza

In molti sistemi è fondamentale **gestire operazioni concorrenti** di accesso ai dati

Processing di più di 42 milioni
di transazioni ogni giorno

Source: PayPal

La maggior parte dei DBMS forniscono un **livello di granularità di locking** più fine di quello convenzionale (a livello di tabella, pagina, o singola entry)

Caratteristiche di un DBMS
Concorrenza

Un DBMS deve garantire il fatto che **accessi da parte di applicazioni diverse non interferiscano tra loro**, lasciando il sistema in uno **stato inconsistente**

Es. Sistema informativo dei conti bancari

2 richieste da gestire al tempo t:

Prelievo di 100 euro dal conto X

Prelievo di 80 euro dal conto X

Saldo del conto X at tempo t: 120 euro

Caratteristiche di un DBMS
Concorrenza

ESEMPIO di ESECUZIONE (**non corretta**!!)

OP1
Leggi X
Calcola X-100
Scrivi X

OP2
Leggi X
Calcola X-80
Scrivi X

Schedule	**Valore X**
OP1: Leggi X	120
OP2: Leggi X	120
OP1: Calcola X-100	20
OP2: Calcola X-80	40
OP1: Scrivi X	20
OP2: Scrivi X	**40 (????)**

Per prevenire tali situazioni, i DBMS implementano **algoritmi di controllo della concorrenza** tali che operazioni sui dati (**transazioni**) eseguite in concorrenza producano lo stesso risultato di un'esecuzione seriale

Caratteristiche di un DBMS
Concorrenza

Lock Manager

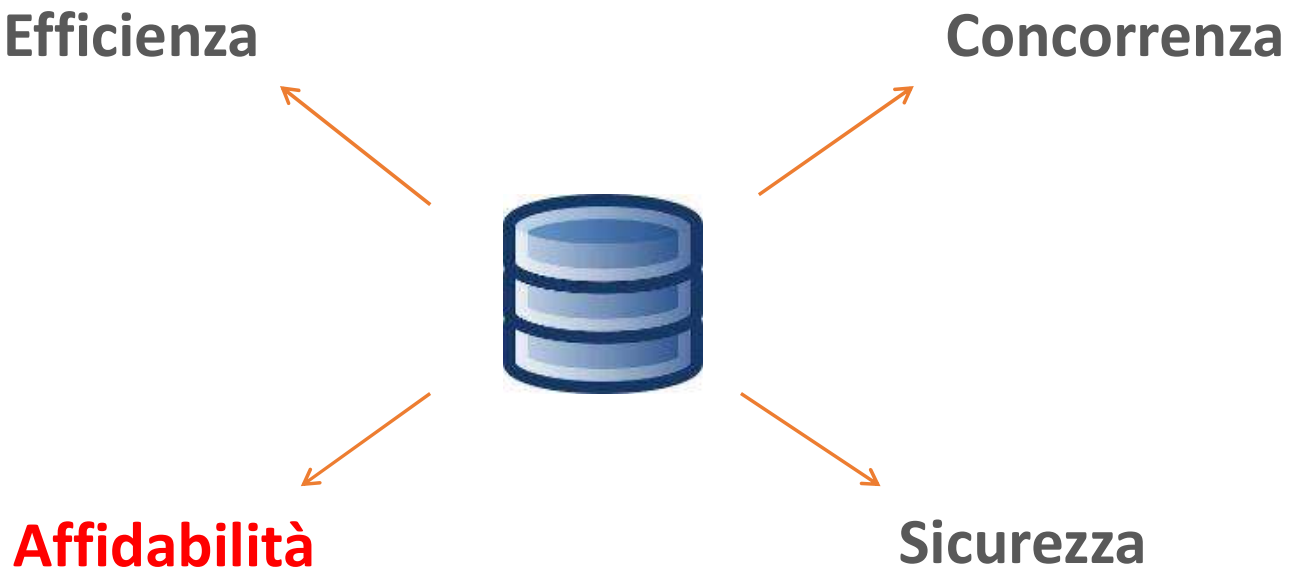
Componente del DBMS responsabile di gestire i lock alle risorse del DB, e di rispondere alle richieste delle transazioni.

OP1
Lock(x)
Leggi X
Calcola X-100
Scrivi X
Unlock(x)

OP2
Lock(x)
Leggi X
Calcola X-80
Scrivi X
Unlock(x)

- Utilizzo di **lock** in lettura/scrittura per accesso a risorse condivise (dati)
- Algoritmi (**2PL**, **S2PL**) per gestire ordine di acquisizione dei lock

Caratteristiche di un DBMS



UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS
Affidabilità

Alcune operazioni sui dati sono **particolarmente delicate**, e devono essere gestite in maniera opportuna, secondo la regola del **tutto o niente**

Es. Trasferimento di denaro (100 €) dal conto X al conto Y.

Op1: X=X-100

Op2: Y=Y+100

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS
Affidabilità

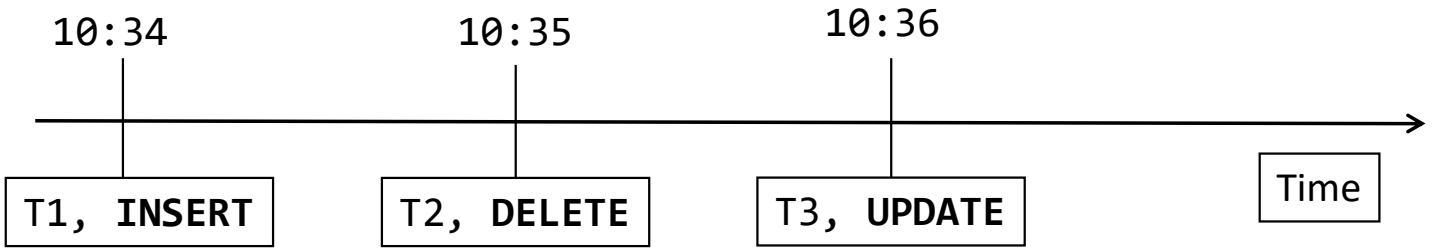
I DBMS devono fornire appositi strumenti per **annullare operazioni non completate** e fare **roll-back** dello stato del sistema

In molti casi i DBMS mettono a disposizione appositi **strumenti ed algoritmi per garantire la persistenza dei dati** anche in presenza di **malfunzionamenti hardware/software**

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS
Affidabilità

Il controllore di affidabilità utilizza dei **log**, nel quale sono indicate tutte le **operazioni svolte dal DBMS**



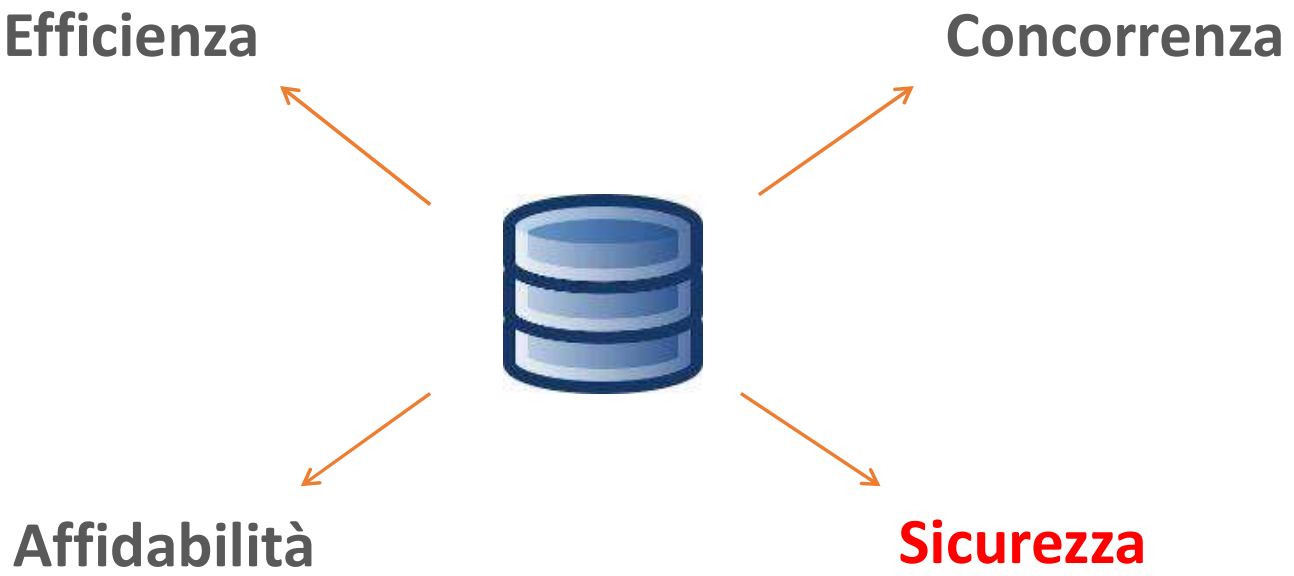
Algoritmi ad-hoc (es. algorimo di ripresa a caldo/a freddo) per ripristinare lo stato dei dati a partire dai log del DBMS (do/undo delle operazioni)

Caratteristiche di un DBMS
Affidabilità

Transazione
Operazioni atomica eseguita su un DB dal DBMS

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS



Caratteristiche di un DBMS
Multi-Utenza e Sicurezza

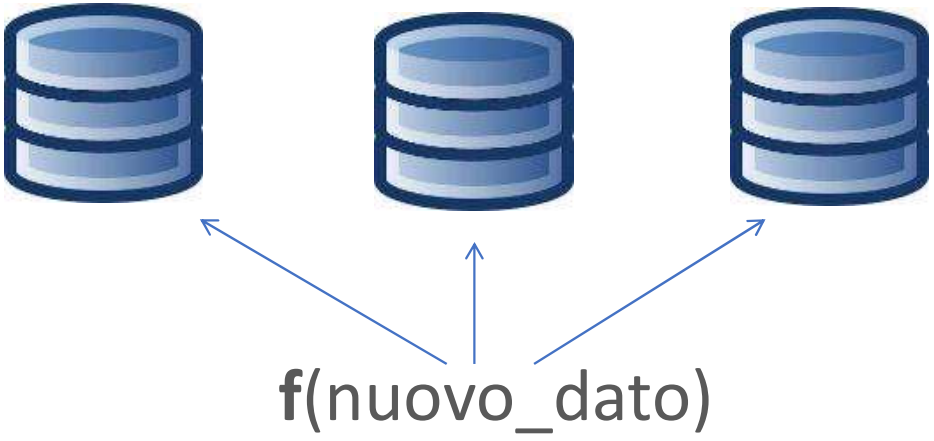
La maggior parte dei DBMS implementa **politiche di controllo degli accessi** ad i dati mediante **sistemi di permessi**:

- Quali operazioni sono consentite all'utente X?
- Quali dati appartengono all'utente X?

USER	OPERATION	DATA	AUTHORIZATION
User X	Read	User X Income	**GRANT**
User X	Read	User Y Income	**DENY**
User Y	Write	User Y Income	**DENY**

Caratteristiche di un DBMS
Scalabilità (Orizzontale)

Possibilità di gestire grandi moli di dati aumentando il **numero di nodi** usati per lo storage (**database distribuito**)

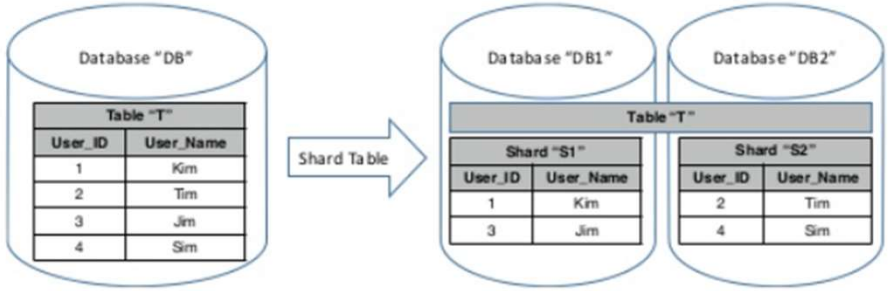


La funzione di **sharding** determina la politica di distribuzione dei dati tra i nodi

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS
Scalabilità (Orizzontale)

1. Meccanismi di **Load-balancing**
2. Meccanismi di **Gestione delle repliche** dati

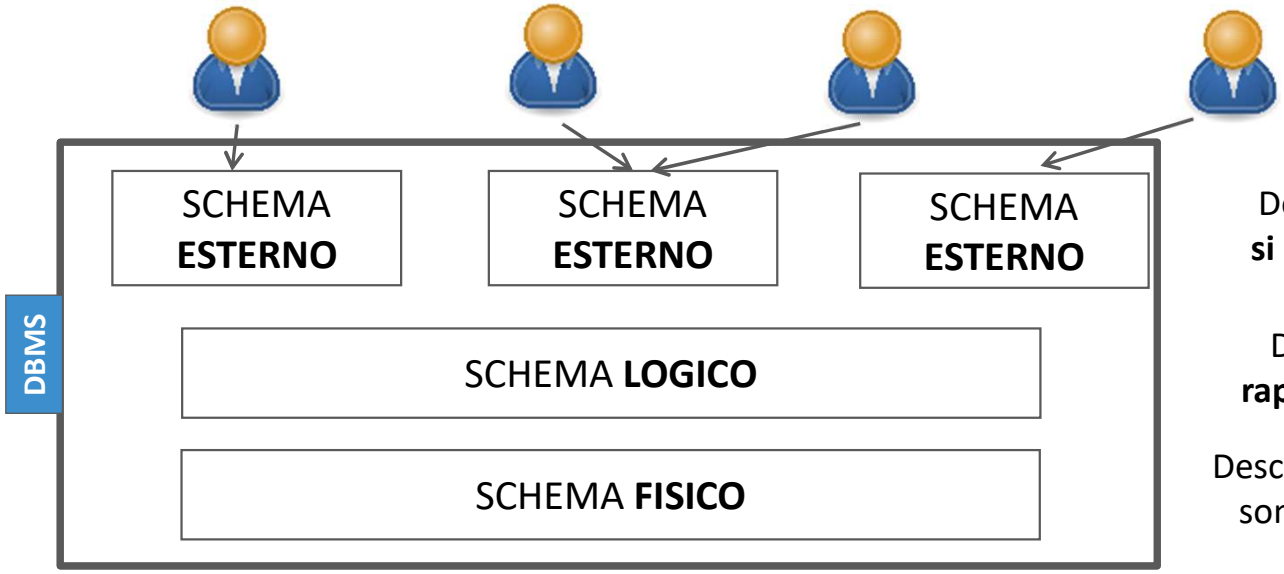


PROBLEMA: Come gestire la **consistenza** delle repliche dati in presenza di partizionamenti della rete e perdita di messaggi? -> **CAP Theorem**

UNIVERSITÀ DEGLI STUDI DI TERAMO

Caratteristiche di un DBMS

Un DBMS può essere visto come un'**architettura software** a 3 livelli



Descrive **come si presenta** il DB

Descrive **cosa rappresenta** il DB

Descrive **come/dove** sono memorizzati i dati

UNIVERSITÀ DEGLI STUDI DI TERAMO

Livello Logico di un DBMS

I DBMS possono differire sulla base del modello logico dei dati che supportano

- Modello **Relazionale** (di fatto, il più usato)
- Modello Gerarchico
- Modello Reticolare
- Modello ad Oggetti
- Approcci NoSQL (diversi)

Livello Logico di un DBMS

Esempio Modello Relazionale

Base di dati che gestisce le informazioni relative alla programmazione didattica di un Corso di Laurea: elenco corsi, con numero ore, semestre, crediti, nome e codice identificativo di ciascun corso

Codice	Nome	NumOre	Semestre	Crediti
010	Basi di Dati	72	1	9
001	Algoritmi	90	1	12

SCHEMA

ISTANZE

Nel modello relazionale, i dati sono organizzati in tabelle

Livello Esterno di un DBMS

Il livello esterno consente di avere **viste** personalizzate della base di dati da parte di diversi utenti/applicazioni!

Codice	Nome	Cognome	Data Nascita	Livello	Stipendio
001	Marco	Rossi	10/10/1970	1	24000
002	Michele	Bianchi	10/10/1970	1	32000

VISTA Ufficio **Anagrafe**

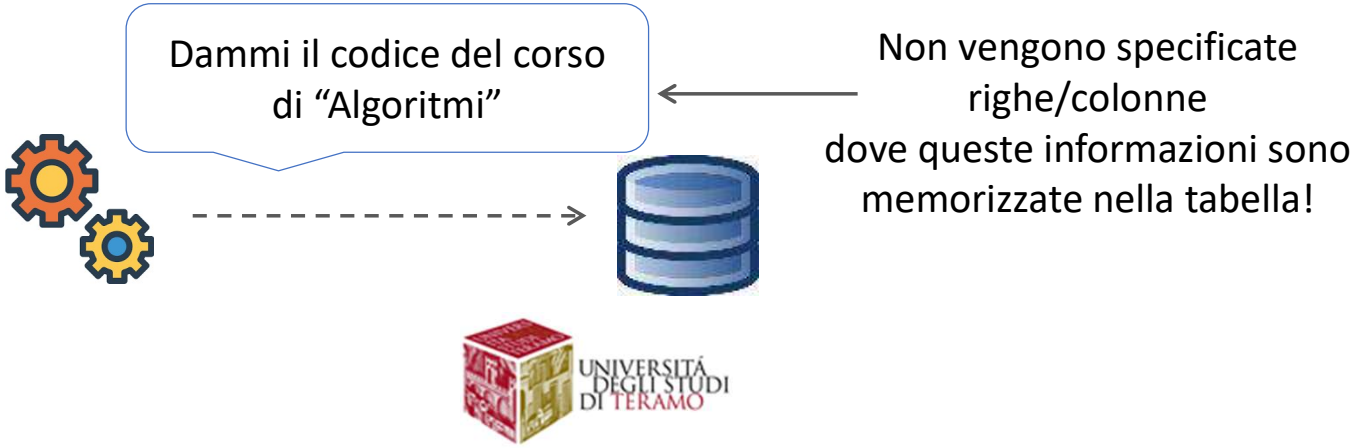
VISTA Ufficio **Stipendi**

Livello Logico di un DBMS

INDIPENDENZA MODELLO LOGICO – MODELLO FISICO

L'organizzazione logica dei dati **non dipende** dalle strutture dati usate per l'effettiva memorizzazione su disco!

In pratica, **le applicazioni accedono al DBMS specificando i concetti logici** del modello dei dati, piuttosto che i dettagli relativi alla loro memorizzazione.



Interazione con un DBMS

Come possono utenti ed applicazioni interagire con un DBMS?

INTERFACCIA GRAFICA

LINGUAGGIO ORIENTATO AI DATI

`SELECT … FROM … WHERE`

LINGUAGGIO INTERNO

$$\Pi_{A_1 A_2 .. A_n}(\sigma_{Condizione}(T_1 \triangleright\triangleleft T_2 \triangleright\triangleleft \ldots \triangleright\triangleleft T_m))$$

Interazione con un DBMS

Come possono utenti ed applicazioni interagire con un DBMS?

Quasi tutti i DBMS mettono a disposizione dei **linguaggi**:

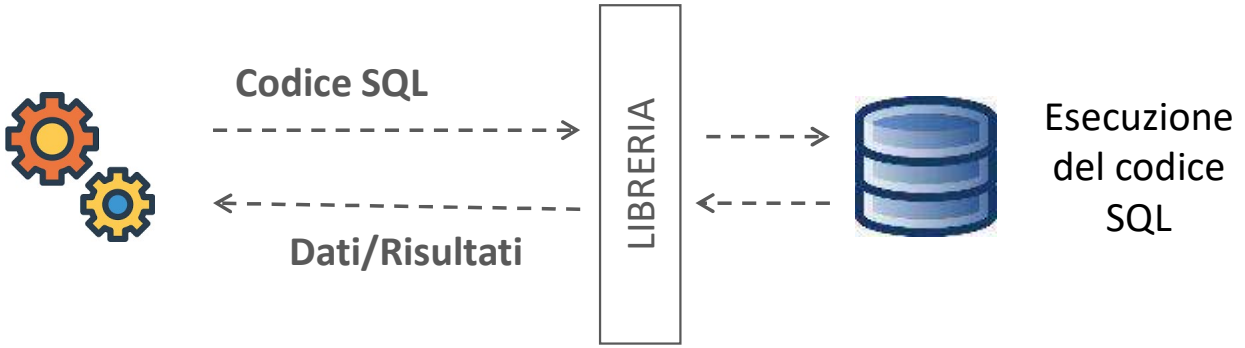
- Definizione dello schema logico (**Linguaggio DDL**)
- Manipolazione delle istanze (**Linguaggio DML**)

Linguaggi **orientati ad i dati**, molto diversi da linguaggi di programmazione “tradizionali” (es. C/C++/Java/etc)!

Interazione con un DBMS

Le applicazioni che si interfacciano con un DBMS:

- Integrano **codice SQL** all'interno del loro codice
- Utilizzano opportune **librerie** (fornite dal DBMS)per gestire la connessione al DBMS



Vantaggi di un DBMS

Quando usare un DBMS in un progetto SW?

- Necessità di gestire **grandi volumi di dati**
- Necessità di costruire **sistemi data-centric** con molte operazioni di accesso ai dati
- Necessità di **condividere dati**, fornendo l'accesso a diversi sistemi SW/applicazioni
- Necessità di garantire la **persistenza dei dati** anche a fronte di possibili guasti e malfunzionamenti HW/SW
- Necessità di implementare **meccanismi di sicurezza** per l'accesso ad i dati in un ambiente multi-utente

Svantaggi di un DBMS

Quando NON usare un DBMS in un progetto SW?

- **Prestazioni:** In alcuni sistemi con richieste di efficienza sull'elaborazione (es. real-time), l'overhead computazionale introdotto dal DBMS può essere eccessivo
- **Costo:** Spese per l'acquisto di DBMS, formazione del personale, amministrazione del DB, etc
- **Complessità:** Applicazioni/sistemi di dimensioni ridotte, single-user e con pochi dati da gestire

Scegliere un DBMS

I DBMS sono tutti uguali?

NO!

Differenze sostanziali, ad esempio in termini di:

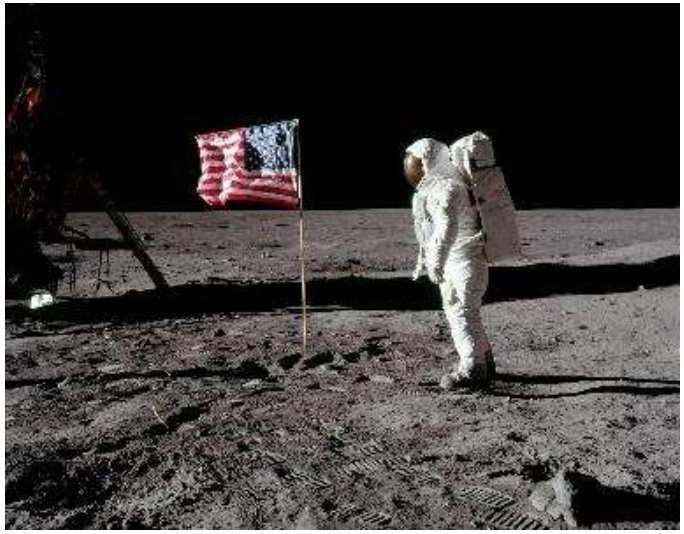
- Modello logico supportato (relazionale? → RDBMS)
- Linguaggio DDL/DML (SQL-2? SQL-3? varianti?)
- Algoritmi di indicizzazione (es. R+ tree?)
- Supporto alla transazioni (es. proprietà ACID?)
- Gestione della concorrenza
- …

DBMS: Un po di storia

Information Management System (**IMS**) è il nome di un software sviluppato da IBM nel 1968

Utilizzato come supporto alle missioni *Apollo* per la gestione dei dati tecnici/amministrativi e delle forniture di materiali

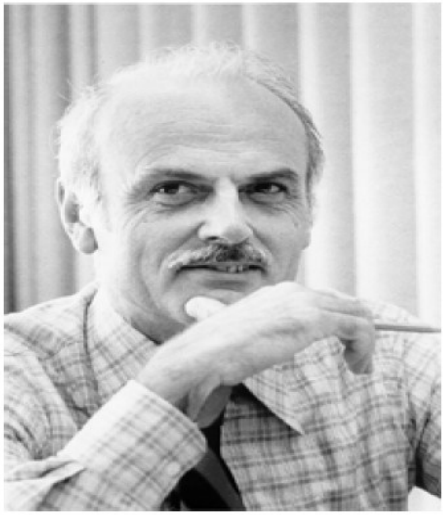
Modello **gerarchico** di gestione dei dati, **motore transazionale** per la concorrenza



DBMS: Un po di storia

Nel 1970, un ricercatore della IBM (**Edgar Codd**) pubblica la sua visione di modello “relazionale” dei dati, basato sul concetto matematico di relazione tra insiemi

Edgar F. Codd *A relational model of data for large shared data banks* Communications of the ACM 13 (6), 377-387. 1970.



UNIVERSITÀ DEGLI STUDI DI TERAMO

DBMS: Un po di storia

Anni ’70: IBM lavora allo sviluppo di un linguaggio basato sul modello relazionale (SQL), ed all’implementazione di un RDBMS sperimentale (**System R → SQL/DS→ DB2)**

Nel 1979, una piccola startup (**Relational Software Inc**) produce un primo esempio di RDBMS commerciale. Nel 1982, Relational Software cambia il proprio nome in **Oracle Corporation**

DBMS: Un po di storia

Anni ’80: compaiono i primi **DBMS basati sul modello ad oggetto (ORDBMS)**, che cercano di emulare il successo del paradigma di programmazione ad oggetti, e facilitare l’integrazione tra DBMS e linguaggi di alto livello (es. C++/Java)

Viene sviluppata **OQL** – omologo di SQL per il paradigma ad oggetti

Nonostante ciò ORDBMS restano poco diffusi

DBMS: Un po di storia

Nel 2019, **il mercato dei RDBMS ha avuto una crescita del 17.4% con ricavi complessivi pari a 54 miliardi di dollari**

Il mercato dei RDBMS tradizionali è dominato da 4 vendor

- Oracle
- IBM
- Microsoft
- SAP

Fonte: IDC

DBMS: Un po di storia

Figure 1: Magic Quadrant for Cloud Database Management Systems

Source: Gartner (November 2020)

UNIVERSITÀ DEGLI STUDI DI TERAMO

DBMS: Un po di storia

Una delle nuove linee evolutive dei DBMS, da qualche anno, è rappresentata dall'approccio **NoSQL**

Idea di base: superare la rigidità del modello relazionale nella definizione dello schema, consentendo una più facile espansione del DB in termini di dati, e di computazione distribuita

Molti approcci sotto la definizione NoSQL:

Es. **Apache Cassandra**, **Apache CouchDB**, etc…

Il Modello Relazionale

Modelli Logici

I DBMS possono differire sulla base del modello logico dei dati che supportano:

- **Modello Relazionale**
- Modello Gerarchico
- Modello Reticolare
- Modello ad Oggetti
- Modelli scheme-less (famiglia di approcci)

Modello Relazionale: Overview

- Proposto nel 1970 da **E.F. Codd**, ricercatore dell'IBM di San Jose, CA.
- Attualmente il più utilizzato tra i tool disponibili
- Garantisce l'indipendenza tra i livelli (esterno/fisico)
- Intuitivo, e basato su nozioni di algebra di base
- DBMS basati sul modello relazionale →
  RDBMS (Oracle, MySQL, DB2, SQL Server etc)

Modello Relazionale: Overview

DEF. Informale

Modello Relazionale → i dati sono organizzati in record di dimensione fissa, e divisi in tabelle (**relazioni**)

Nome	Codice Corso	Nome Docente
Basi di dati	0121	A. Stuardi
Programmazione	1213	S. Clementoni
Sistemi Operativi	1455	F. Figliola

- *Colonne* della tabella → **Attributi**
- *Intestazione* della tabella → **Schema** della relazione
- *Righe* della tabella → **Istanze** della relazione

Modello Relazionale: Overview

CORSI

Nome	Codice Corso	Nome Docente
Basi di dati	0121	A. Stuardi
Programmazione	1213	S. Clementoni
Sistemi Operativi	1455	F. Figliola

Nome della relazione: *CORSI*

Attributi*: Nome, Codice del Corso, Nome Docente*

Schema della relazione: *CORSI(Nome, Codice del Corso, Nome Docente)*

Istanze della relazione: es. *<Basi di dati, 0121, A. Stuardi>*

Modello Relazionale: Overview

CORSI

Nome	Codice Corso	Nome Docente
Basi di dati	0121	A. Stuardi
Programmazione	1213	S. Clementoni
Sistemi Operativi	1455	F. Figliola

VINCOLI sull'**ordine** dei dati:

- L'ordinamento delle righe è irrilevante
- L'ordinamento delle colonne è irrilevante

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

CORSI

Nome	Codice Corso	Nome Docente	Nome Docente
11/01/2012	0121	A. Stuardi	A. Stuardi
Programmazione	0123	S. Clementoni	S. Clementoni
Sistemi Operativi	1455	F. Figliola	F. Figliola
Sistemi Operativi	1455	F. Figliola	F. Figliola

(1)

(3)

(2)

VINCOLI sui dati della relazione

- Non possono esistere **attributi uguali** (**1**)
- Non possono esistere **righe uguali** (**2**)
- I dati di una **colonna** devono essere **omogenei** (**3**)

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

E’ possibile avere uno schema di relazioni **senza istanze** (es. in fase di creazione del DB)

CORSI

Nome	Codice Corso	Nome Docente

Il viceversa è impossibile!

…	…	…	…
Sistemi Operativi	1455	F. Figliola	S. Clementoni
Sistemi Operativi	1451	F. Figliola	S. Clementoni

Modello Relazionale: Overview

CORSI

Nome	Codice Corso	Nome Docente
Basi di dati	0121	A. Stuardi
Programmazione	1213	S. Clementoni
Sistemi Operativi	1455	F. Figliola

Ogni attributo dispone di un **DOMINIO** che definisce l'insieme di valori validi per quell'attributo.

Es. dom(Nome) = *string*

E' possibile avere **domini ripetuti** nella stessa relazione!

Normalizzazione

Normalizzazione: procedimento volto alla eliminazione della ridondanza informativa e del rischio di incoerenza della base dati

CORSI

Relazione **NON** normalizzata

Nome	Codice Corso	Info Docente
Basi di dati	2121	A. Stuardi, Professore, Codice: 13435
Programmazione	1213	S. Clementoni, Professore, Codice: 6575
Sistemi Operativi	1455	F. Figliola, Professore, Codice: 43242

Normalizzazione

Si può dire che il processo di normalizzazione garantisce la "**qualità**" dello schema relazionale

Esistono varie forme normali:

- 1NF
- 2NF
- 3NF
- Forma normale di Boyce e Codd (BCNF)
- 4NF
- 5NF

Normalizzazione

Ad esempio, una relazione si dice in **Prima Forma Normale** (1FN) se tutti gli attributi sono definiti su domini atomici e non su domini complessi

CORSI

Relazione Normalizzata (**1FN**)

Nome	Codice Corso	Docente	Ruolo	Codice Docente
Basi di dati	2121	A. Stuardi	Professore	13435
Programmazione	1213	S. Clementoni	Professore	6575
Sistemi Operativi	1455	F. Figliola	Professore	43242

Relazione valida ma **non ben progettata**!

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

Q1: Perchè si chiama modello relazionale?

A: Una **relazione sui dati** può essere vista come una **relazione matematica**! (con una leggera variazione).

Q2: Com' è definita una relazione matematica (nella *teoria degli insiemi)*?

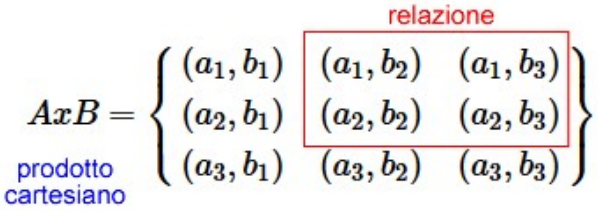
Modello Relazionale: Overview

DEF. Dati n insiemi D_1, D_2, … D_n, una **relazione matematica** sugli insiemi D_1, D_2, … D_n è definita come un **sottoinsieme del prodotto cartesiano** D_1 x D_2 x … x D_n.

DEF. Il **prodotto cartesiano** degli insiemi D_1, D_2, … D_n è definito come **l'insieme delle tuple ordinate** (d_1, d_2, … d_n), con $d_i \in D_i \ \forall i = 1, 2, \ldots, n$

Modello Relazionale: Overview

In matematica una relazione è un sottoinsieme del prodotto cartesiano AxB di due insiemi A e B



A = (a1, a2, a3)
B = (b1, b2, b3)

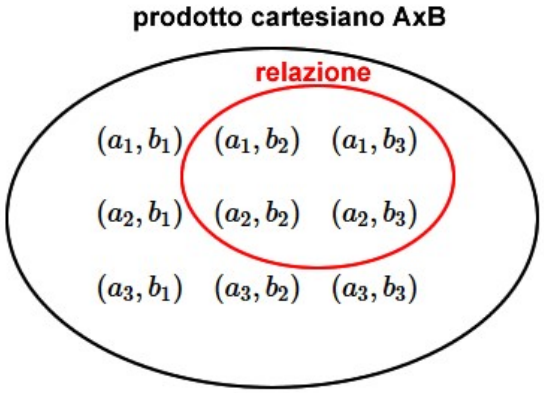
Nota:
In una relazione:

- I domini possono anche essere più di due
- Gli insiemi del prodotto cartesiano possono anche essere uguali tra loro

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

Il prodotto cartesiano degli insiemi A e B è un insieme composto da tutte le coppie ordinate (a,b) dove a∈A e b∈B



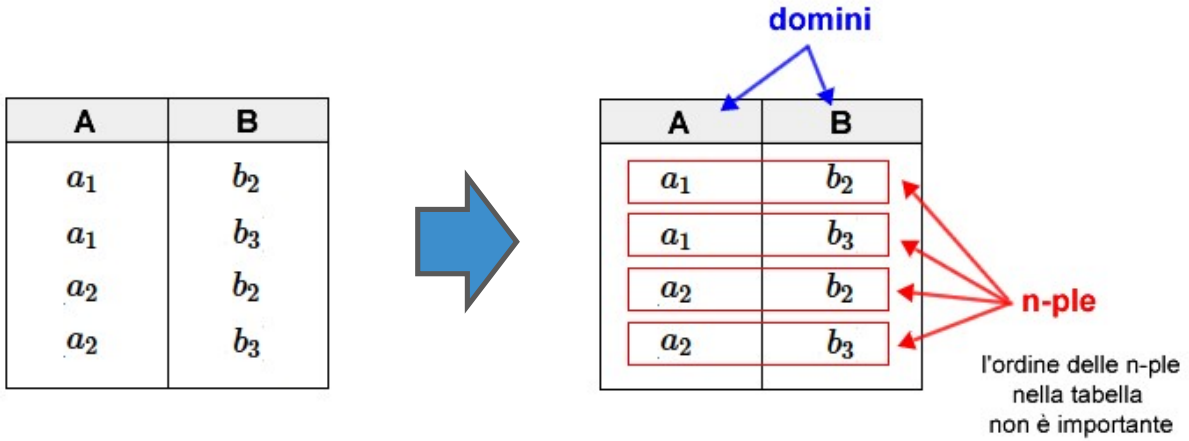
La relazione è un sottoinsieme del prodotto cartesiano AxB composto da n-ple ordinate e distinte tra loro. Ogni n-pla è composta dagli elementi dei due insiemi A e B disposti.

Nota:
Ogni n-pla è ordinata. Vuol dire che in un prodotto cartesiano AxB la prima posizione nella n-pla deve essere sempre un elemento dell'insieme A, mentre la seconda posizione nella n-pla un elemento dell'insieme B.

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

Le n-ple possono essere rappresentate con una tabella



Quindi, la tabella è una rappresentazione della relazione matematica!

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

Consideriamo tre insiemi contenenti diversi elementi: l'insieme delle capitali, delle nazioni e dei continenti (tre domini di tipo stringa)

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

Il prodotto cartesiano dei tre domini è composto dalle 27 combinazioni tra capitale, nazione e continente (3x3x3). Ogni combinazione è una tupla composta da tre elementi di tipo stringa ossia <stringa> x <stringa> x <stringa>

(Roma, Giappone, America)	(Tokyo, Giappone, America)	(Washington, Giappone, America)
(Roma, Giappone, Asia)	(Tokyo, Giappone, Asia)	(Washington, Giappone, Asia)
(Roma, Giappone, Europa)	(Tokyo, Giappone, Europa)	(Washington, Giappone, Europa)
(Roma, Stati Uniti, America)	(Tokyo, Stati Uniti, America)	(Washington, Stati Uniti, America)
(Roma, Stati Uniti, Asia)	(Tokyo, Stati Uniti, Asia)	(Washington, Stati Uniti, Asia)
(Roma, Stati Uniti, Europa)	(Tokyo, Stati Uniti, Europa)	(Washington, Stati Uniti, Europa)
(Roma, Italia, America)	(Tokyo, Italia, America)	(Washington, Italia, America)
(Roma, Italia, Asia)	(Tokyo, Italia, Asia)	(Washington, Italia, Asia)
(Roma, Italia, Europa)	(Tokyo, Italia, Europa)	(Washington, Italia, Europa)

Modello Relazionale: Overview

Consideriamo la seguente relazione (sottoinsieme del prodotto cartesiano tra i 3 insiemi). La relazione è un sottoinsieme composto da tre tuple. Pertanto, ha una cardinalità (numero elementi) pari a 3.

UNIVERSITÀ DEGLI STUDI DI TERAMO

Modello Relazionale: Overview

In un RDBMS posso rappresentare la relazione in una tabella

capitali	nazioni	continenti
Roma	Italia	Europa
Tokyo	Giappone	Asia
Washington	Stati Uniti	America

- Ogni colonna è detta attributo e contiene i dati omogenei del proprio dominio stringa (capitali, nazioni, continenti)
- Ogni riga della tabella è invece una n-pla ordinata detta tupla o record
- Il numero di righe della tabella coincide con la cardinalità

Perché usare la tabella? Per gran parte degli utenti finali la rappresentazione tabellare è sicuramente più semplice e intuitiva della relazione matematica tra insiemi

N.B. A differenza della relazione matematica, l'ordine delle colonne e delle righe nella tabella non è importante

Modello Relazionale: Overview

Nel modello relazionale ogni riga (n-upla, tupla o record) della tabella è diversa dalle altre per almeno un elemento

capitali	nazioni	continenti
Roma	Italia	Europa
Tokyo	Giappone	Asia
Washington	Stati Uniti	America
~~Roma~~	~~Italia~~	~~Europa~~

Nel modello relazionale non può esistere una tupla doppia con gli stessi elementi!

Perché le righe non possono essere uguali? Un insieme non contiene elementi ripetuti. Quindi non possono esistere due elementi uguali in un insieme. Poiché la relazione è sottoinsieme e la tabella è una rappresentazione della relazione, questa regola vale anche per la tabella.

Il Linguaggio SQL

Interazione con un DBMS

Il **modello relazionale** definisce i concetti generali ed i vincoli per modellare e strutturare i dati di una certa applicazione o dominio d'interesse

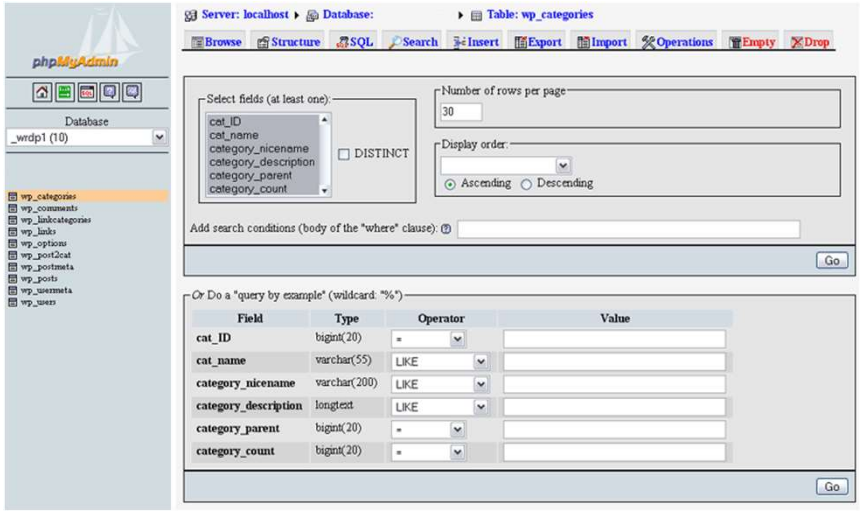
Q. Come implementare il modello relazionale di un DB all'interno di un RDBMS?

Q.1 Come costruire lo schema del DB?
Q.2 Come manipolare le istanze?

A. Attraverso opportuni **linguaggi** data-oriented!

Interazione con un DBMS

1. **Interfacce** grafiche



Creare un nuovo DB
Creare tabelle
Definire vincoli
Inserire istanze
Rimuovere istanze
Cercare istanze
…

UNIVERSITÀ DEGLI STUDI DI TERAMO

Interazione con un DBMS

2. Linguaggi basati sulle proprietà **algebrico/ logiche** del modello relazionale

- Calcolo relazionale sui domini
- Algebra relazionale $\Pi_{A_1 A_2 .. A_n}(\sigma_{Condizione}(T_1 \triangleright\triangleleft T_2 \triangleright\triangleleft ... \triangleright\triangleleft T_m))$

Interazione con un DBMS

3. SQL (*Structured Query Language*)

- SQL-86 → Costrutti base
- SQL-89 (SQL1) → Integrità referenziale
- SQL-92 (SQL2) → SQL Interattivo, sistema tipi
- SQL:1999 (SQL3) → Modello ad oggetti
- SQL:2003 (SQL3) → Nuove parti: SQL/JRT, SQL/XML
- SQL:2006 (SQL3) → Estensione di SQL/XML
- SQL:2008 (SQL3) → Lievi aggiunte
- SQL:2011 (SQL3) -> Supporto ai Temporal Databases
- SQL:2016 (SQL3) -> Gestione del formato JSON

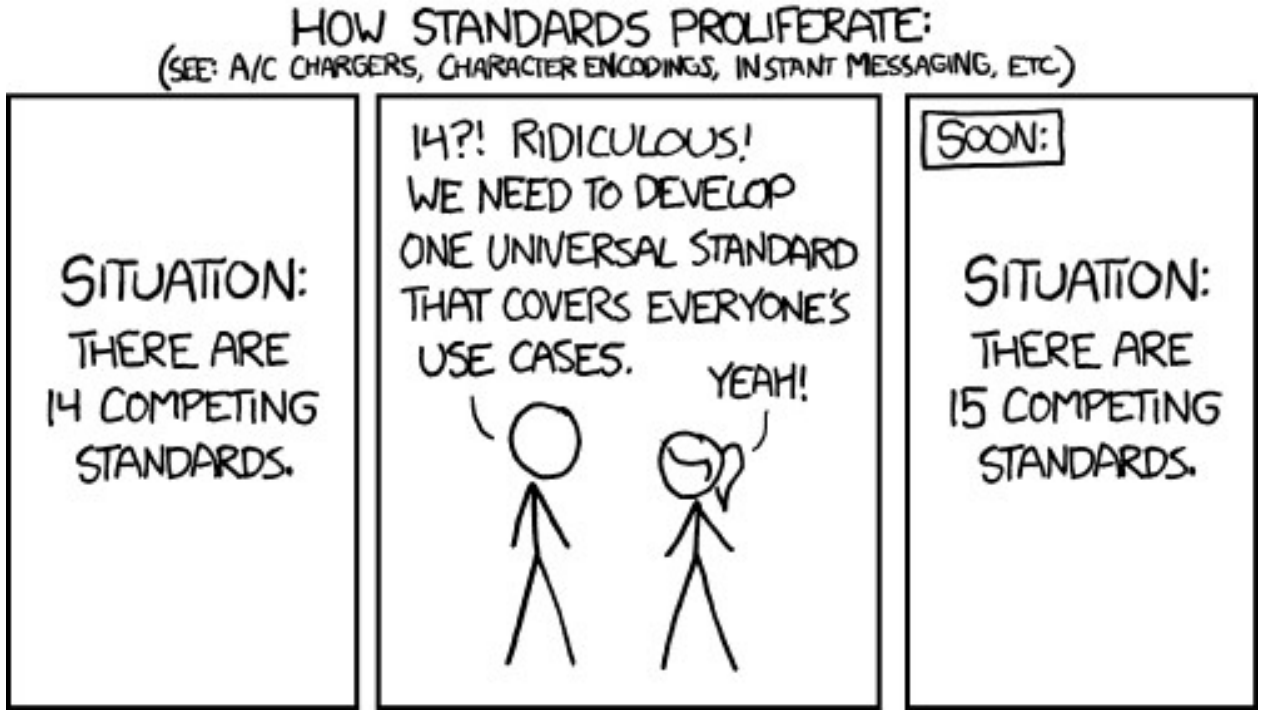
Il Linguaggio SQL

SQL è un linguaggio per DBMS basati sul modello relazionale

Valgono i concetti generali del modello relazionale visto fin qui, ma con qualche differenza:

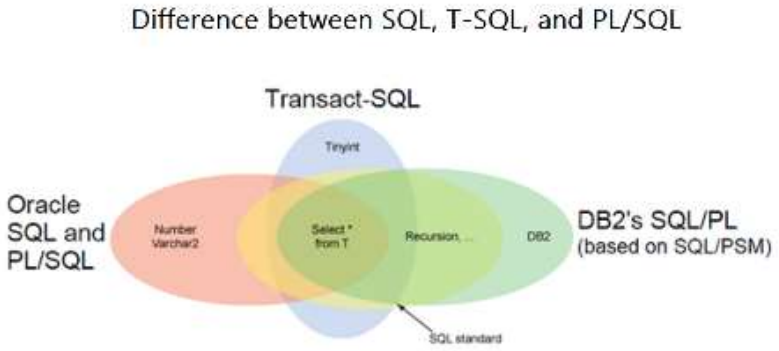
- Si parla di tabelle (e non relazioni)
- Il risultato di un'operazione sui dati può restituire una tabella con righe duplicate
- Il sistema dei vincoli è piu' espressivo
- Il vincolo di integrità referenziale (chiave esterna) è meno stringente

Il Linguaggio SQL: Dialetti



UNIVERSITÀ DEGLI STUDI DI TERAMO

Il Linguaggio SQL: Dialetti



Source	Common name	Full name
ANSI/ISO Standard	SQL/PSM	SQL/Persistent Stored Modules
Interbase / Firebird	PSQL	Procedural SQL
IBM DB2	SQL PL	SQL Procedural Language (implements SQL/PSM)
IBM Informix	SPL	Stored Procedural Language
Microsoft / Sybase	T-SQL	Transact-SQL
Mimer SQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
MySQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)

UNIVERSITÀ DEGLI STUDI DI TERAMO

Il Linguaggio SQL

Due **componenti** principali:

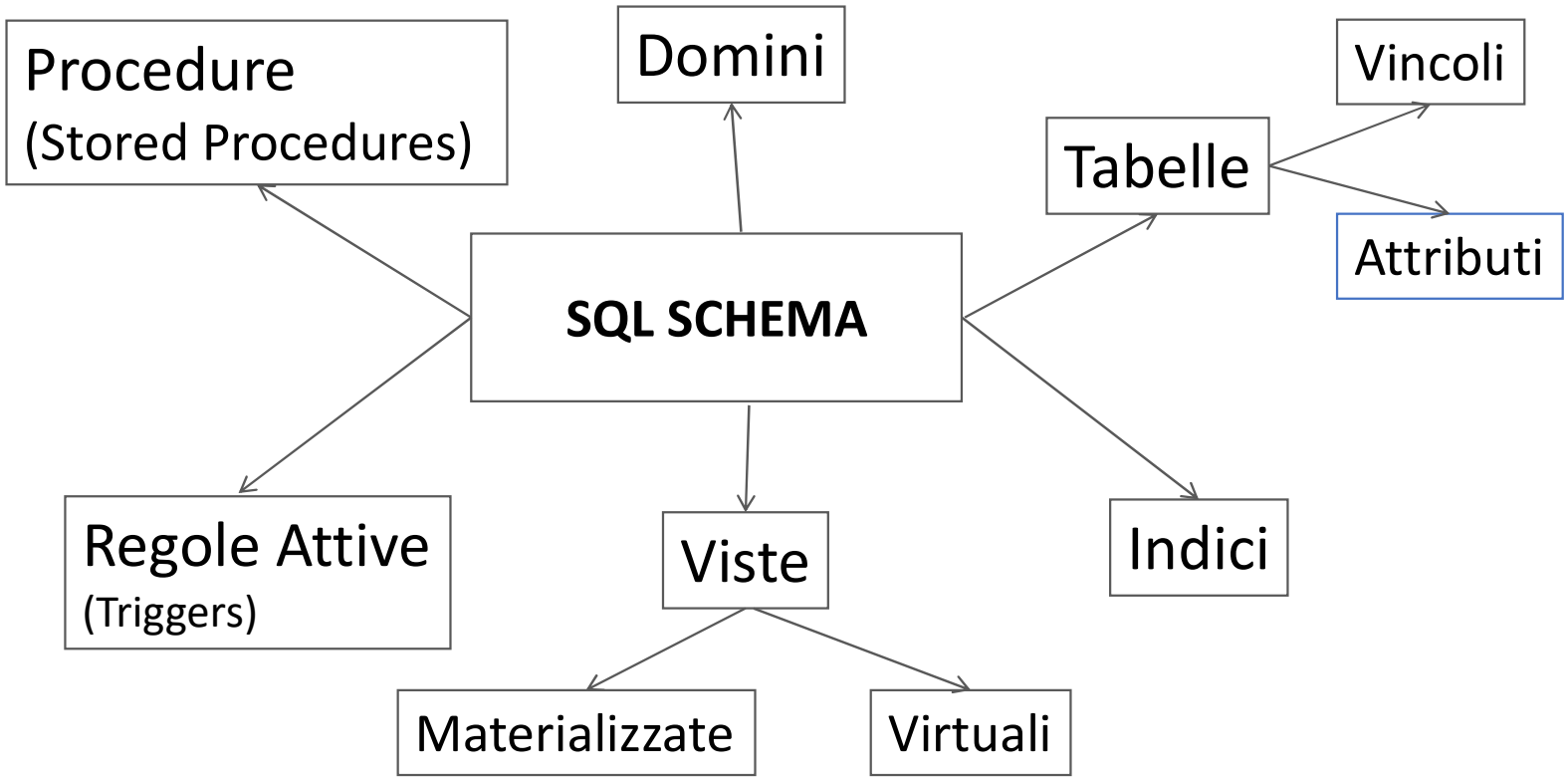
- **DDL** (*Data Definition Language*)

  Contiene i costrutti necessari per la creazione/modifica dello **schema** della base di dati

- **DML** (*Data Manipulation Language*)

  Contiene i costrutti per le interrogazioni e di inserimento/eliminazione/modifica di dati

SQL: Data Definition Language



UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Esempio di interrogazione (**query**) → Recuperare nome e cognome dello studente con numero di matricola pari a 4678

STUDENTI

Matricola	Nome	Cognome	DataNascita
4566	Marco	Rossi	3/5/1989
4678	Michele	Bianchi	2/5/1989
4900	Antonio	Rossi	14/3/1990

Nome	Cognome
Michele	Bianchi

SQL: Data Manipulation Language

Le operazioni di **interrogazione** vengono implementate dal costrutto di `select`

```
select    Attributo1, … AttributoM
from      Tabella1, … ,TabellaN
where     Condizione
```

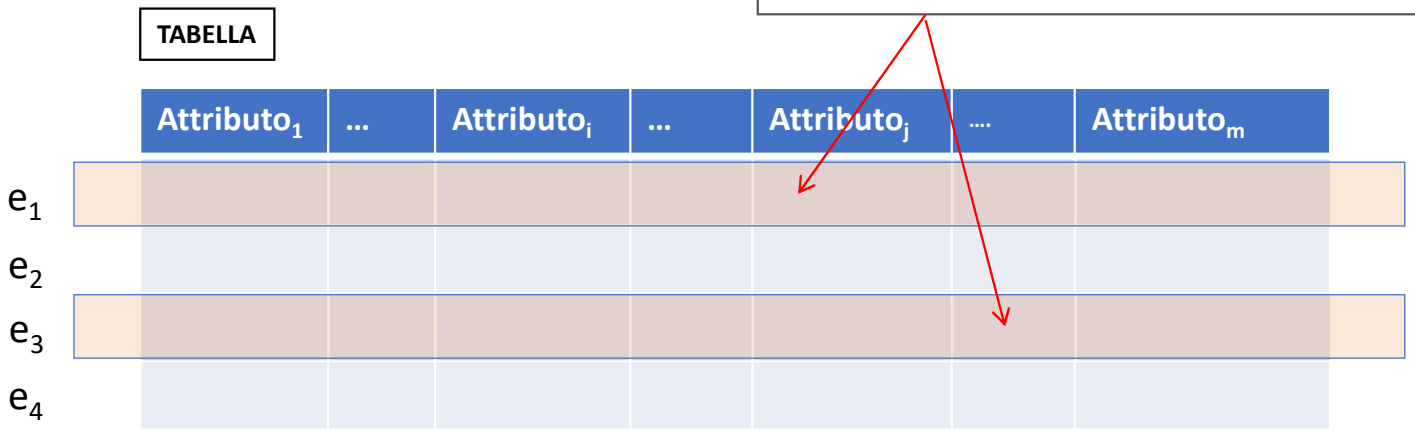
SEMANTICA: Effettua il **prodotto cartesiano** di $Tabella_1$, .., $Tabella_N$. Da queste, **estrai le righe** che rispettano la Condizione. Di quest'ultime, **preleva solo le colonne corrispondenti** a $Attributo_1$, …, $Attributo_M$

SQL: Data Manipulation Language

Nel caso di una sola tabella:

```
select    Attributo_i, Attributo_j, ... Attributo_m
from      Tabella
where     Condizione
```

STEP1: Si selezionano le ennuple (e) della tabella che soddisfano la condizione



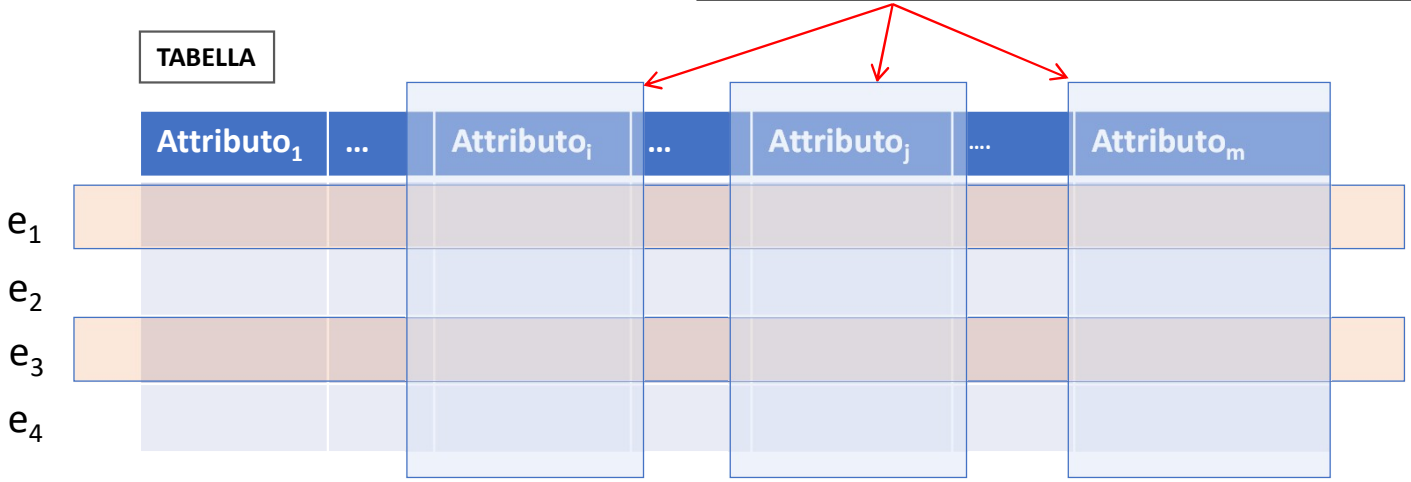
UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Nel caso di una sola tabella:

```
select    Attributo_i, Attributo_j, ... Attributo_m
from      Tabella
where     Condizione
```

STEP2: Si selezionano le colonne/attributi specificati dalla SELECT



UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Nel caso di una sola tabella:

```
select    Attributo_i, Attributo_j, ... Attributo_m
from      Tabella
where     Condizione
```

STEP3: Si costruisce la tabella risultato …

Numero di colonne definito dalla clausola **SELECT**

Numero di righe definito dalla clausola **WHERE**

$Attributo_1$	$Attributo_i$	$Attributo_m$

SQL: Data Manipulation Language

Esempio1. Selezionare i nomi degli impiegati che lavorano nell'ufficio A

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	15000
125	Michele	Monti	B	18000
134	Antonio	Verdi	A	25000
156	Giorgio	Rossi	A	32000

Nome
Marco
Antonio
Giorgio

```
SELECT NOME FROM IMPIEGATI WHERE UFFICIO='A'
```

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Esempio2. Selezionare i nomi degli impiegati che guadagnano più di 20000 euro annui

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	15000
125	Michele	Monti	B	18000
134	Antonio	Verdi	A	25000
156	Giorgio	Rossi	A	32000

Nome
Antonio
Giorgio

```
SELECT NOME FROM IMPIEGATI WHERE STIPENDIO>20000
```

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Esempio3. Selezionare nomi e cognomi degli impiegati che lavorano nell'ufficio B e guadagnano più di 20000 euro annui

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	15000
125	Michele	Monti	B	18000
134	Antonio	Verdi	A	25000
156	Giorgio	Rossi	A	32000

Nome
Antonio
Giorgio

```
SELECT NOME,COGNOME FROM IMPIEGATI WHERE
(STIPENDIO>20000 AND UFFICIO='B')
```

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

La clausola **`where`** specifica quali righe delle tabelle devono comparire nel risultato finale

La condizione della clausola può contenere un'**espressione** booleana, o una combinazione di espressioni mediante gli operatori `and`, `or`, `not`

```
SELECT CODICE
FROM IMPIEGATI
WHERE NOT(NOME='Marco' AND UFFICIO='A')
```

SQL: Data Manipulation Language

E' possibile **ridenominare** le colonne del risultato di una query attraverso il costrutto as

```
SELECT NOME as Name, Cognome as LastName
FROM IMPIEGATI
WHERE (NOME='Marco')
```

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	C	36000

Name	LastName
Marco	Marchi
Marco	Bianchi

SQL: Data Manipulation Language

E’ possibile usare *espressioni aritmetiche* (semplici) sui valori degli attributi di una **SELECT**

```
SELECT NOME as Name, Stipendio/12 as SalaryM
FROM IMPIEGATI
WHERE (NOME=‘Marco’)
```

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	C	36000

Name	SalaryM
Marco	1000
Marco	2000

SQL: Data Manipulation Language

La clausola **`from`** specifica la lista delle tabelle cui si deve accedere (nel caso #tabelle>1, si effettua il **prodotto cartesiano** delle stesse)

E' possibile specificare degli alias per i nomi delle tabelle, mediante il costrutto `as`:

```
SELECT CODICE
FROM IMPIEGATI AS I
WHERE (NOME='MARCO')
```

SQL: Data Manipulation Language

Vediamo come funziona la SELECT su più tabelle

Es. Selezionare il numero di telefono dell'impiegato con codice 145

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	A	36000
187	Giorgio	Rossi	B	12000

SEDI

UffNum	Telefono
A	2034333
B	2035434

SQL: Data Manipulation Language

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

COSA FA QUESTA QUERY?

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	A	36000
187	Giorgio	Rossi	B	12000

SEDI

UffNum	Telefono
A	2034333
B	2035434

SQL: Data Manipulation Language

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

STEP1. Si effettua il prodotto cartesiano delle due tabelle

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

SQL: Data Manipulation Language

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

STEP2. Si selezionano le righe con valori comuni nelle tue tabelle

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

STEP3. Si selezionano le righe relative all'impiegato 145

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

SQL: Data Manipulation Language

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

STEP4. Si seleziona la colonna dell'attributo Telefono

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

SQL: Data Manipulation Language

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

STEP5. Si costruisce il risultato finale

TEL
2035434

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Q. Che accade se le tabelle della clausola `from` hanno **attributi con nomi uguali**?

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFICIO) AND (CODICE=145)
```

???? ERRORE!!!

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	A	36000
187	Giorgio	Rossi	B	12000

SEDI

Ufficio	Telefono
A	2034333
B	2035434

SQL: Data Manipulation Language

In questi casi, si può utilizzare la notazione **NomeTabella.NomeAttributo** per far riferimento ad un attributo in maniera non ambigua.

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (IMPIEGATI.UFFICIO=SEDI.UFFICIO) AND (CODICE=145)
```

```
SELECT TELEFONO AS TEL
FROM IMPIEGATI AS I, SEDI AS S
WHERE (I.UFFICIO=S.UFFICIO) AND (CODICE=145)
```

SQL: Data Manipulation Language

ATTENZIONE: Il risultato di una query SQL **potrebbe avere righe duplicate!**

```
SELECT NOME AS NAME
FROM IMPIEGATI AS I
WHERE (STIPENDIO >20000)
```

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	C	36000

Name
Marco
Marco

SQL: Data Manipulation Language

Il costrutto `distinct` (nella `select`) consente di rimuovere i **duplicati** nel risultato

Il costrutto `all` (nella `select`) NON rimuove i duplicati (comportamento di default)

```
SELECT DISTINCT NOME AS NAME
FROM IMPIEGATI AS I
WHERE (STIPENDIO >20000)
```

Name
Marco

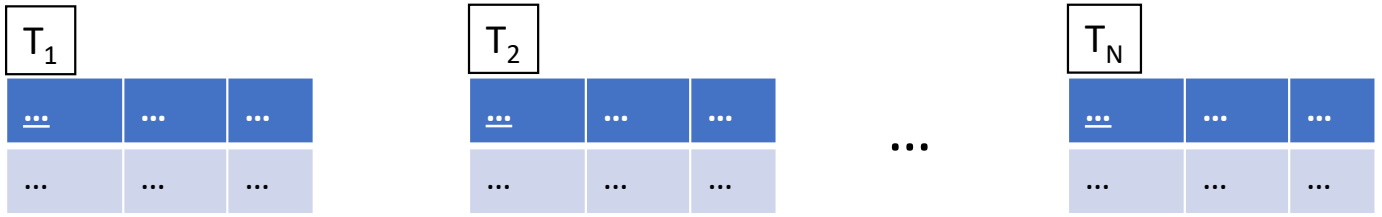
SQL: Data Manipulation Language

Sintassi Generale

SELECT **OP**(Attributo)
FROM ListaTabelle
WHERE Condizione

- sum
- max
- min
- avg
- count → count(*)

STEP 0: Si considerano le tabelle indicate nella clausola **FROM**

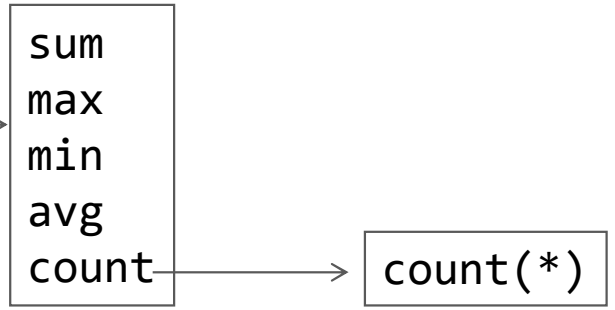


UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Sintassi Generale

SELECT **OP**(Attributo)
FROM ListaTabelle
WHERE Condizione



STEP 1: Si effettua il **prodotto cartesiano** delle tabelle

...	...	...								

UNIVERSITÀ DEGLI STUDI DI TERAMO

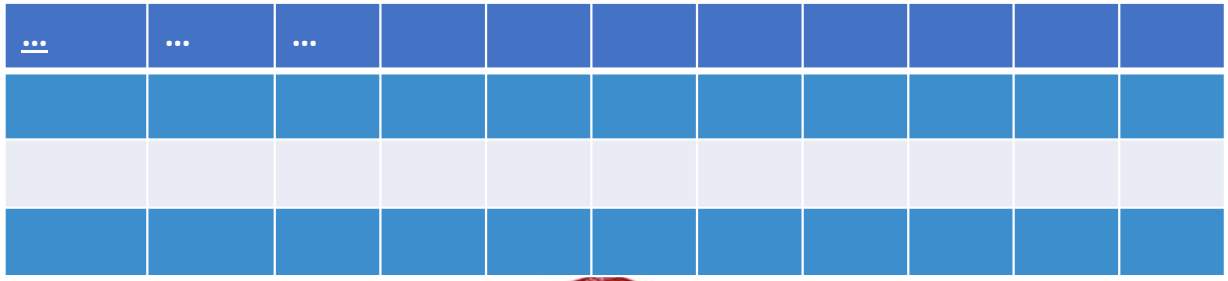
SQL: Data Manipulation Language

Sintassi Generale

SELECT **OP**(Attributo)
FROM ListaTabelle
WHERE Condizione

- `sum`
- `max`
- `min`
- `avg`
- `count` → `count(*)`

STEP 2: Si selezionano le righe che soddisfano la condizione **WHERE**

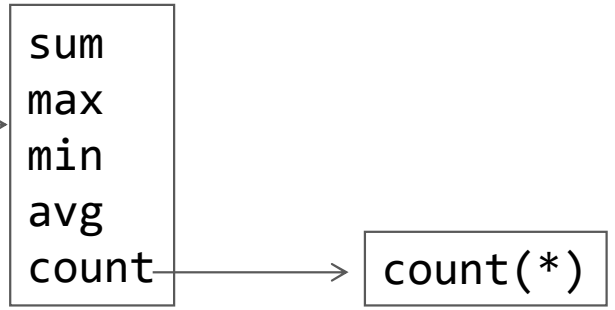


UNIVERSITÀ DEGLI STUDI DI TERAMO

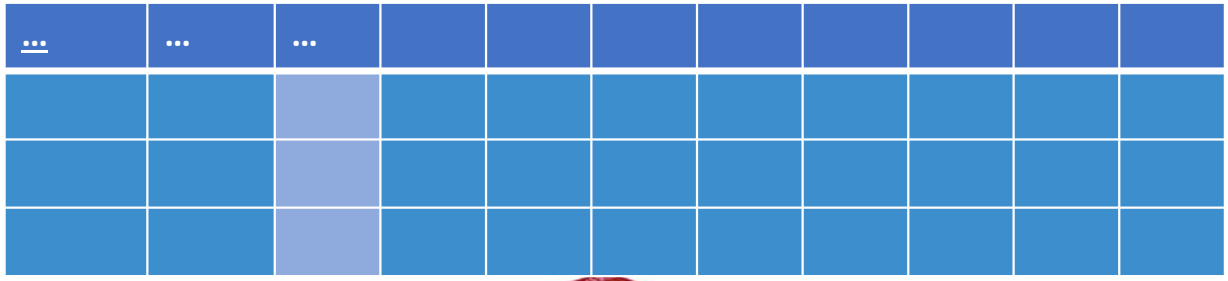
SQL: Data Manipulation Language

Sintassi Generale

SELECT OP(Attributo)
FROM ListaTabelle
WHERE Condizione



STEP 3: Si considera l'Attributo della **SELECT** e si applica **l'operatore aggregato** su tutti i valori della colonna



UNIVERSITÀ DEGLI STUDI DI TERAMO

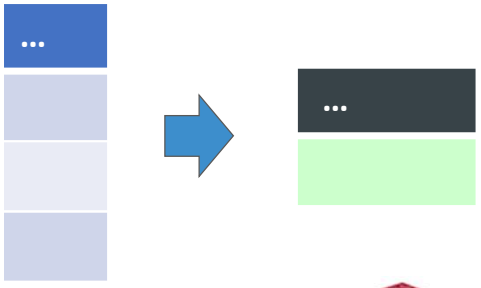
SQL: Data Manipulation Language

Sintassi Generale

SELECT OP(Attributo)
FROM ListaTabelle
WHERE Condizione

- `sum`
- `max`
- `min`
- `avg`
- `count` → `count(*)`

STEP 4: Dalla colonna si calcola **un solo valore** come risultato della query



Se non si usa l'operatore AS, la colonna risultato non ha un nome

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Es. Contare il numero di strutturati che lavorano nel *Dipartimento di Fisica*

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: Data Manipulation Language

```
SELECT COUNT(*) AS CONTATORE
FROM STRUTTURATI
WHERE (DIPARTIMENTO='FISICA')
```

Contatore
3

SQL: Data Manipulation Language

Es. Contare la somma complessiva degli stipendi degli strutturati *del dipartimento di Fisica*

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: Data Manipulation Language

```
SELECT SUM(STIPENDIO) AS TOTALE
FROM STRUTTURATI
WHERE (DIPARTIMENTO='FISICA')
```

Totale
70000

SQL: Data Manipulation Language

Es. Determinare il valore dello stipendio più alto tra i professori associati

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: Data Manipulation Language

```
SELECT MAX(STIPENDIO) AS MAXSTIPENDIO
FROM STRUTTURATI
WHERE (TIPO="ASSOCIATO")
```

MaxStipendio
50000

SQL: Data Manipulation Language

Es. Estrarre codice e stipendio del professore associato che ha lo stipendio più alto

ERRORE!

```
SELECT CODICE, MAX(STIPENDIO)
FROM STRUTTURATI
WHERE (TIPO='ASSOCIATO')
```

L'operatore aggregato **restituisce un solo valore**, mentre la prima parte della select **restituisce un valore per ogni tupla selezionata**!

COME FARE? Con **interrogazioni annidate**

SQL: Data Manipulation Language

```
SELECT MAX(STIPENDIO) AS MAXSTIPENDIO
FROM STRUTTURATI
WHERE (TIPO="ASSOCIATO")
```

MaxStipendio
50000

SQL: Data Manipulation Language

Es. Estrarre codice e stipendio del professore associato che ha lo stipendio più alto

ERRORE!

```
SELECT CODICE, MAX(STIPENDIO)
FROM STRUTTURATI
WHERE (TIPO='ASSOCIATO')
```

L'operatore aggregato **restituisce un solo valore**, mentre la prima parte della select **restituisce un valore per ogni tupla selezionata**!

COME FARE? Con **interrogazioni annidate**

SQL: Data Manipulation Language

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: Data Manipulation Language

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124		Micheli	Associato	Fisica	20000
125				Fisica	30000
126				Informatica	32000
127				Informatica	15000
129				Fisica	20000

RISULTATO

Dipartimento	Numero
Chimica	1
Fisica	3
Informatica	2

SQL: Data Manipulation Language

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio

Soluzione **1**

```
SELECT COUNT(*) AS NUMERO
FROM STRUTTURATI
```

Numero
6

NON FA QUANTO RICHIESTO!

SQL: Data Manipulation Language

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio

Soluzione **2**

```
SELECT COUNT(*) AS NUMERO,
DIPARTIMENTO
FROM STRUTTURATI
```

QUERY ERRATA!

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio

Soluzione **3**

```
SELECT COUNT(*) AS NUMERO
FROM STRUTTURATI
WHERE
Dipartimento='Fisica'
```

Numero
3

CASO PARTICOLARE

SQL: Data Manipulation Language

Operatori di query visti fin qui:

- `SELECT ATTRIBUTI FROM WHERE`

  Valuta i valori di ciascuna riga **in isolamento**

- `SELECT OP(ATTRIBUTI) FROM WHERE`

  Valuta i valori delle righe corrispondenti alle colonne della SELECT **in modo aggregato**

Q. Possibilità di **combinare i due approcci**?

SQL: Data Manipulation Language

Operatori di query visti fin qui:

- `SELECT ATTRIBUTI FROM WHERE`
  Valuta i valori di ciascuna riga **in isolamento**

- `SELECT OP(ATTRIBUTI) FROM WHERE`
  Valuta i valori delle righe corrispondenti alle colonne della SELECT **in modo aggregato**

Q. Possibilità di **combinare i due approcci**?

Estrarre informazioni aggregate da gruppi di righe

SQL: Data Manipulation Language

L'operatore di **raggruppamento** consente di dividere la tabella in **gruppi**, ognuno caratterizzata da un valore comune dell'attributo specificato nell'operatore

SELECT ListaAttributi1
FROM ListaTabelle
WHERE Condizione
GROUPBY ListaAttributi2

ListaAttributi1 deve essere un sottoinsieme di ListaAttributi2, può contenere operatori aggregati!

Ogni gruppo produce **una sola riga** nel risultato finale!

SQL: Data Manipulation Language

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO
FROM STRUTTURATI
GROUP BY DIPARTIMENTO
```

DIP	Numero
Chimica	1
Fisica	3
Informatica	2

SQL: Data Manipulation Language

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO
FROM STRUTTURATI
GROUP BY DIPARTIMENTO
```

STRUTTURATI

STEP1: Partizionamento della tabella

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: Data Manipulation Language

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO
FROM STRUTTURATI
GROUP BY DIPARTIMENTO
```

STEP1: Partizionamento della tabella

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	**Chimica**	20000
Codice	**Nome**	**Cognome**	**Tipo**	Dipartimento	Stipendio
124	Michele	Micheli	Associato	**Fisica**	20000
125	Lucia	Di Lucia	Ordinario	**Fisica**	30000
129	Michele	Bianchi	Associato	**Fisica**	20000
Codice	**Nome**	**Cognome**	**Tipo**	Dipartimento	Stipendio
126	Dario	Rossi	Ordinario	**Informatica**	32000
127	Mario	Rossi	Ricercatore	**Informatica**	15000

SQL: Data Manipulation Language

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO
FROM STRUTTURATI
GROUP BY DIPARTIMENTO
```

STEP2: Si applica la select su ciascun gruppo

Codice	Nome	C	artimento	Stipendio
123	Marco	M	**mica**	20000

Dipartimento	Numero
Chimica	1

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
124	Michele	M		**ca**	20000
125	Lucia	D		**ca**	30000
129	Michele	Bianchi	Associato	**Fisica**	20000

Dipartimento	Numero
Fisica	3

Codice	Nome	C	artimento	Stipendio
126	Dario	F	**ormatica**	32000
127	Mario	Rossi	**Informatica**	15000

Dipartimento	Numero
Informatica	2

UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO
FROM STRUTTURATI
GROUP BY DIPARTIMENTO
```

STEP3: Si costruisce il risultato finale

Dipartimento	Numero
Chimica	1
Fisica	3
Informatica	2

SQL: Data Manipulation Language

Es. Calcolare, per ogni dipartimento, lo stipendio medio degli strutturati

STRUTTURATI

Codice	**Nome**	**Cognome**	**Tipo**	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: Data Manipulation Language

```
SELECT DIPARTIMENTO AS DIP, AVG(STIPENDIO) AS STIPENDIOMEDIO
FROM STRUTTURATI
GROUP BY DIPARTIMENTO
```

STEP3: Si costruisce il risultato finale

Dipartimento	StipendioMedio
Chimica	20000
Fisica	23333
Informatica	23500

Attenzione! Nella SELECT possono comparire solo un sottoinsieme degli attributi della clausola GROUP BY oppure operatori aggregati

SQL: Data Manipulation Language

E’ possibile **filtrare** i gruppi in base a determinate condizioni, attraverso il costrutto `having`

```
SELECT ListaAttributi1
…
GROUPBY ListaAttributi2
HAVING Condizione
```

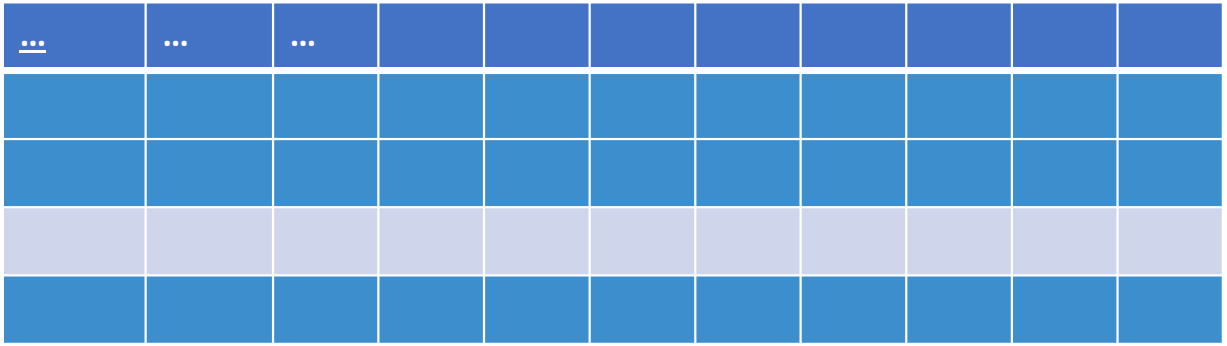
- clausola `where` → valutata riga per riga
- clausola `having` → valutata su ciascun gruppo, contiene operatori aggregati o condizioni su `ListaAttributi2`

SQL: Data Manipulation Language

Sintassi Generale

```
SELECT ListaAttributi1
FROM ListaTabelle
WHERE Condizione
GROUP BY ListaAttributi2
HAVING Condizione
```

STEP0: Prodotto cartesiano delle tabelle + Estrazione delle righe che rispettano la condizione della clausola `WHERE`



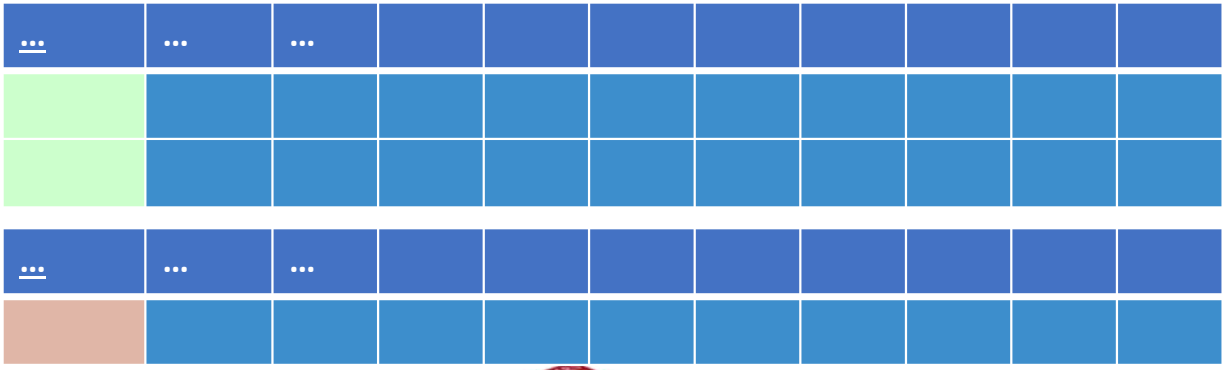
UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Sintassi Generale

```
SELECT ListaAttributi1
FROM ListaTabelle
WHERE Condizione
GROUP BY ListaAttributi2
HAVING Condizione
```

STEP1: Partizionamento della tabella



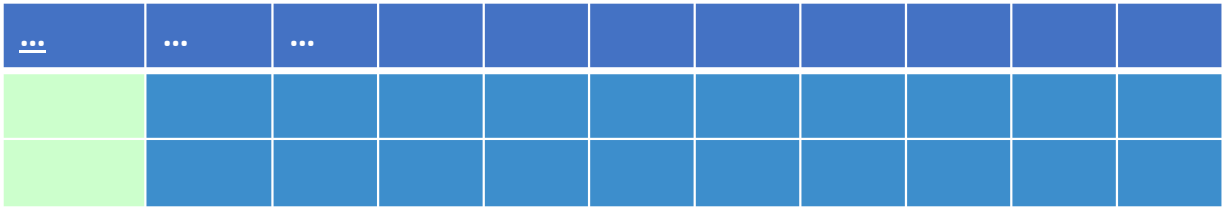
UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Sintassi Generale

```
SELECT ListaAttributi1
FROM ListaTabelle
WHERE Condizione
GROUP BY ListaAttributi2
HAVING Condizione
```

STEP2: Selezione dei gruppi



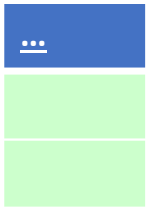
UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Sintassi Generale

```
SELECT ListaAttributi1
FROM ListaTabelle
WHERE Condizione
GROUP BY ListaAttributi2
HAVING Condizione
```

STEP2: Selezione dei valori delle colonne o esecuzione degli operatori aggregati su ciascuno dei gruppi e composizione della tabella finale



UNIVERSITÀ DEGLI STUDI DI TERAMO

SQL: Data Manipulation Language

Es. Estrarre il nome dei dipartimenti che hanno **almeno due strutturati** nel suo organico

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: Data Manipulation Language

```
SELECT DIPARTIMENTO AS DIP
FROM STRUTTURATI
GROUP BY DIPARTIMENTO
HAVING COUNT(*) > 2
```

DIP
Fisica
Informatica

SQL: Data Manipulation Language

Costrutto `select` nella sua forma più generale

```
SELECT ListaAttributi
FROM ListaTabelle
WHERE Condizione
LIMIT Number
GROUPBY  AttributiRaggruppamento
HAVING CondizioniGruppi
ORDERBY ListaAttributiOrdinamento
```

SQL: Data Manipulation Language

In SQL, è possibile effettuare **operazioni insiemistiche** tra tabelle o in generale tra risultati di SELECT

- `UNION [ALL]`
- `INTERSECT [ALL]`
- `EXCEPT [ALL]`

Gli attributi della SELECT devono avere tipi di dato compatibili e (possibilmente) gli stessi nomi.

SQL: Data Manipulation Language

Es. Estrarre nome e cognome di tutto il personale universitario (strutturati + tecnici)

STRUTTURATI

Codice	Nome	Cognome	Tipo
123	Marco	Marchi	Associato
124	Michele	Micheli	Associato
125	Lucia	Di Lucia	Ordinario
126	Dario	Rossi	Ordinario
127	Mario	Rossi	Ricercatore
129	Michele	Bianchi	Associato

TECNICI

Codice	Nome	Cognome	Livello
445	Michele	Marini	5
356	Daniele	Marini	6
154	Giovanna	Bianchi	5
156	Lucia	Di Lucia	4

SQL: Data Manipulation Language

Es. Estrarre nome e cognome di tutto il personale universitario (strutturati + tecnici)

```
SELECT NOME, COGNOME
FROM STRUTTURATI
UNION
SELECT NOME, COGNOME
FROM TECNICI
```

Nome	Cognome
Marco	Marchi
Michele	Micheli
Lucia	Di Lucia
Dario	Rossi
Mario	Rossi
Michele	Bianchi
Michele	Marini
…	…

SQL: Data Manipulation Language

Es. Estrarre nome e cognome degli strutturati che hanno degli omonimi che lavorano come tecnici

```
SELECT NOME, COGNOME
FROM STRUTTURATI
INTERSECT
SELECT NOME, COGNOME
FROM TECNICI
```

Nome	Cognome
Lucia	Di Lucia

SQL: Data Manipulation Language

Es. Estrarre nome e cognome degli strutturati che NON hanno degli omonimi che lavorano come tecnici

```
SELECT NOME, COGNOME
FROM STRUTTURATI
EXCEPT
SELECT NOME, COGNOME
FROM TECNICI
```

Nome	Cognome
Marco	Marchi
Michele	Micheli
Dario	Rossi
Mario	Rossi
Michele	Bianchi
Michele	Marini

SQL: Data Manipulation Language

Attenzione. Gli attributi delle SELECT nelle due tabelle devono avere **tipi compatibili**

```
SELECT RUOLO
FROM STRUTTURATI
UNION
SELECT LIVELLO
FROM TECNICI
```

ERRORE!
STRUTTURATO.Ruolo e' una **stringa**
TECNICI.Livello e' un **intero**.

SQL: Data Manipulation Language

Oltre ad i comandi di interrogazione, la parte DML definisce anche le operazioni per la **modifca dell'istanza** della base di dati

- `insert` → inserisce una o più righe
- `delete` → cancella una o più righe
- `update` → aggiorna un attributo o più

SQL: Data Manipulation Language

E’ possibile **inserire una riga** esplicitando i valori degli attributi oppure estraendo le righe da altre tabelle del database

- `insert into NomeTabella [ListaAttributi] values (ListaValori)`

```
INSERT INTO IMPIEGATI(Codice, Nome, Cognome,
Ufficio) values (‘8’,‘Vittorio’,’Rossi’,’A’)
```

SQL: Data Manipulation Language

E' possibile **inserire una riga** esplicitando i valori degli attributi oppure estraendo le righe da altre tabelle del database

- `insert into NomeTabella [ListaAttributi] values (ListaValori)`

```
INSERT INTO IMPIEGATI(Codice, Nome, Cognome,
Ufficio) values ('8','Vittorio','Rossi')
```

`Ufficio` non specificato: NULL o default

SQL: Data Manipulation Language

E’ possibile **inserire una riga** esplicitando i valori degli attributi oppure estraendo le righe da altre tabelle del database

- `insert into NomeTabella SQLSelect`

```
INSERT INTO IMPIEGATI
(Codice,Nome,Cognome,Ufficio) VALUES (
     SELECT * FROM IMPIEGATICOMUNE
)
```

SQL: Data Manipulation Language

E’ possibile **cancellare** tutte le righe che soddisfano una condizione (cancella tutto se non specificata)

- `delete from Tabella where Condizione`

```
DELETE FROM IMPIEGATI
DELETE FROM IMPIEGATI WHERE (UFFICIO=“A”)
DELETE FROM TABELLA WHERE NOME IN (
     SELECT NOME FROM IMPIEGATICOMUNE)
```

SQL: Data Manipulation Language

E’ possibile **aggiornare** il contenuto di uno o più attributi di una tabella che rispettano una certa condizione

- `update NomeTabella`
  `set attributo=expr|SELECT|null|default`
  `[where Condizione]`

```
UPDATE IMPIEGATI SET NOME=“Mario” WHERE
(CODICE=5)
```

SQL: Data Manipulation Language

E’ possibile **aggiornare** il contenuto di uno o più attributi di una tabella che rispettano una certa condizione

- `UPDATE IMPIEGATI SET NOME=‘MARCO’ WHERE (CODICE=5)`

- `UPDATE IMPIEGATI SET NOME=(SELECT NOME FROM IMPIEGATICOMUNE WHERE CODICE=5) WHERE (CODICE=5)`

SQL: Data Manipulation Language

E' possibile implementare il **join** tra tabelle in due modi distinti (ma equivalenti nel risultato)

- Inserendo le condizioni del `JOIN` direttamente nella clausola del `WHERE`
- Attraverso l'utilizzo dell'operatore di `inner JOIN` nella clausola `FROM`

```
SELECT ListaAttributi
FROM Tabella JOIN Tabella ON CondizioneJoin
[WHERE Condizione]
…
```

SQL: Data Manipulation Language

Esempio di **query con utilizzo dell'inner join**

GUIDATORI

NrPatente	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

Targa	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

```
SELECT Modello FROM GUIDATORI, VEICOLI
WHERE GUIDATORI.NrPatente=VEICOLI.NrPatente AND Nome='Sara'
```

EQUIVALE A

```
SELECT Modello FROM GUIDATORI JOIN VEICOLI ON
GUIDATORI.NrPatente=VEICOLI.NrPatente WHERE Nome='Sara'
```

SQL: Data Manipulation Language

Esistono altre **tre** varianti dell'operatore `JOIN`

- **left join** → risultato dell'inner join + righe della tabella di sinistra che non hanno un corrispettivo a destra (completate con valori NULL)

```
SELECT ListaAttributi
FROM Tabella LEFT JOIN Tabella ON CondizioneJoin
[WHERE Condizione]
…
```

SQL: Data Manipulation Language

Esempio di **query con utilizzo del left join**

GUIDATORI

NrPatente	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

Targa	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

```
SELECT Modello
FROM GUIDATORI LEFT JOIN VEICOLI ON GUIDATORI.NrPatente
=VEICOLI.NrPatente
```

SQL: Data Manipulation Language

Esempio di **query con utilizzo del left join**

GUIDATORI

NrPatente	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

Targa	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

NrPatente	Nome	Cognome	Targa	Modello	NrPatente
1243242	Sara	Bianchi	BO2121	Panda	1243242
1243242	Sara	Bianchi	BO4567	Panda	1243242
2656565	Michele	Rossi	NULL	NULL	NULL

SQL: Data Manipulation Language

Esistono altre **tre** varianti dell'operatore `JOIN`

- **right join** → risultato dell'inner join + righe della tabella di destra che non hanno un corrispettivo a destra (completate con valori NULL)

```
SELECT ListaAttributi
FROM Tabella RIGHT JOIN Tabella ON CondizioneJoin
[WHERE Condizione]
…
```

SQL: Data Manipulation Language

Esempio di **query con utilizzo del right join**

GUIDATORI

NrPatente	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

Targa	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

NrPatente	Nome	Cognome	Targa	Modello	NrPatente
1243242	Sara	Bianchi	BO2121	Panda	1243242
1243242	Sara	Bianchi	BO4567	Panda	1243242
NULL	NULL	NULL	BO4896	Yaris	5687876

SQL: Data Manipulation Language

Esistono altre **tre** varianti dell'operatore `JOIN`

- **full join** → risultato dell'inner join + righe della tabella di sinistra/destra che non hanno un corrispettivo a destra/sinistra (completate con valori NULL)

```
SELECT ListaAttributi
FROM Tabella FULL JOIN Tabella ON CondizioneJoin
[WHERE Condizione]
…
```

SQL: Data Manipulation Language

Esempio di **query con utilizzo del full join**

GUIDATORI

NrPatente	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

Targa	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

NrPatente	Nome	Cognome	Targa	Modello	NrPatente
1243242	Sara	Bianchi	BO2121	Panda	1243242
1243242	Sara	Bianchi	BO4567	Panda	1243242
NULL	NULL	NULL	BO4896	Yaris	5687876
2656565	Michele	Rossi	NULL	NULL	NULL

Il Linguaggio SQL

Transazioni

Gestione delle Transazioni

Esempio. Gestione ordini su un sito di ecommerce

ITEM(Codice,Descrizione,Prezzo,Quantita)
ORDINE(Id, Data, Ordinante, ItemOrdinato)

(struttura DB semplificata)

```
SET Stock=(SELECT Quantità FROM ITEM WHERE Codice=CodiceScelto);
IF (Stock>0) THEN
        UPDATE ITEM SET Quantita=Quantita-1 WHERE
        (Codice=CodiceScelto));
        INSERT INTO ORDINE(Data,Ordinante,ItemOrdinato) VALUES
        (NOW(), NomeOrdinante, CodiceScelto);
END IF;
```

UNIVERSITÀ DEGLI STUDI DI TERAMO

Gestione delle Transazioni

Esempio. Gestione ordini su un sito di ecommerce

ITEM(Codice,Descrizione,Prezzo,Quantita)
ORDINE(Id, Data, Ordinante, ItemOrdinato)

(struttura DB semplificata)

```
SET Stock=(SELECT Quantità FROM ITEM WHERE Codice=CodiceScelto);
IF (Stock>0) THEN
        UPDATE ITEM SET Quantita=Quantita-1 WHERE
        (Codice=CodiceScelto));
        INSERT                                    mOrdinato) VALUES
        (NULL, NomeOrdinante, CodiceScelto);
END IF;
```

Il sistema va in crash in questo punto!

UNIVERSITÀ DEGLI STUDI DI TERAMO

Gestione delle Transazioni

Esempio. Gestione ordini su un sito di ecommerce

ITEM(Codice,Descrizione,Prezzo,Quantita)
ORDINE(Id, Data, Ordinante, ItemOrdinato)

(struttura DB semplificata)

```
SET Stock=(SELECT Quantità FROM ITEM WHERE Codice=CodiceScelto);
IF (Stock>0) THEN
        UPDATE ITEM SET Quantita=Quantita-1 WHERE
        (Codice=CodiceScelto));
        INSERT INTO ORDINE(Data,Ordinante,ItemOrdi
        (NOW(), NomeOrdinante, CodiceScelto);
END IF;
```

Due ordini in contemporanea eseguono la query!

UNIVERSITÀ DEGLI STUDI DI TERAMO

Gestione delle Transazioni

Le **transazioni** rappresentano **unità di lavoro** elementare o atomica (insiemi di istruzioni SQL) che modifica il contenuto di una base di dati

```
start transaction
update SalariImpiegati
set conto=conto*1.2
where (CodiceImpiegato = 123)
commit work
```

Le transazioni sono comprese tra una **Start Transaction** ed una **Commit/Rollback**

Gestione delle Transazioni

Le **transazioni** rappresentano **unità di lavoro** elementare o atomica (insiemi di istruzioni SQL) che modifica il contenuto di una base di dati

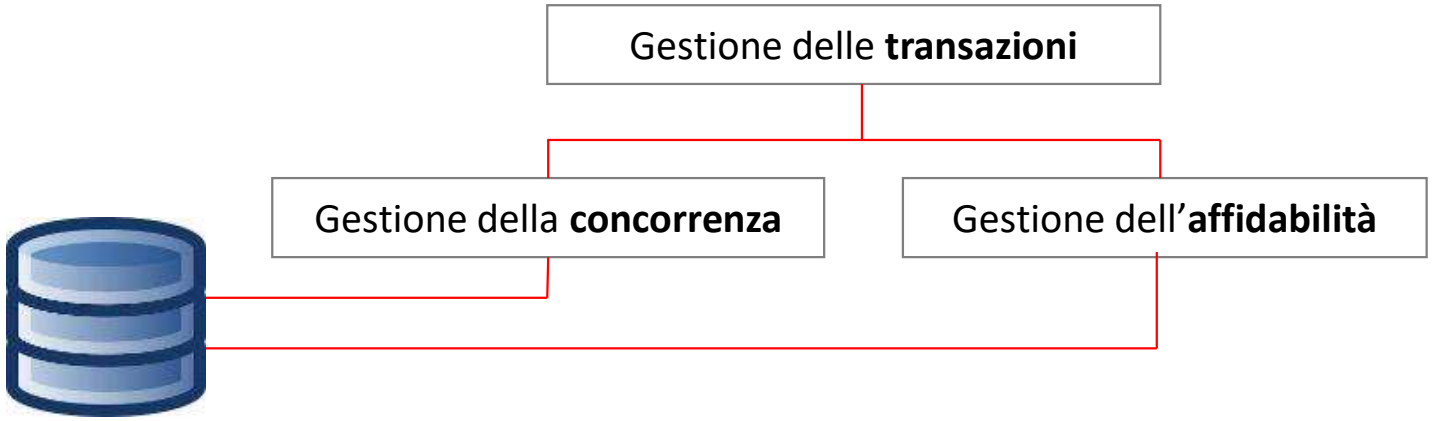
```
start transaction
update SalariImpiegati
set conto=conto-10
where (CodiceImpiegato = 123)
if conto >0 commit work;
else rollback work
```

Le transazioni sono comprese tra una **Start Transaction** ed una **Commit/Rollback**

Gestione delle Transazioni

UNIVERSITÀ DEGLI STUDI DI TERAMO

Gestione delle Transazioni



Gestore della concorrenza
Garantisce l'isolamento in caso di esecuzione concorrente di più transazioni

Gestore dell'affidabilità
Garantisce atomicità e persistenza (usando *log* e *checkpoint)*

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione ai Database NoSQL

Introduzione ai Database NoSQL

NoSQL→ Movimento che promuove l'adozione di DMBS non basati sul modello relazionale

- Il termine NOSQL appare per la prima volta in una pubblicazione di Carlo Strozzi nel 1998
- Oggi, il termine NOSQL viene usato per lo più nell'accezione **NoT Only SQL**

"Next generation databases mostly addressing some of the points: being non-relational, distributed, open source and horizzontally scalable" (definizione da http://nosql-database.org/)

Introduzione ai Database NoSQL

PROPRIETA dei SISTEMI NO-SQL

- Database **distribuiti**
- Strumenti generalmente **open-source**
- NON dispongono di **schema**
- NON supportano operazioni di **join**
- NON implementano le proprietà **ACID** delle transazioni
- Sono **scalabili** orizzontalmente
- Sono in grado di gestire **grandi moli di dati**
- Supportano le **repliche** dei dati

Introduzione ai Database NoSQL

Motivazioni legate alla diffusione dei database **NoSQL**

- Gestione dei **Big-data**
- Limitazioni del **modello relazionale**
- Teorema **CAP**

Big Data

Big data: moli di dati, eterogenee, destrutturate, difficili da gestire attraverso tecnologie tradizionali

- Tecnologie tradizionali → RDBMS
- Il termine big-data è oggi usato sia per denotare tipologie di dati, sia le **tecnologie** e i **tool** di gestione degli stessi

Big Data

Volume: Big data = grosse moli di data

Esempio: Dati da esperimenti scientifici

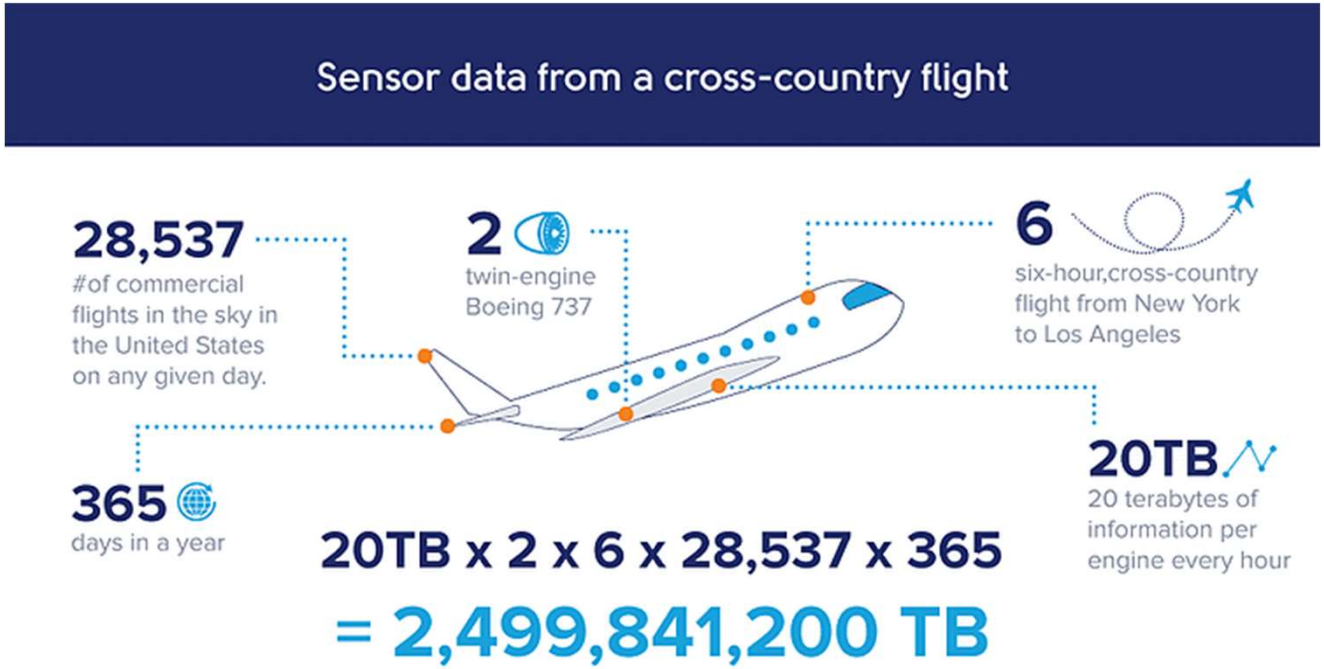
Acceleratore di particelle del CERN

- 600 milioni di collisioni al secondo
- 30 Petabyte di dati annui relativi ai *collision event*

http://home.web.cern.ch/about/computing

UNIVERSITÀ DEGLI STUDI DI TERAMO

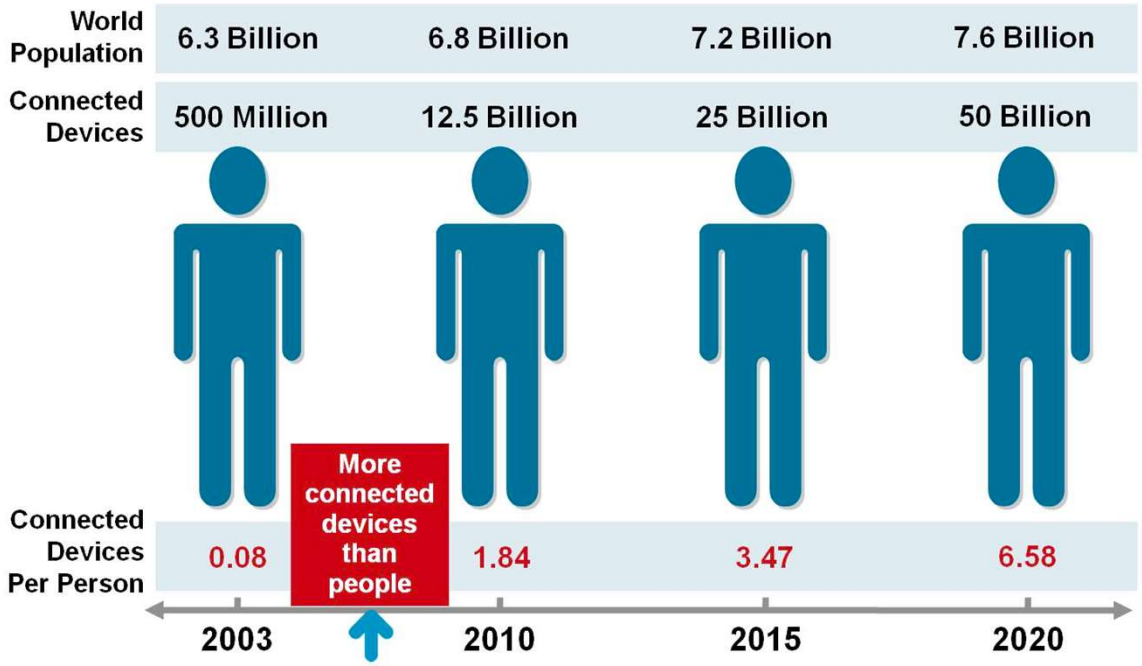
Big Data



UNIVERSITÀ DEGLI STUDI DI TERAMO

Big Data

From an Internet of Humans to **IoT** (**Internet of Things)**



Big Data

Wearables

e-Health

Automazione Industriale

Smart Cities

Domotica

Automotive

UNIVERSITÀ DEGLI STUDI DI TERAMO

Big Data

UNIVERSITÀ DEGLI STUDI DI TERAMO

Big Data

- Le applicazioni legate alla sensoristica possono generare **moli ingenti di dati**, a frequenza costante o variabile
- Tuttavia, più che il contenuto dei dati singoli, spesso l'interesse delle applicazioni è rivolto alla **conoscenza "estraibile" dai dati**

UNIVERSITÀ DEGLI STUDI DI TERAMO

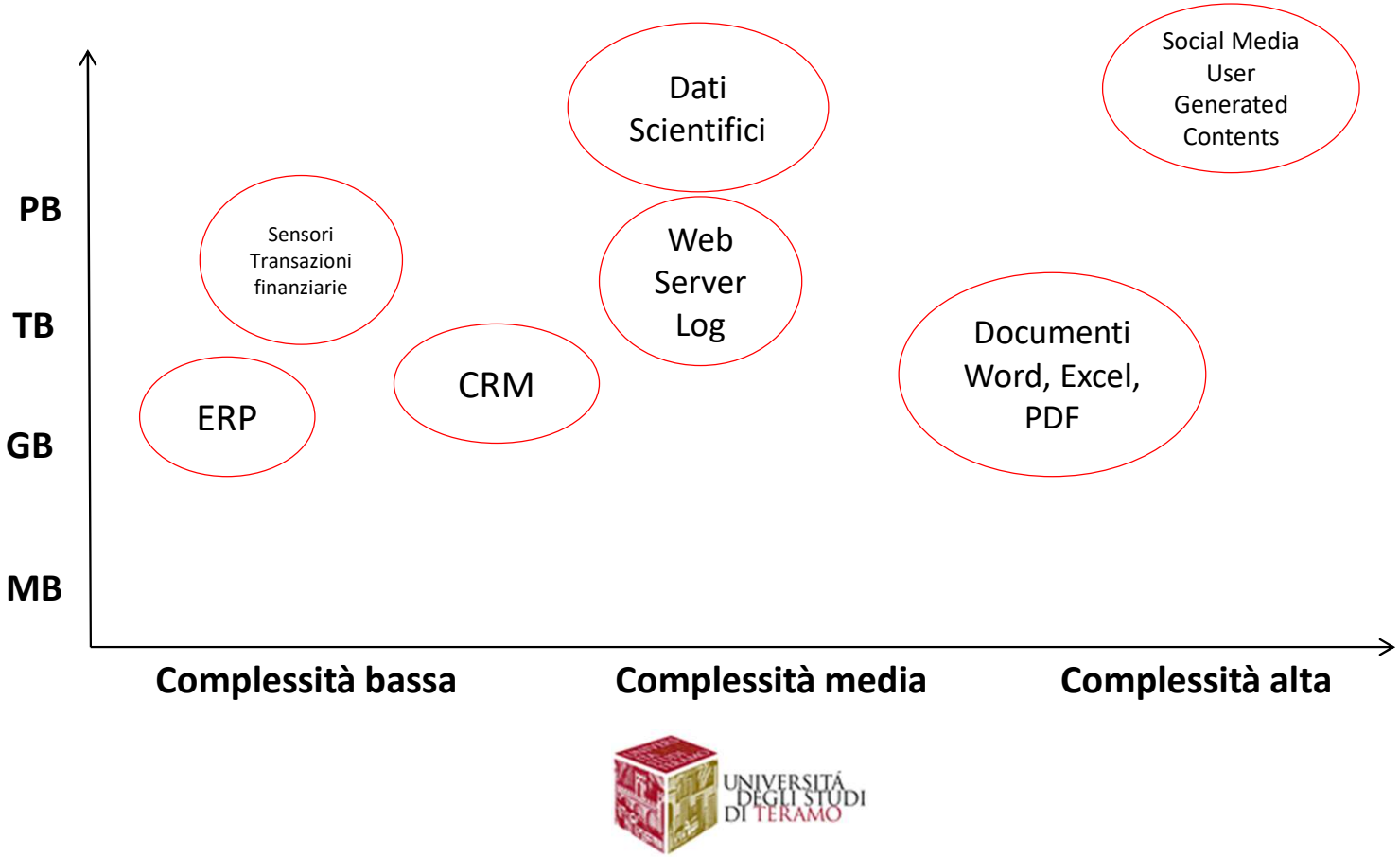
Big Data

Data Mining: tecniche di **apprendimento computerizzato** per analizzare ed estrarre **conoscenze** da collezioni di dati

Pattern e **relazioni** non note a priori e non immediatamente identificabili

Disciplina complessa: utilizzo di tecniche di **machine learning**, **intelligenza artificiale** e **statistica**

Big Data



UNIVERSITÀ DEGLI STUDI DI TERAMO

Big Data

Velocity: Big data = *stream* di dati

Esempio: Sistemi health-care

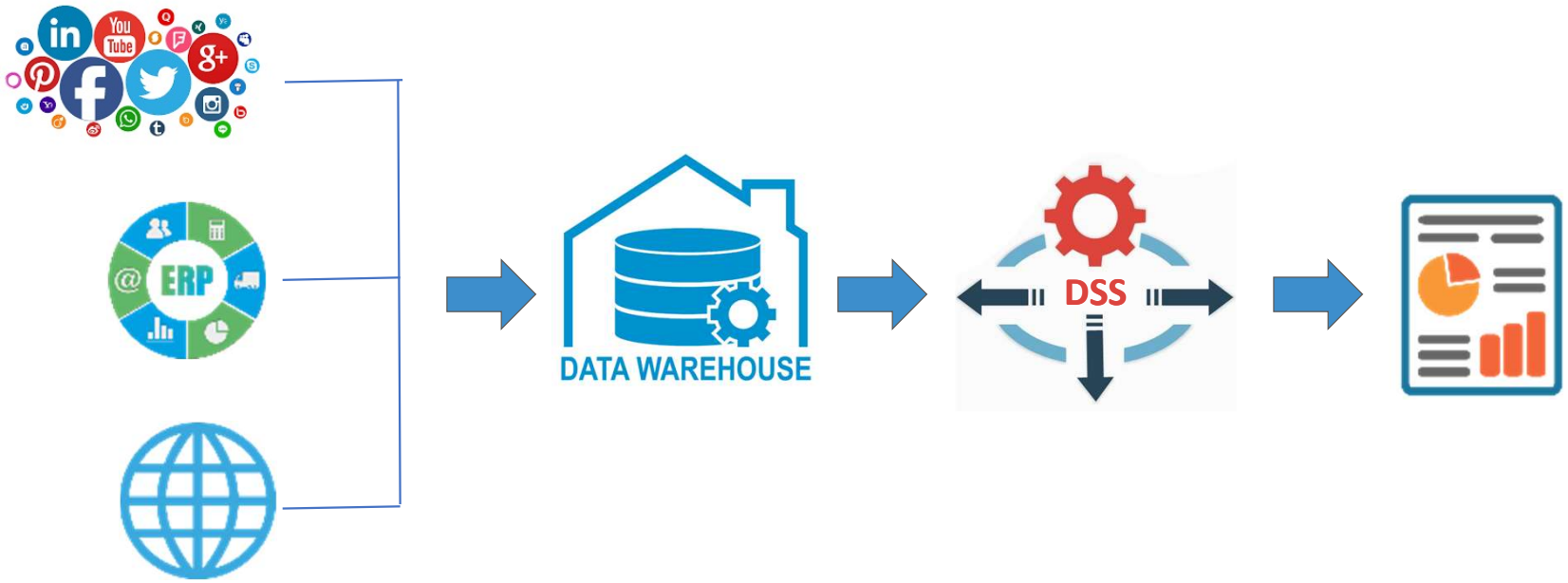
Early-Warning System

- Predict rising brain pressure in patients with traumatic brain injuries
- Stream di 1000 dati al secondo

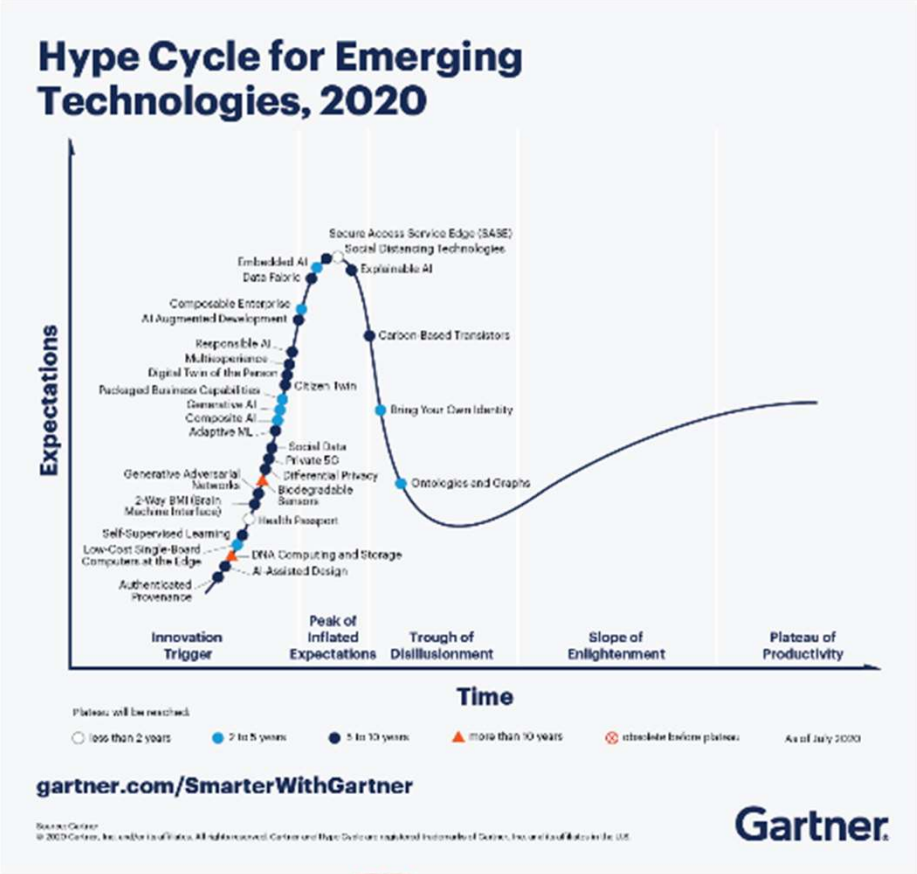
UNIVERSITÀ DEGLI STUDI DI TERAMO

Big Data

Variety: Big data = *dati eterogenei, multi-sorgente*



Big Data



UNIVERSITÀ DEGLI STUDI DI TERAMO

Big Data

ESEMPI DI **TECNOLOGIE** BIG DATA

- **Acquisizione**
  API Social Media, Web Scraping, Apache Flume, Microsoft StreamInsight
- **Organizzazione/Storage**
  Hadoop, DMBS NoSQL
- **Integrazione**
  Hive, Sqoop
- **Analisi**
  Pig, R, Mahout

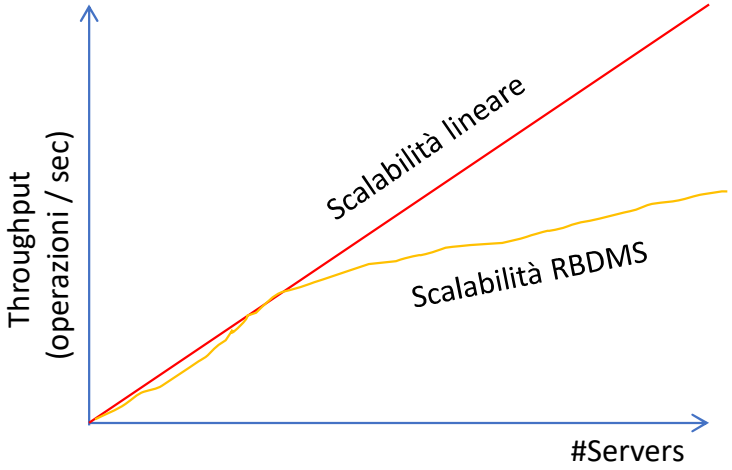
Introduzione ai Database NoSQL

Motivazioni legate alla diffusione dei database **NoSQL**

- Gestione dei **Big-data**
- Limitazioni del **modello relazionale**
- Teorema **CAP**

Limitazioni del Modello Relazionale

1. Il modello relazionale presuppone una rappresentazione tabellare, che accade se i dati non si presentano in tale forma? (es. pagina web)
2. Alcune operazioni non possono essere implementate in SQL (es. Memorizzazione di un grafo, e calcolo del percorso minimo tra due punti)
3. Scalabilità orizzontale dei DMBS relazionali



PROBLEMI

- Gestione dei vincoli
- Repliche dei dati
- Gestione delle transazioni
- Soddisfacimento delle proprietà ACID

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione ai Database NoSQL

Motivazioni legate alla diffusione dei database **NoSQL**

- Gestione dei **Big-data**
- Limitazioni del **modello relazionale**
- Teorema **CAP**

Teorema CAP

Il teorema di Brewer (**CAP Theorem**) afferma che un sistema distribuito può soddisfare al massimo solo due delle tre proprietà elencate sotto:

- **Consistency** → Tutti i nodi della rete vedono gli stessi dati
- **Availability** → Il servizio è sempre disponibile
- **Partion Tolerance** → Il servizio continua a funzionare correttamente anche in presenza di perdita di messaggi o di partizionamenti della rete

Teorema CAP

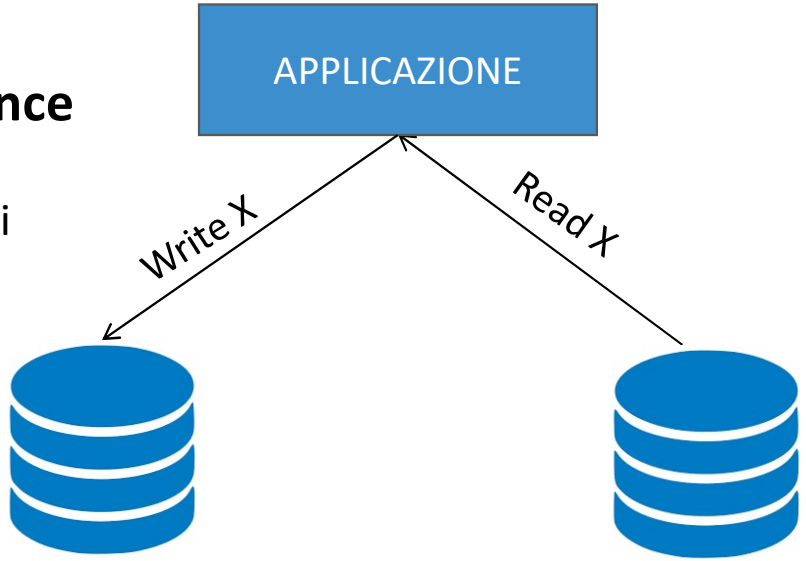
Nel caso di un DB distribuito (gestito da un cluster), è possibile soddisfare al massimo solo due delle tre proprietà elencate sotto:

- **Consistency** → Se l'utente A modifica il dato X sul server 1, e B legge X dal server 2, B legge l'ultima versione disponibile di X
- **Availability** → Se un utente effettua una query sul server A o B, la query restituisce un risultato
- **Partion Tolerance** → Il servizio continua a funzionare correttamente anche in presenza di perdita di messaggi o di partizionamenti della rete

Teorema CAP

CASO 1: Consistency + Availability

NO Partition Tolerance
Il sistema non funziona correttamente in caso di perdita di messaggi

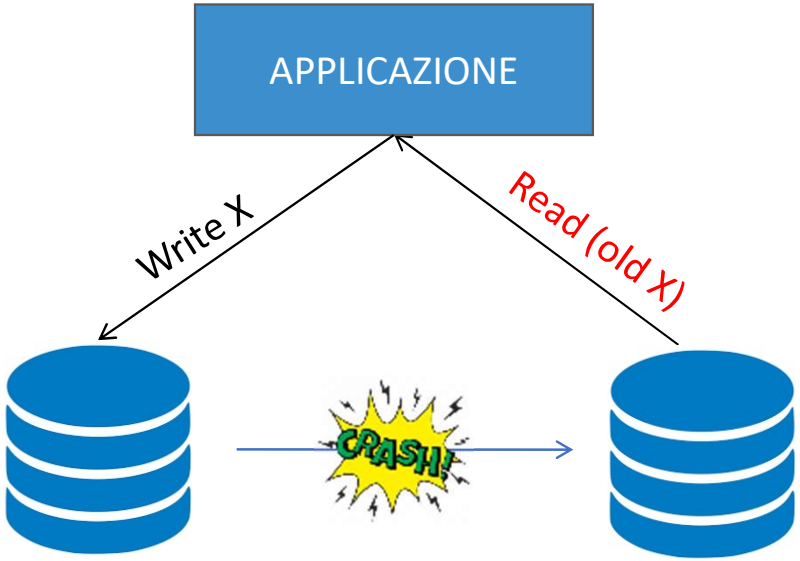


UNIVERSITÀ DEGLI STUDI DI TERAMO

Teorema CAP

CASO 2: Availability + Partition Tolerance

NO Consistency
Repliche del dato
non aggiornate!

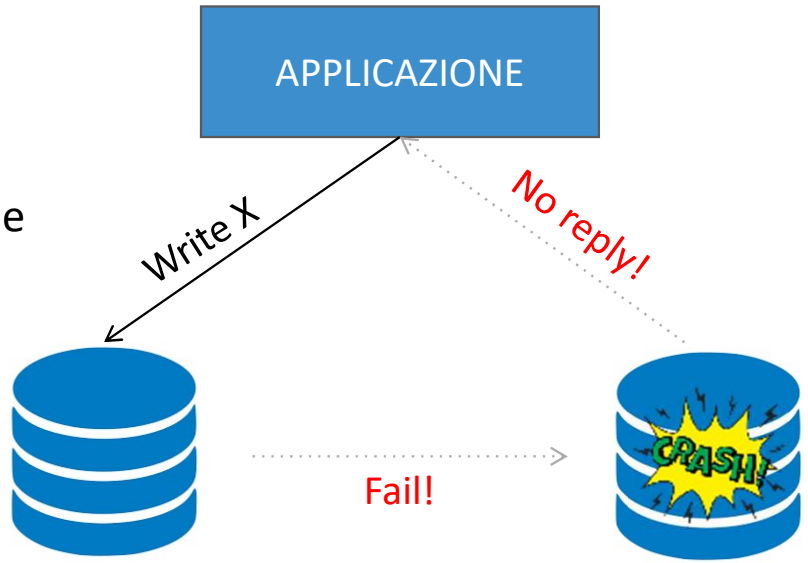


UNIVERSITÀ DEGLI STUDI DI TERAMO

Teorema CAP

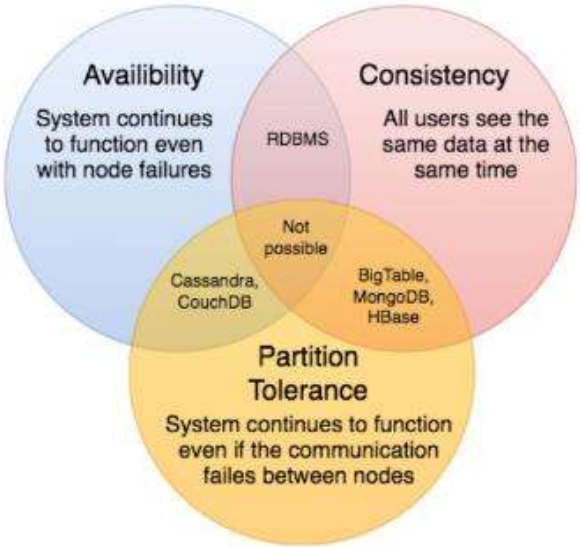
CASO 3: Consistency+ Partition Tolerance

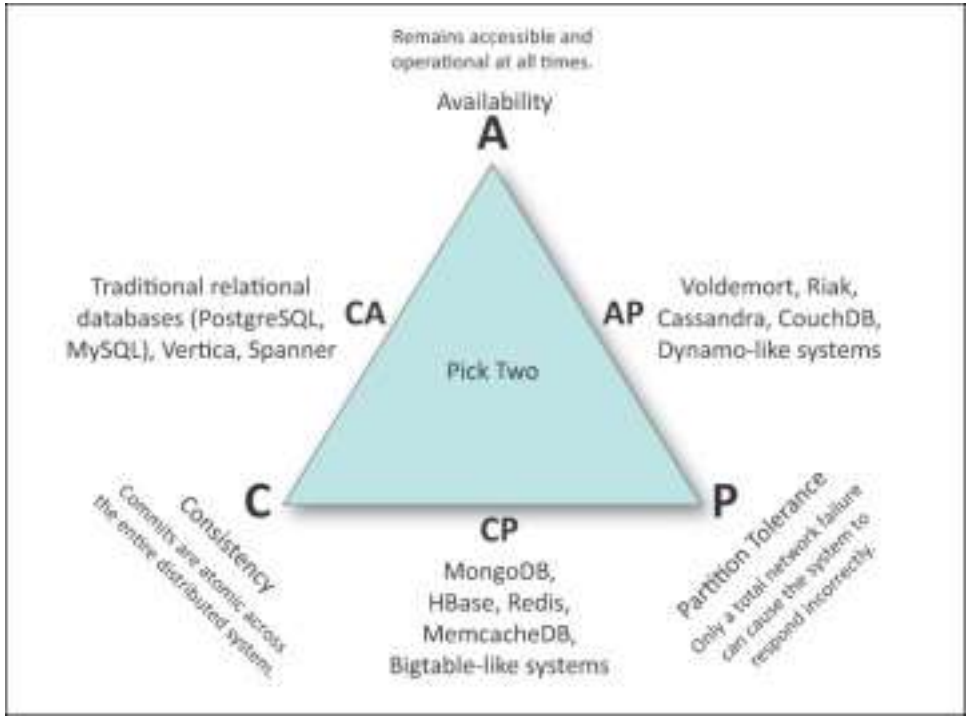
NO Availability
La query non produce risposta



UNIVERSITÀ DEGLI STUDI DI TERAMO

Teorema CAP





UNIVERSITÁ DEGLI STUDI DI TERAMO

Introduzione ai Database NoSQL

PROPRIETA' di **BASE**

Basically Available → I nodi del sistema distribuito possono essere soggetti a guasti, ma il servizio è sempre disponibile

Soft State → La consistenza dei dati non è garantita in ogni istante

Eventually Consistent → Il sistema diventa consistente dopo un certo intervallo di tempo, se le attività di modifica dei dati cessano

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione ai Database NoSQL

Il termine NoSQL identifica **una moltitudine di DBMS**, basati su modelli logici differenti

- Database **chiave/valore**
- Database **document-oriented**
- Database **column-oriented**
- Database **graph-oriented**

Database Chiave-Valore

Esempi: **BerkeleyDB, Project Voldemort, Redis**

Dati di un DB come liste di coppie chiave/valore (array associativi o dizionari)

Chiave → valore univoco per operazioni di ricerca
Valore → qualsiasi cosa

Chiave	Valore
1	{Mario Rossi, 02311323}
2	{Mario Bianchi, 23}
3	{Dipartimento Informatica, Via Triboletti, 0858942005}

Introduzione ai Database NoSQL

Il termine NoSQL identifica **una moltitudine di DBMS**, basati su modelli logici differenti

- Database **chiave/valore**
- Database **document-oriented**
- Database **column-oriented**
- Database **graph-oriented**

Database Document-Oriented

- Esempi: **MongoDB**, **CouchDB**
- Gestione di dati **eterogeneei** e **complessi** (semi-strutturati)
- Scalabili **orizzontalmente**, supporto per partizionamento (*sharding*) dei dati in sistemi distribuiti
- **Documenti** → coppie chiave/valore (JSON)
- Forniscono **funzionalità** per aggregazione/analisi dei dati (MapReduce)

Database Document-Oriented

Documento → collezione coppie chiave-valore

```
{name: ‘Claudio’,
cognome: ‘Rossi’,
eta: 22,
data:new Date(2021,4,3,12,30)
address: {via: Triboletti, numero: 7}
           {via: Milli, numero: 49}
email: [rossi@gmail.com,
rossi@tiscali.it]
}
```

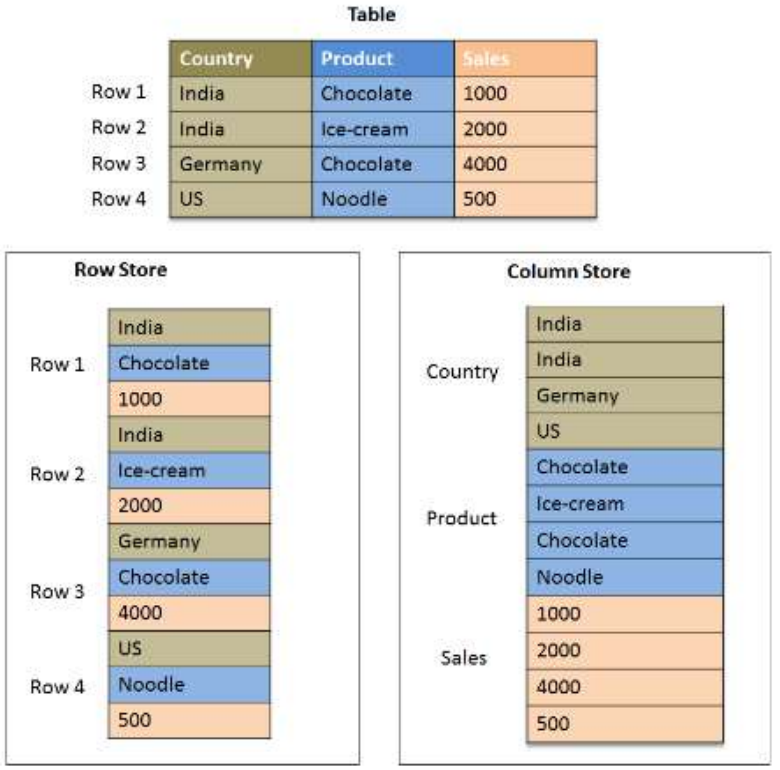
Introduzione ai Database NoSQL

Il termine NoSQL identifica **una moltitudine di DBMS**, basati su modelli logici differenti

- Database **chiave/valore**
- Database **document-oriented**
- Database **column-oriented**
- Database **graph-oriented**

Database Column-Oriented

- Esempi: **HBase**, **Cassandra, HanaDB**
- Dati organizzati su colonne anziché su righe
- Column family → contenitore di colonne
  Ogni column family è scritta su un file diverso.
  Ogni riga dispone di una chiave primaria (**row key**)



Database Chiave-Valore

- Schema **flessibile**
- Maggiore efficienza nello **storage**
- Maggiore possibilità di **compressione dati**
- Usato in sistemi dati **read-oriented** (es. warehousing)

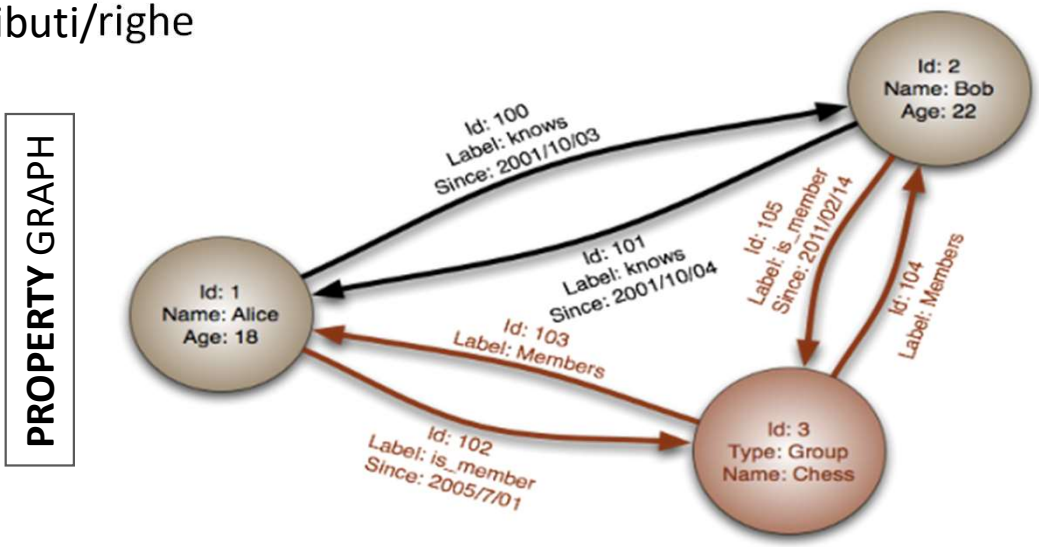
Introduzione ai Database NoSQL

Il termine NoSQL identifica **una moltitudine di DBMS**, basati su modelli logici differenti

- Database **chiave/valore**
- Database **document-oriented**
- Database **column-oriented**
- Database **graph-oriented**

Database Graph-Oriented

- Esempi: **Neo4J**, **Titan**
- Dati strutturati sotto forma di grafi:
  nodi → attributi/righe
  archi → relazioni tra attributi/righe



UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al
Data Mining

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Data Mining: tecniche di **apprendimento computerizzato** per analizzare ed estrarre **conoscenze** da collezioni di dati

Pattern e **relazioni** non note a priori e non immediatamente identificabili

Disciplina complessa: utilizzo di tecniche di **machine learning**, **intelligenza artificiale** e **statistica**

Introduzione al Data Mining

REAL ESTATE DATASET

Date	Age House	Distance to nearest MRT station	Number of convenience stores	House Price
2012,917	32	84,87882	10	137,9
2012,917	19,5	306,5947	9	142,2
2013,583	13,3	561,9845	5	147,3
2013,500	13,3	561,9845	5	154,8
2012,833	5	390,5684	5	143,1
…	…	…	…	…

Goal 1. Scoprire correlazioni tra il prezzo della casa e la sua età

Goal 2. Scoprire le regole che consentono di predire il prezzo di una casa a partire dai suoi attributi

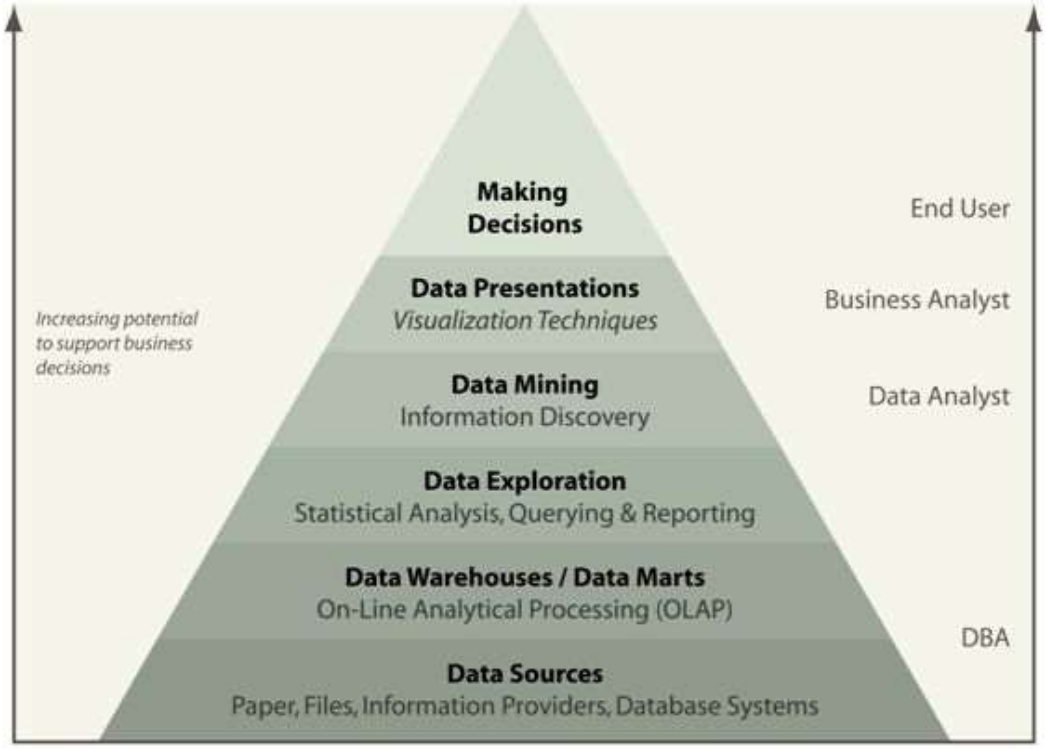
UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

BUSINESS INTELLIGENCE (BI) → (def.) Insieme di processi aziendali, metodologie tool per raccogliere i dati di un'azienda, ed estrarre **informazioni di supporto alla decisioni strategiche**

DATA MINING → componente essenziale del processo di BI, si occupa di estrarre informazioni utili dai dati per aiutare il processo decisionale

Introduzione al Data Mining



SOURCE: http://www.conbusinessintelligence.com/

UNIVERSITÁ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Data mining → estrae informazioni da un DB

Data query (**SELECT**) → estrae dati da un DB relazionale (in particolare, dalle tabelle della **FROM**)

Q. Che differenza esiste tra i due approcci?

A. Il processo di data mining estrae **regolarità e pattern sui dati** che non sono note a priori, e che non possono essere ricavate da query SQL

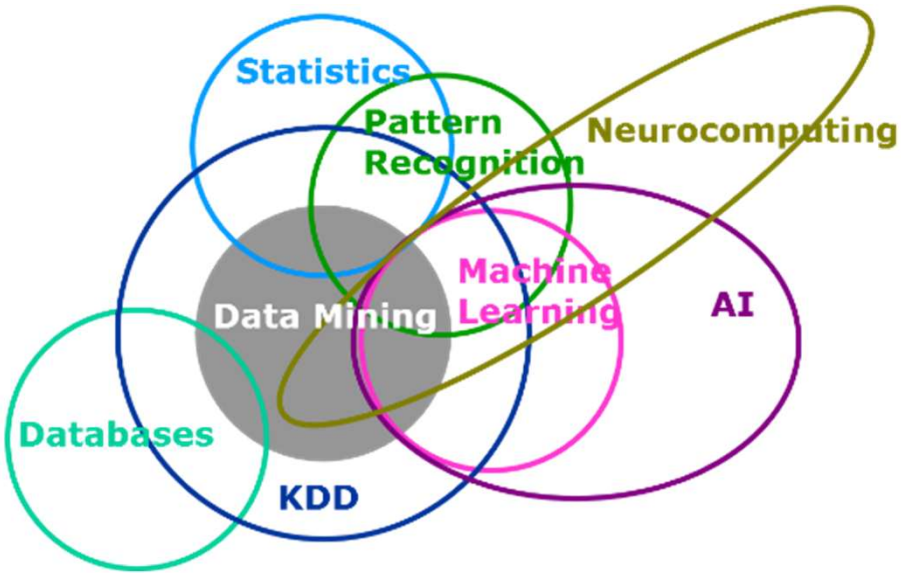
Introduzione al Data Mining

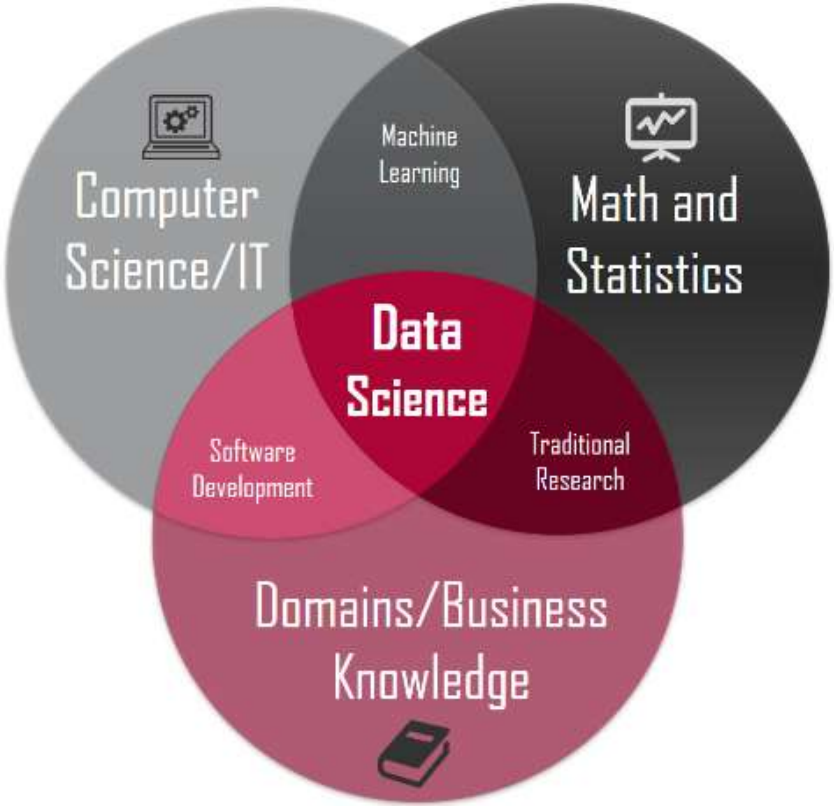
ESEMPI di APPLICAZIONI (aziendali)

- **Recommendation system**
- **Previsioni di dati temporali** (es. vendite)
- **Market Basket Analysis** (vi siete mai chiesti come mai tanti tornei di golf sono sponsorizzati da societa' di brokeraggio?)
- **Scoperta di truffe** (es. clonazioni di carte di credito)
- **Campagne pubblicitarie** mirate
- **Churn Analysis** (analisi della clientela che potrebbe passare alla concorrenza)
- **Segmentazione** della clientela
- …

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining





UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Q. **Da dove derivano i dati da analizzare?**

Social Media

Sistemi Aziendali (es. ERP)

Web (es. OPEN DATA)

Introduzione al Data Mining

DATA MINING ATTRAVERSO CASI D'USO

Che tipo di conoscenza posso estrarre dai dati?

1. REGRESSIONE

UNIVERSITÀ DEGLI STUDI DI TERAMO

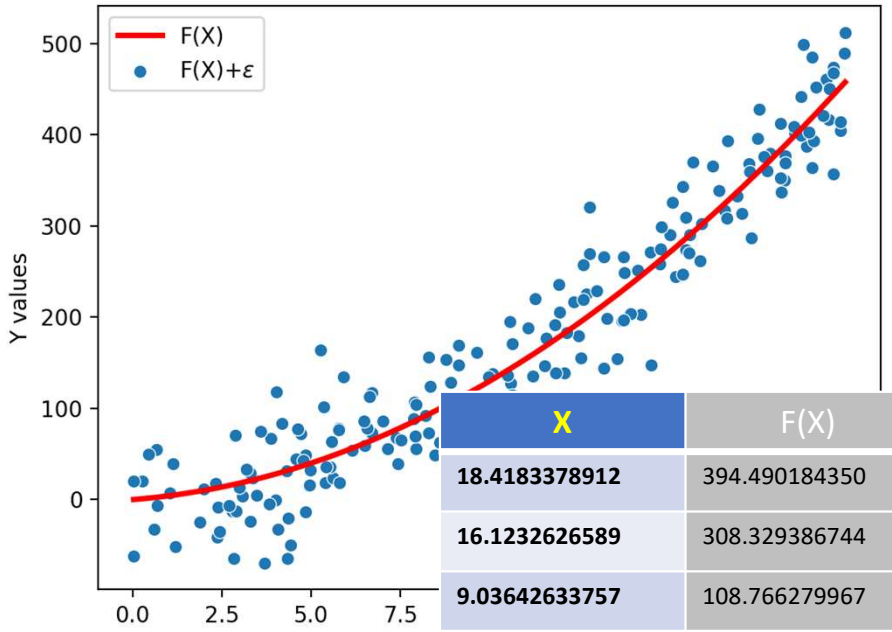
Introduzione al Data Mining

- Dato un attributo **quantitativo** *y* e *p* differenti attributi del dataset ($x_1, x_2, \ldots, x_p$)
- Trovare la relazione che lega *y* e $\boldsymbol{x} = (x_1, x_2, \ldots, x_p)$: $y = f(\boldsymbol{x}) + \epsilon$
- ϵ denota la distribuzione dell'errore casuale, che è indipendente da $\boldsymbol{x}$ e ha media pari a zero
- La funzione f che connette l'input e l'output è **sconosciuta** → L'obiettivo è trovare una buona stima di f che denotiamo come $\hat{f}$

$$\hat{y} = \hat{f}(\boldsymbol{x})$$

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining



$$Y = F(X) + \epsilon$$

$$Y = X^2 + 3X + \mathcal{N}(0,40)$$

DATASET

X	F(X)	ϵ	Y=F(X) +ϵ
18.4183378912	394.490184350	37.5662180915	**356.923966259**
16.1232626589	308.329386744	-2.87868760592	**305.450699138**
9.03642633757	108.766279967	51.0748597503	**159.841139717**
8.47101497537	97.1711396392	-42.7643064745	**54.40683316466**
…	…	…	…

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining

- Analisi pagine FB delle **Destination Management Organizations** (DMO) su scala regionale
- Analisi utilizzo dei social media per fini di marketing del turismo
- Individuazione best-practice per pubblicazione dei contenuti

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining

Impatto del profilo FB misurato attraverso l'**engagement**:

$$\frac{(Likes + Comments + Shares)}{(Total\_Posts * Total\_Fans(end\_of\_the\_month))} * 100$$

Quale fattore incide **positivamente** sull'engagement?
Quale fattore incide **negativamente** sull'engagement?

STRATEGIE PER PUBBLICAZIONE DEI CONTENUTI

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining

REGRESSIONE LINEARE

Coefficiente

$$Y = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \ldots + \beta_M * X_M$$

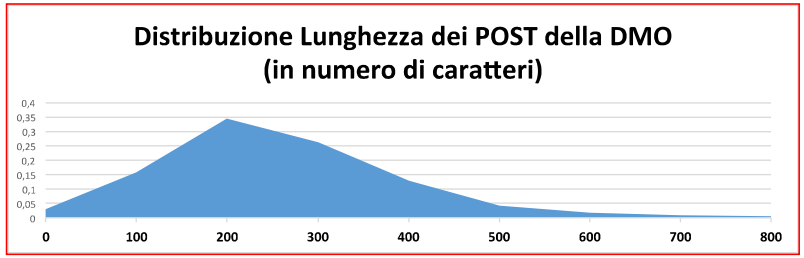
Variabile dipendente:
Engagement

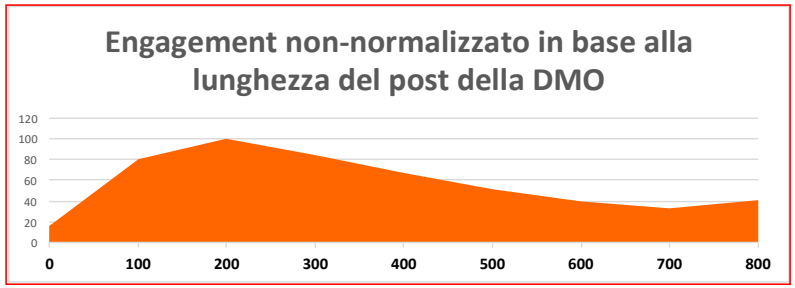
Variabile esplicativa:
Es. Geografia, Stagione, Tipo Post, Frequenza Post, etc

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining

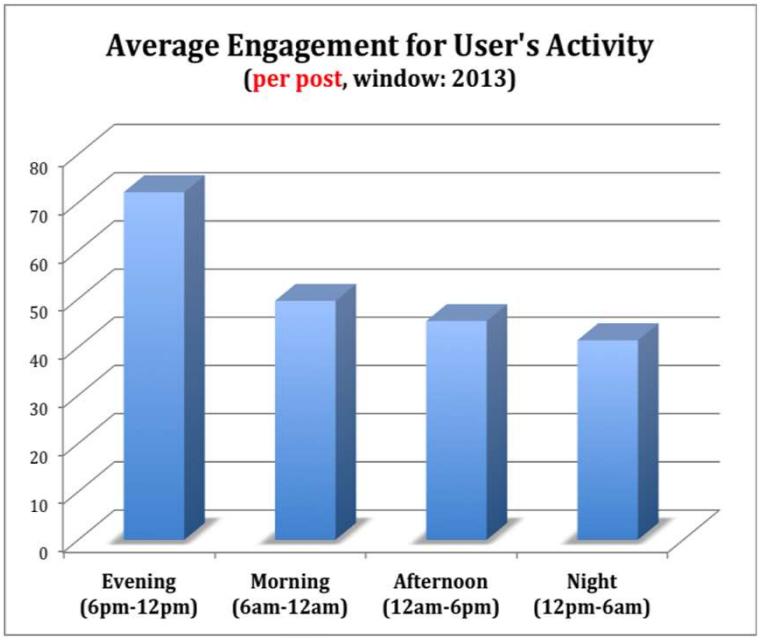




UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining



UNIVERSITÀ DEGLI STUDI DI TERAMO

Figure Professionali

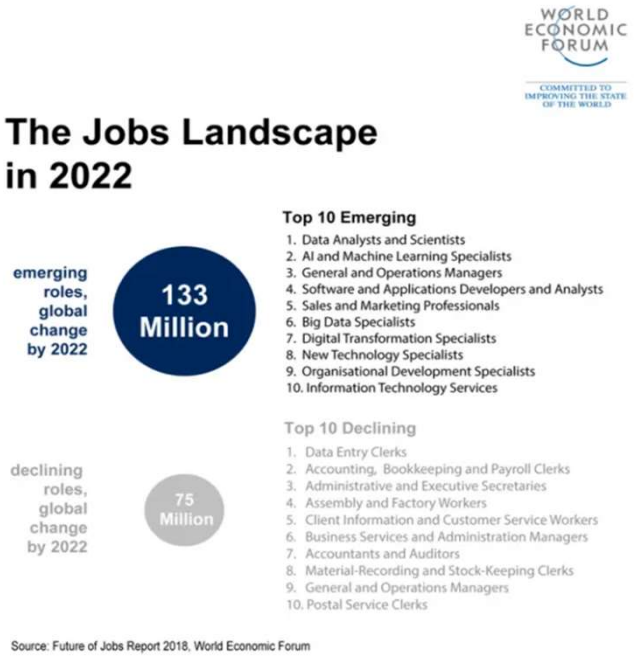
NUOVE PROFESSIONI

Perché serve un data scientist per ogni azienda

di Alberto Magnani

VIDEO

https://www.ilsole24ore.com/art/tecnologie/2017-01-13/perche-serve-data-scientist-ogni-azienda-181239.shtml?uuid=ADkcBTYC



WORLD ECONOMIC FORUM
COMMITTED TO IMPROVING THE STATE OF THE WORLD

The Jobs Landscape in 2022

emerging roles, global change by 2022

133 Million

Top 10 Emerging

1. Data Analysts and Scientists
2. AI and Machine Learning Specialists
3. General and Operations Managers
4. Software and Applications Developers and Analysts
5. Sales and Marketing Professionals
6. Big Data Specialists
7. Digital Transformation Specialists
8. New Technology Specialists
9. Organisational Development Specialists
10. Information Technology Services

declining roles, global change by 2022

75 Million

Top 10 Declining

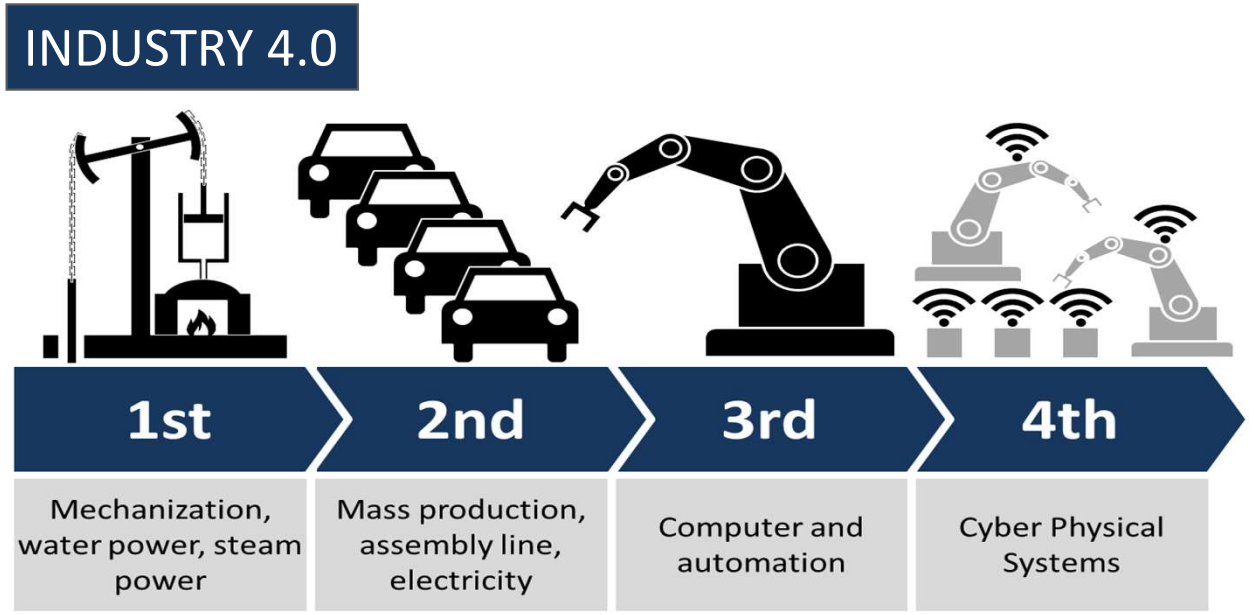
1. Data Entry Clerks
2. Accounting, Bookkeeping and Payroll Clerks
3. Administrative and Executive Secretaries
4. Assembly and Factory Workers
5. Client Information and Customer Service Workers
6. Business Services and Administration Managers
7. Accountants and Auditors
8. Material-Recording and Stock-Keeping Clerks
9. General and Operations Managers
10. Postal Service Clerks

Source: Future of Jobs Report 2018, World Economic Forum

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Nuovi contesti applicativi in ambito **aziendale**



UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Nuovi contesti applicativi in ambito **aziendale**

INDUSTRY 4.0

SENSING → DATA FILTERING → DATA ANALYSIS → MODEL BUILDING → DIGITAL SYSTEM

UNDERSTANDING
correlation between system parameters and between inputs/outputs

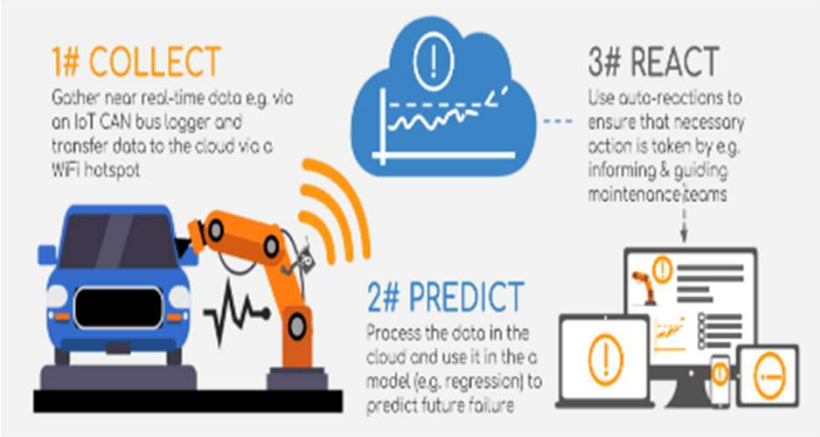
PREDICTING
system behaviour at time t+1

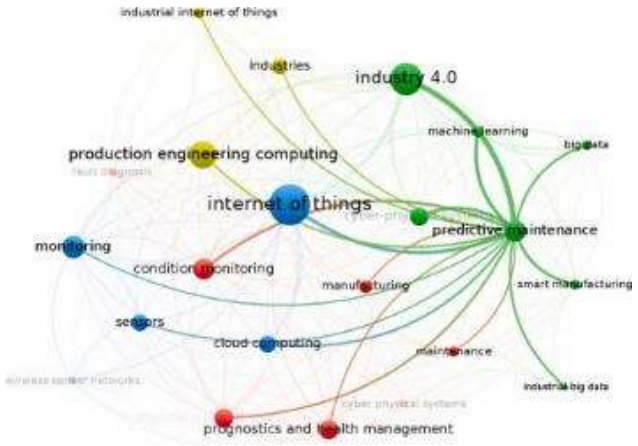
PLANNING
what-if analysis

Introduzione al Data Mining

Nuovi contesti applicativi in ambito **aziendale**

INDUSTRY 4.0: PREDICTIVE MAINTENANCE





UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

DATA MINING ATTRAVERSO CASI D'USO

Che tipo di conoscenza posso estrarre dai dati?

2. CLASSIFICAZIONE

Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining

SVILUPPO di APPLICAZIONI **CONTEXT-AWARE** PER IL RICONOSCIMENTO DELLA MOBILITA' UTENTE A PARTIRE DAI SENSORI DELLO SMARTPHONE

SCENARI D'USO:

(1) Adattare le funzionalità dell'app alla mobilità
(2) Fruizione di contenuti specifici in mobilità

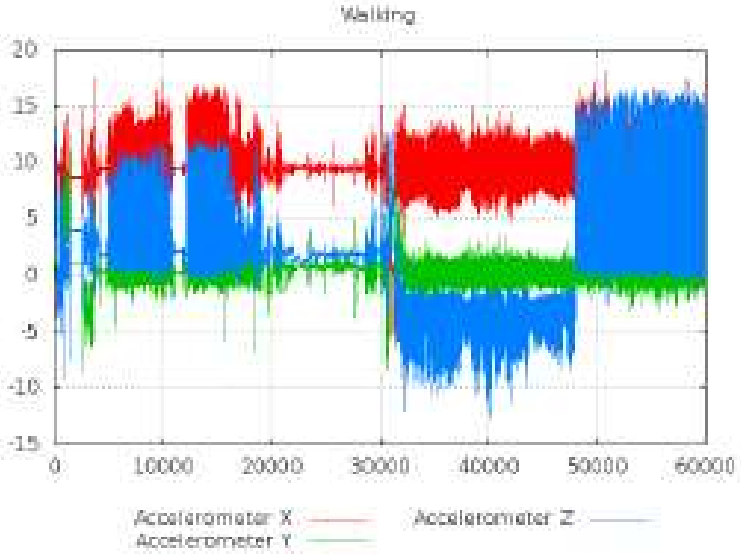
INPUT: Valori dei **sensori dello smartphone** (accelerometro, giroscopio, GPS, etc)

OUTPUT: Classi di mobilità (Piedi, Macchina, Treno, Bici, Autobus urbano, Autobus extra)

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining



(a) Accelerometer values (walking)

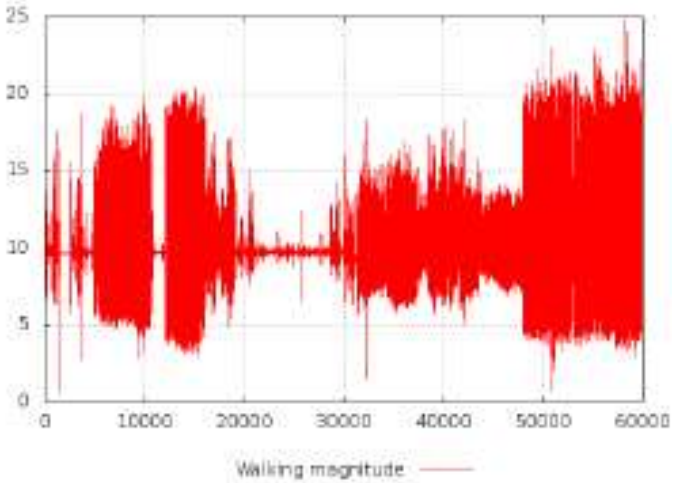
La funzione di **magnitude** consente di sintetizzare i valori di un sensore triassale in un unico valore indipendente dalla posizione o orientamento del dispositivo

$$magnitude(s) = |v_s| = \sqrt{v_{x,s}^2 + v_{y,s}^2 + v_{z,s}^2}$$

UNIVERSITÀ DEGLI STUDI DI TERAMO

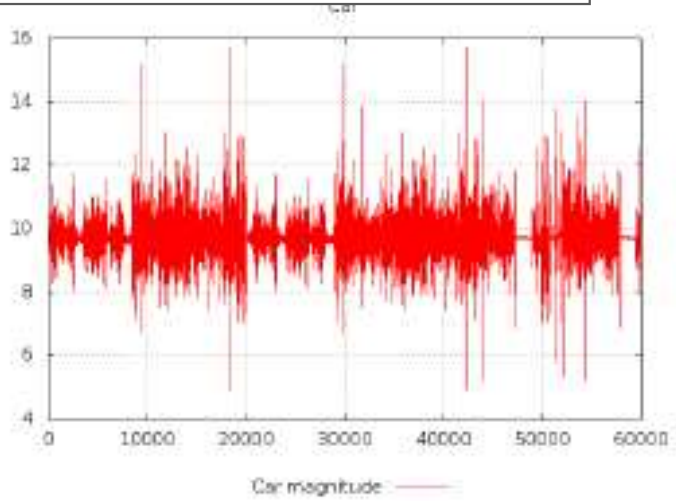
Introduzione al Data Mining

Un esempio di applicazione di tecniche di data-mining



(b) Accelerometer magnitude (walking)

WALKING



(c) Accelerometer magnitude (car)

CAR

UNIVERSITÁ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

DATA MINING ATTRAVERSO CASI D'USO

Che tipo di conoscenza posso estrarre dai dati?

3. REGOLE ASSOCIATIVE

Introduzione al Data Mining

RECOMMENDATION SYSTEMS in E-COMMERCE

80 percent of movies watched on Netflix
came through recommendations

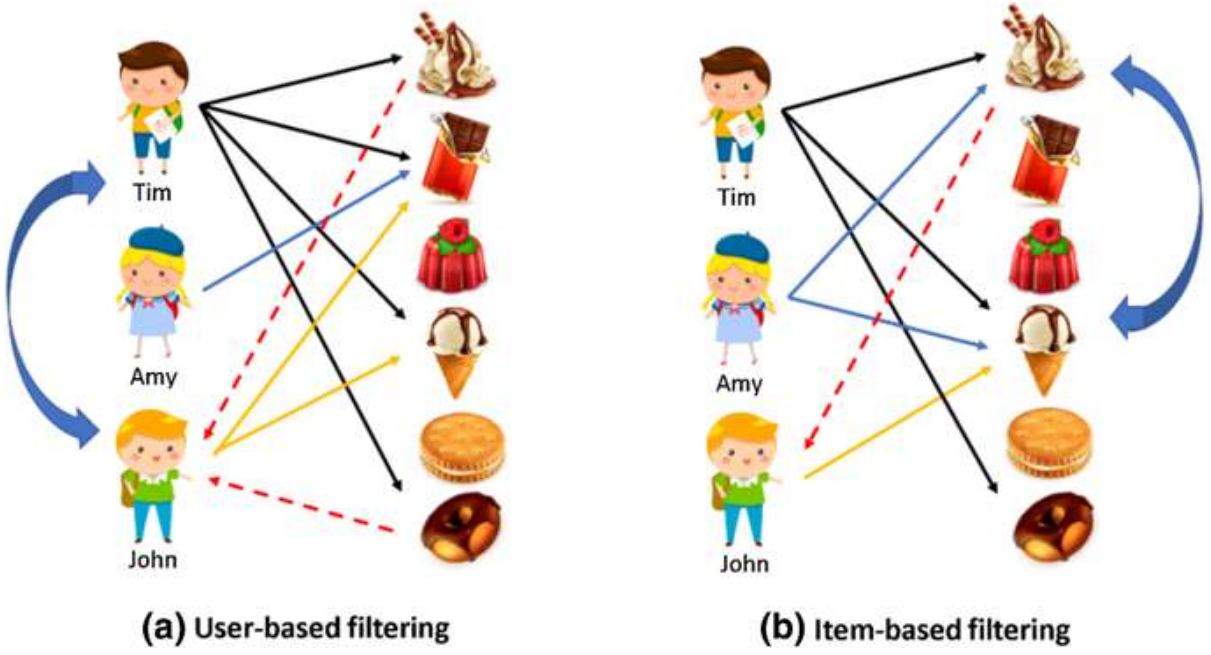
amazon

PUNTI DI ATTENZIONE

- Milioni di clienti, milioni di item
- Qualità delle recommendation
- Poche informazioni per i nuovi clienti, molte per clienti storici
- I dati dei clienti sono volatili

Introduzione al Data Mining

RECOMMENDATION SYSTEMS in E-COMMERCE

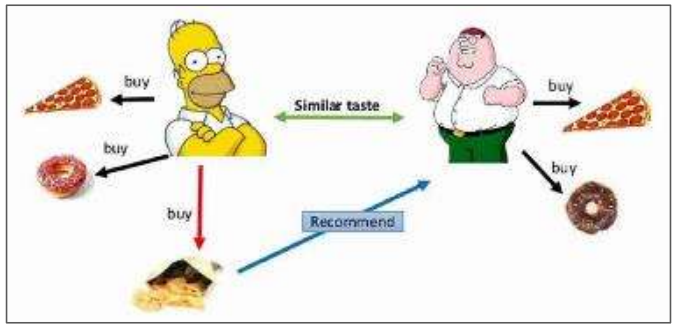


UNIVERSITÁ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

RECOMMENDATION SYSTEMS in E-COMMERCE

USER-BASED COLLABORATIVE FILTER



Segmenta la customer base in più gruppi (**cluster**)

Calcola un indice di **similarità** tra ogni gruppo ed l'utente corrente

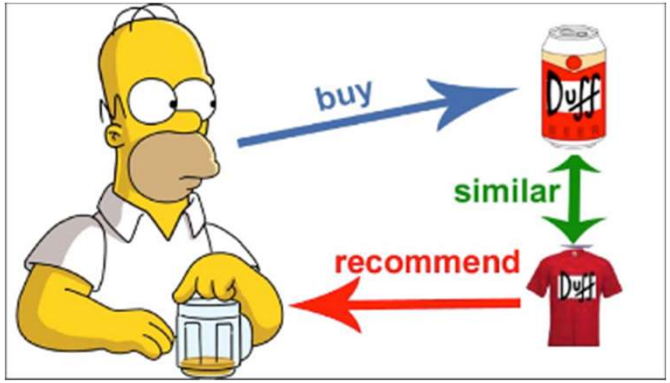
Seleziona il gruppo più simile, e suggerisci gli item più frequenti del gruppo

UNIVERSITÀ DEGLI STUDI DI TERAMO

Introduzione al Data Mining

RECOMMENDATION SYSTEMS in E-COMMERCE

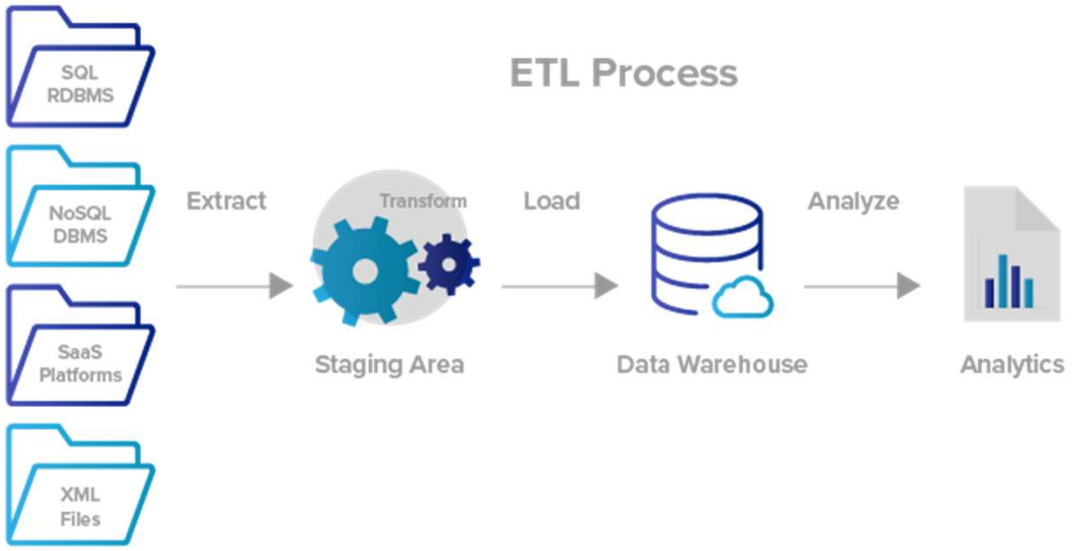
ITEM-BASED COLLABORATIVE FILTER



Focus su item simili più che su utenti simili → Identifica **item** che sono spesso acquistati assieme → Calcola la similarità tra oggetti acquistati assieme e restituisci quello con valore più alto

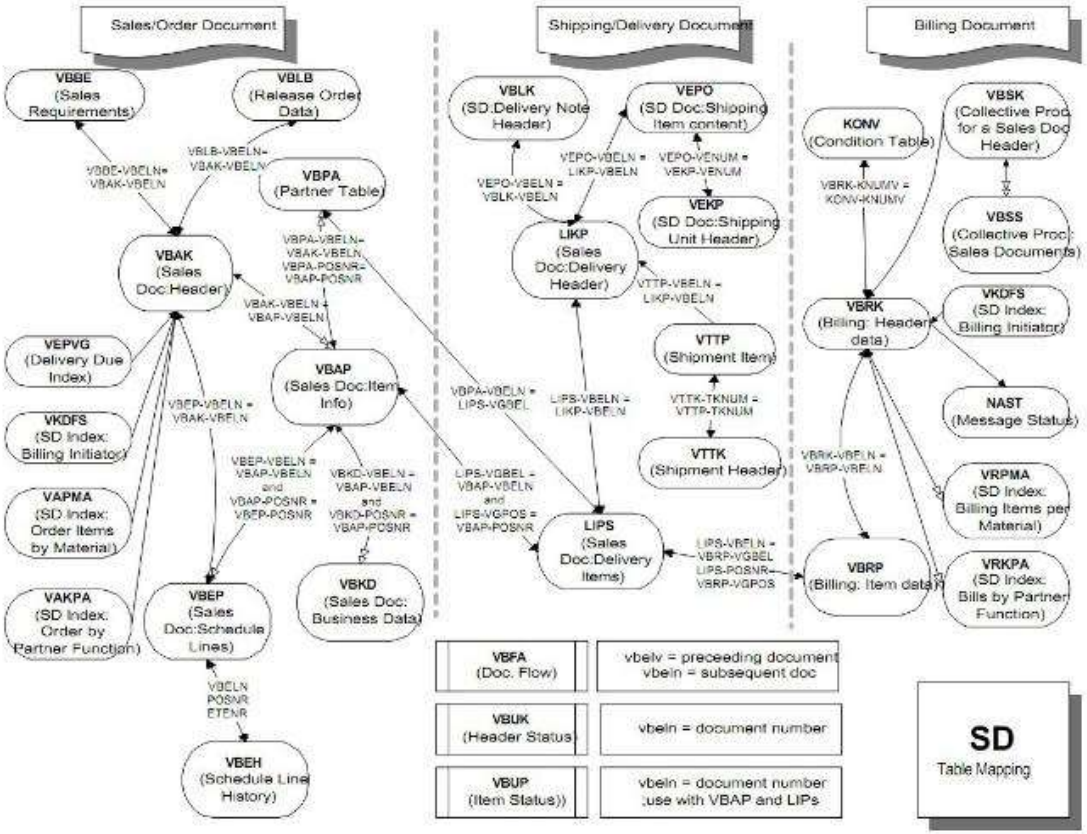
UNIVERSITÀ DEGLI STUDI DI TERAMO

Processo ETL



UNIVERSITÀ DEGLI STUDI DI TERAMO

SAP SD E-R Diagram



UNIVERSITÁ DEGLI STUDI DI TERAMO