Basi di Dati
Sistemi Informativi (modulo BD)

Franco Scarselli
franco at ing.unisi.it
http://www.dii.unisi.it/~franco

Programma

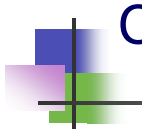
- Introduzione alle basi di dati
  - Sistemi informativi
  - DBMS - Database Management Systems
  - Modelli dei dati
  - Linguaggi per basi di dati
- Il modello relazionale
  - Prodotto cartesiano e relazioni
  - Relazioni e tabelle
  - Relazioni e basi di dati
  - Vincoli di integrità
  - Chiavi - chiavi primarie
  - Vincoli di integrità referenziale

Programma

- Algebra relazionale
  - Operatori dell'algebra relazionale
  - Unione, intersezione, differenza
  - Ridenominazione, selezione, proiezione
  - Join
  - Interrogazioni in algebra relazionale
  - Calcolo relazionale
- SQL (argomenti svolti in laboratorio)
  - Definizione dei dati
  - Interrogazioni
  - Manipolazione dei dati
  - Vincoli di integrità
  - Controllo dell'accesso

Programma

- Progettazione di basi di dati
  - Il modello Entità-Relazione
  - Strategie di progetto
  - Ristrutturazione di schemi E-R
  - Traduzione verso il modello relazionale
  - La normalizzazione
  - Forme normali e decomposizioni
- Introduzione a OLAP, big data, data mininng
  - On-Line Analytical Processing: datawarehouse, datamart e implementazione
  - Big data, data mining: definizioni e tecnologie



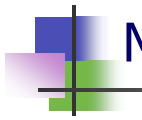
Contatti e ricevimento

Ricevimento

- Su appuntamento

Materiale didattico e home page

- L'home page del corso è accessibile sull'home page del corso su moodle



Materiale didattico

Slides

- Le slides conterranno tutti gli argomenti trattati nel corso

Libri consigliati per approfondimenti

- Atzeni, Ceri, Paraboschi, Torlone
  Basi di dati – Modelli e Linguaggi di Interrogazione
  McGraw-Hill
  (include la maggior parte degli argomenti)
- Atzeni, Ceri, Paraboschi, Torlone
  Basi di dati – Architetture e linee di evoluzione
  McGraw-Hill
  (include OLAP)

Modalità di verifica

- 2 prove intermedie scritte (contengono esercizi) 80%
- Orale (sulla teoria) 20%
- Progetto piccolo punteggio in aggiunta/in meno

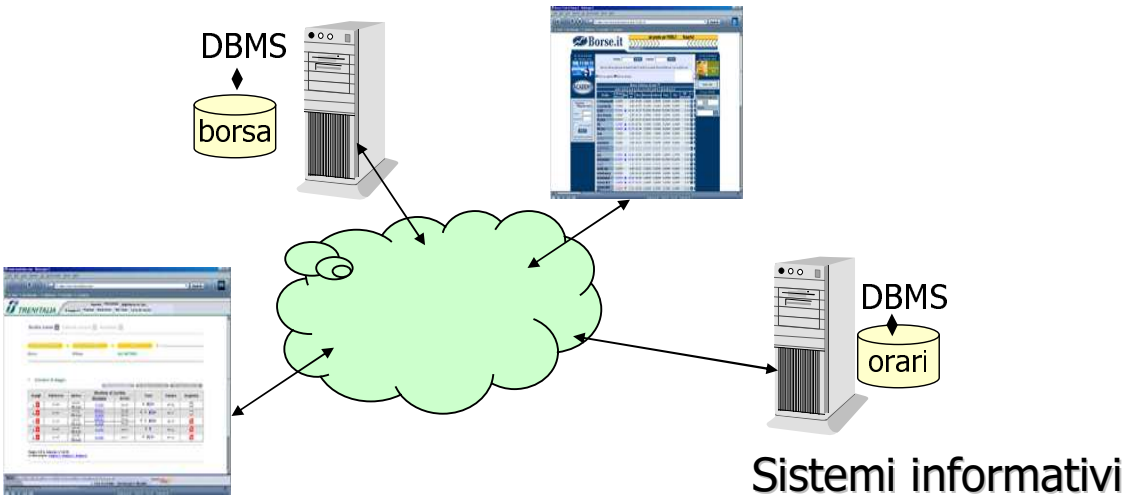
- Le prove intermedie non fatte/insufficienti si recuperano durante gli appelli
- Il progetto si consegna durante gli appelli
- Non esiste un ordine predefinito con cui superare le 4 parti, ma si suggerisce di fare il progetto dopo aver studiato la parte relativa agli esercizi
- Maggiori dettagli

Progetto

- Richiede la progettazione di un piccolo database usando tutte le tecniche che impareremo
- Si può fare in 1 o 2 studenti
- Si consegna durante gli appelli
- Mi potete chiedere una prima revisione prima degli appelli

- Esiste un progetto di esempio su moodle
- Maggiori dettagli su moodle

Introduzione



Sistemi informativi

Informazioni

Dati

Sistema informativo

- Sottosistema di una organizzazione che gestisce
  - acquisizione
  - elaborazione
  - conservazione
  - produzione

  delle informazioni di interesse
- Ogni organizzazione ha un sistema informativo
  - Le informazioni sono un "bene"
- L'esistenza del sistema informativo è indipendente dalla sua automazione (sistema informatico)

Sistema informatico

- Gestisce un sistema informativo in modo automatizzato
- Garantisce che i dati siano conservati in modo permanente sui dispositivi di memorizzazione
- Permette un rapido aggiornamento dei dati per riflettere rapidamente le loro variazioni
- Rende i dati accessibili alle interrogazioni degli utenti
- Può essere distribuito sul territorio
- Anche prima di essere automatizzati, molti sistemi informativi si sono evoluti verso una

razionalizzazione e standardizzazione delle procedure

e dell'organizzazione delle informazioni

Gestione delle informazioni

- Modo di gestire e comunicare le informazioni

  accesso, elaborazione, trasmissione

  - lingua scritta e parlata
  - disegni, grafici, schemi
  - codici e numeri

- Modo di memorizzare l'informazione
  - ricordata a memoria
  - copia cartacea
  - in formato elettronico

Sistemi per la gestione di informazioni

- Nelle attività standardizzate dei sistemi informativi complessi, sono state introdotte col tempo

  forme di organizzazione e
  codifica delle informazioni

- Ad esempio, nei servizi anagrafici si è iniziato con registrazioni discorsive e poi
  - nome e cognome
  - estremi anagrafici
  - codice fiscale .... una chiave per tutti!

Informazioni e dati

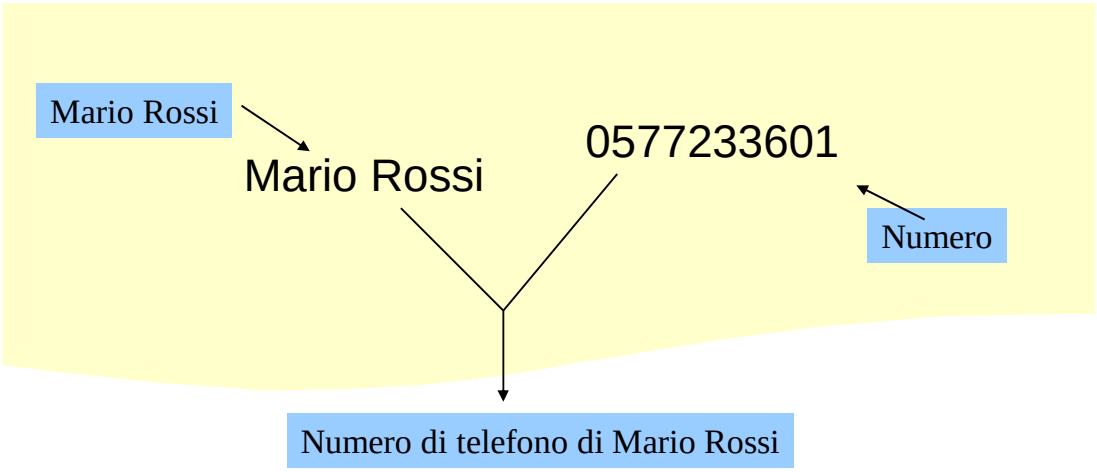
- Le informazioni vengono rappresentate e codificate in modo essenziale attraverso

  dati

- I dati devono essere interpretati per recuperare l'informazione
  - **informazione.** Notizia, dato o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni, modi di essere.
  - **dato.** Ciò che è immediatamente presente alla conoscenza, prima di ogni elaborazione; (in informatica) elementi di informazione costituiti da simboli che debbono essere elaborati.

Dai dati all'informazione

- L'informazione è ottenuta interpretando e correlando fra loro i dati



Un esempio: la rubrica telefonica

Dati

Nome Cognome Indirizzo Telefono

Mario Rossi Via Garibaldi 10 05773576373

Operazioni

- Cercare un numero dato il nome
- Inserire un nuovo numero
- Modificare un indirizzo
- ……

Base di dati

- E' una collezione di dati utilizzata per rappresentare le informazioni di interesse in un sistema informativo
- L'elemento chiave è il dato
  - La rappresentazione precisa di forme più ricche di informazione e conoscenza è difficile
  - I dati costituiscono spesso una risorsa strategica, perché più stabili nel tempo di altre componenti (processi, tecnologie, ruoli umani)
  - I dati rimangono gli stessi nella *migrazione* da un sistema al successivo

Basi di dati: quali applicazioni?

- Anagrafe - anagrafe studenti
- Servizi bancari - conti correnti - bancomat - carte di credito
- Elenchi delle utenze telefoniche
- Orari ferroviari
- Prenotazioni voli
- Catalogo prodotti (CD,…) - E-commerce
- Cataloghi di biblioteche
- Cartelle cliniche
- .....

Basi di dati: come si accede?

- in modo distribuito (Sportelli bancomat, POS negozi...)
- da Internet (Orari FFSS, siti di e-commerce, cddb, ...)
- da terminali dedicati (Agenzie di viaggio, biblioteche,anagrafe...)
- sul proprio personal (database personali)
- .....

Gestione di basi di dati

- Utilizzare un linguaggio di programmazione generico (C, C++, COBOL, Java) memorizzando i dati su file su disco
  - Applicazioni specifiche
  - Duplicazione dei dati
  - Incoerenza dei dati condivisi
- Utilizzare un sistema di gestione di basi di dati

  (Database Management System - DBMS)

  - Sistema software progettato per gestire dati
  - Capacità di gestire collezioni di dati grandi, condivise, persistenti

Caratteristiche dei DBMS

- grandi dimensioni
  - Possono avere dimensioni di Terabyte
  - Fanno uso della memoria secondaria
- condivisione
  - accesso di più utenti/applicazioni a dati comuni
  - si riduce la ridondanza dei dati e la possibilità di inconsistenze
  - Il DBMS deve disporre del controllo di concorrenza
- persistenza
  - i dati hanno una vita che va oltre quella dell'esecuzione delle applicazioni che li utilizzano

Requisiti dei DBMS

- affidabilità
  - capacità di non perdere i dati in caso di malfunzionamento hardware o software (backup e recovery)
- privatezza
  - ciascun utente è riconosciuto da un *username* e una *password*
  - un utente ha autorizzazione solo per certe operazioni
- efficienza
  - capacità di svolgere le operazioni in tempo ragionevole e con risorse di calcolo e memoria accettabili
  - dipende dall'implementazione
  - può imporre dei vincoli sull'hardware

DBMS vs filesystem [1]

- La gestione di insiemi di dati grandi e persistenti è possibile anche attraverso sistemi più semplici

  il file system del sistema operativo

- I file system prevedono forme di condivisione *tutto o niente*. Nei DBMS c'è maggiore flessibilità
- I DBMS estendono le funzionalità del file system, fornendo più servizi ed in maniera integrata
- Comunque, i DBMS, a causa della varietà di funzioni, non sono necessariamente più efficienti dei file system

DBMS vs filesystem [2]

- Nei programmi tradizionali che accedono a file, ogni programma contiene una descrizione della struttura del file stesso, con i conseguenti rischi di incoerenza fra le descrizioni (ripetute in ciascun programma) e i file stessi.
- Nei DBMS, esiste una porzione della base di dati (il catalogo o dizionario) che contiene una descrizione centralizzata dei dati, che può essere utilizzata dai vari programmi.

Descrizione dei dati nei DBMS

- Livelli diversi di descrizione e rappresentazione dei dati
  - permettono l'indipendenza dei dati dalla rappresentazione fisica
  - i programmi fanno riferimento alla struttura a livello più alto
  - le rappresentazioni sottostanti possono essere modificate senza necessità di modifica dei programmi
- Si introduce il concetto di

  modello dei dati
  - modalità usata per organizzare i dati e descriverne la struttura
  - offre meccanismi di strutturazione simili ai costruttori di tipo dei linguaggi di programmazione

I modelli dei dati

- modelli logici - utilizzati nei DBMS esistenti per l'organizzazione dei dati
  - utilizzati dai programmi
  - indipendenti dalle strutture fisiche
- modelli concettuali - permettono di rappresentare i dati in modo indipendente da ogni sistema
  - cercano di descrivere i concetti del mondo reale
  - sono utilizzati nelle fasi preliminari di progettazione

Modello Entity-Relationship (E-R)

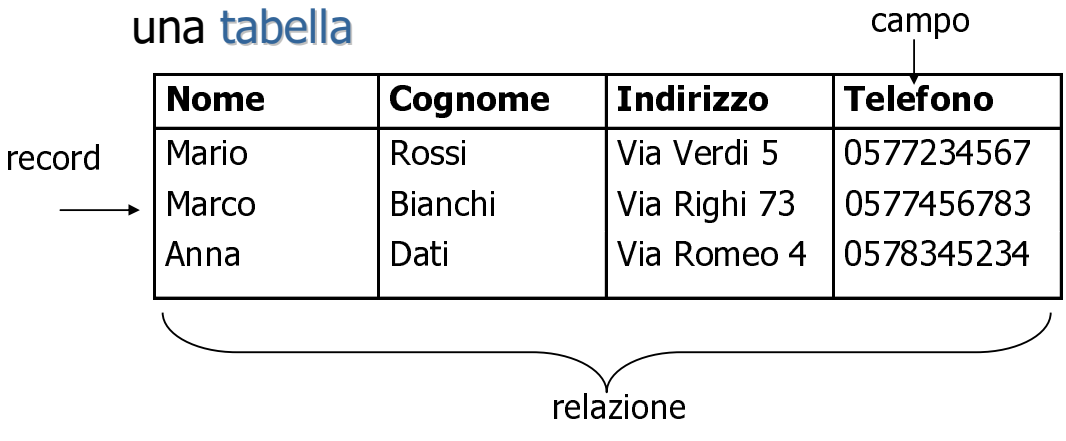
I modelli logici dei dati

- Esistono vari modelli (logici) di dati proposti
  - modello gerachico ('60)

    Basato sull'uso di strutture ad albero (gerarchie).
  - modello reticolare - CODASYL ('70)
    Basato sull'uso di grafi.
  - modello relazionale ('70-'80)
    Basato sull'uso del concetto di relazione.
  - modello ad oggetti ('80)
    Evoluzione del modello relazionale. Basato sulle idee della programmazione ad oggetti.

Il modello relazionale

- Permette di organizzare in insiemi di record a struttura fissa (concetto di relazione)
- Una relazione è spesso rappresentata per mezzo di una tabella



Schemi ed istanze

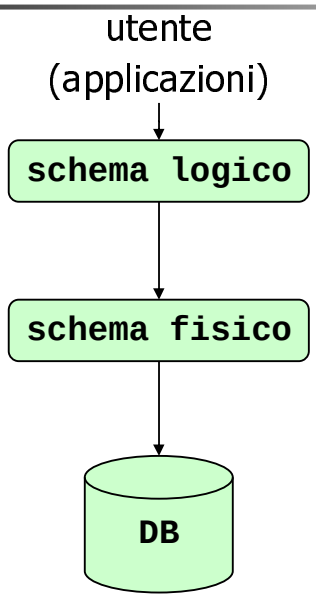
- Schema - Parte sostanzialmente invariabile dei dati (componente intensionale)

RUBRICA(Nome,Cognome,Indirizzo,Telefono)

- Istanza - valori effettivi contenuti nella base di dati (stato) (componente estensionale)

Mario	Rossi	Via Verdi 5	0577234567
Marco	Bianchi	Via Righi 73	0577456783
Anna	Dati	Via Romeo	0578345234

Schemi e astrazione



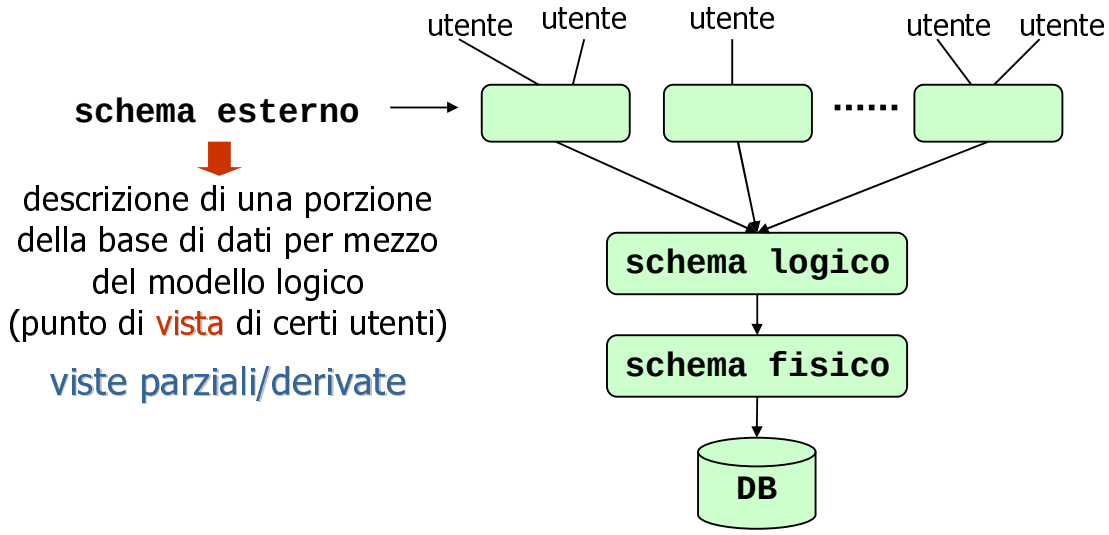
descrizione della base di dati nel modello logico (es, la struttura della tabella)

rappresentazione dello schema logico per mezzo di strutture memorizzazione (file)

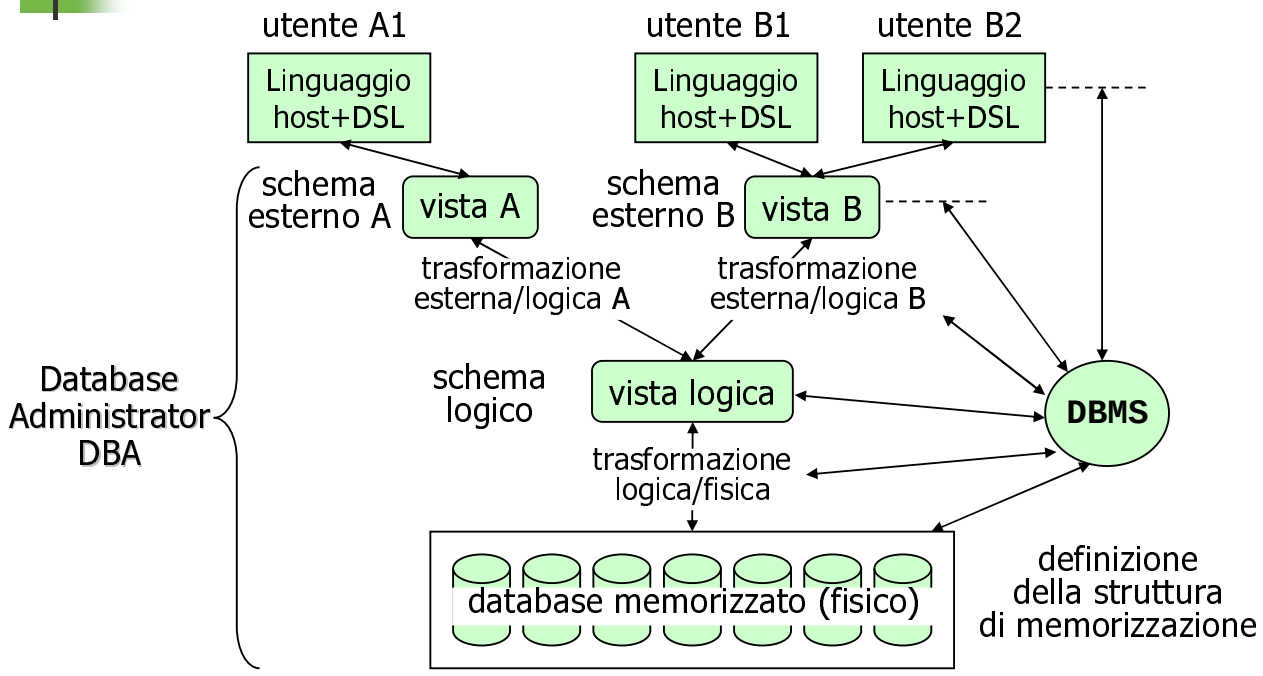
il livello logico è indipendente da quello fisico
una tabella è utilizzata nello stesso modo qualunque sia la sua realizzazione fisica (che può anche cambiare nel tempo)

Architettura ANSI/SPARC

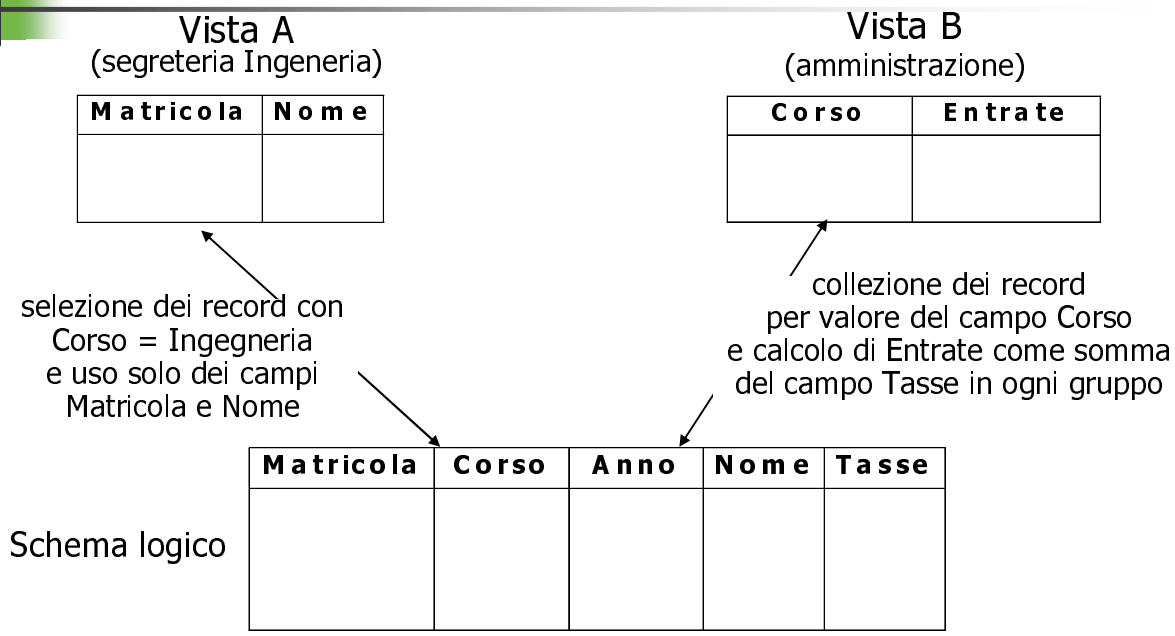
Architettura a 3 livelli proposta dall'ANSI/SPARC Study Group on DBMS



Architettura dettagliata



Le viste esterne



Il DataBase Administrator

- E' la persona (o gruppo di persone) responsabile di
  - progettazione (logica/fisica)
  - definizione delle politiche di sicurezza (autorizzazioni)
  - amministrazione (backup, monitoraggio delle prestazioni)

  della base di dati
- Media le esigenze degli utenti mantenendo un controllo centralizzato sui dati
- Configura il DBMS per garantire sufficienti prestazioni
- Assicura l'affidabilità del sistema
- Gestisce le autorizzazioni di accesso.

Utenti di un DBMS

- Progettisti e programmatori di applicazioni
  - realizzazione di programmi che accedono alla base di dati
  - uso di linguaggi convenzionali (C, Java, PHP, ..) o linguaggi proprietari specifici del sistema con funzionalità specifiche (generazione di grafici, stampe, maschere sul video)
  - uso di strumenti di sviluppo per la creazione di interfacce verso la base di dati
- Utenti finali
  - interazione basata su transazioni (attività frequenti e predefinite) utilizzando form o programmi guidati da menu (applicazioni sviluppate ad hoc dai programmatori)
  - oppure uso di linguaggi interattivi e interrogazioni *casuali* (non predefinite)

Transazioni

- Programmi che realizzano attività frequenti e predefinite, con poche eccezioni, previste a priori
- Esempi:
  - versamento presso uno presso sportello bancario
  - emissione di certificato anagrafico
  - prenotazione aerea
- Le transazioni sono di solito realizzate con programmi in linguaggio ospite (tradizionale o ad hoc)
- Il termine transazione ha un'altra accezione, più specifica: sequenza indivisibile di operazioni (o vengono eseguite tutte o nessuna)

Linguaggi per basi di dati

- Linguaggi per la definizione dei dati

  Data Definition Languages - DDL

  definizione degli schemi logici, fisici e delle autorizzazioni di accesso
- Linguaggi di manipolazione dei dati

  Data Manipulation Languages - DML

  interrogazione e aggiornamento delle basi di dati

Alcuni linguaggi come SQL (Structured Query Language) presentano integrate funzioni di entrambe le categorie

SQL come DDL

- Definizione di una tabella (relazione)

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

orario

insegnamento	docente	aula	ora

SQL come DML

- Interrogazione

```
SELECT Insegnamento, Docente
       FROM Orario
       WHERE insegnamento = 'Basi di dati'
```

Produce una tabella (relazione) che soddisfa il criterio (clausola WHERE)

insegnamento	docente
Basi di dati	Maggini

Linguaggi per basi di dati

- Disponibilità di vari linguaggi, interfacce diverse e supporto alla scrittura di programmi
  - linguaggi testuali interattivi (Command Line Interface CLI, ad esempio basate su SQL)
  - comandi immersi in un linguaggio ospite (es. SQLJ java embedded SQL)
  - disponibilità di librerie di interfaccia col DBMS per linguaggi di programmazione generici (es. JDBC - Java DataBase Connectivity)
  - comandi in un linguaggio ad hoc, anche con funzionalità specifiche (es. per grafici o stampe strutturate)
  - con interfacce amichevoli (senza linguaggio testuale)

SQLJ Java embedded SQL

- Supportato da Oracle, IBM, Sybase, Tandem, JavaSoft, Microsoft, Informix, XDB
  - Implementazione http://www.oracle.com/st/products/jdbc/sqlj
- E' un linguaggio semplice e coinciso per inglobare comandi SQL in programmi Java
- Lo standard garantisce la portabilità fra diversi DBMS
- Si inseriscono i comandi SQL nel programma
  - i comandi SQLJ iniziano con **"#sql"** e terminano con **";"**
  - si può far riferimento alle variabili Java usando il prefisso **":"**
  - il testo SQL è posto fra parentesi graffe **"{..}"**

SQLJ vs JDBC

- JDBC (Java DataBase Connectivity) mette a disposizione una libreria di classi per accedere al DBMS eseguendo comandi SQL

```
//  SQLJ
int n;
#sql {INSERT INTO emp
       VALUES (:n)};
```

variabile Java

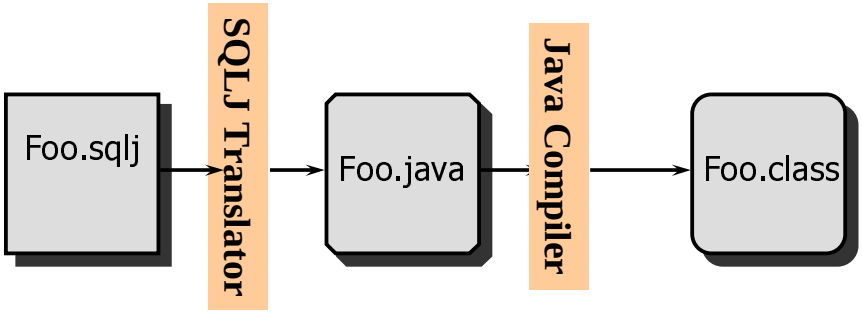
variabile Java nell' sqlj

classe della libreria java.sql

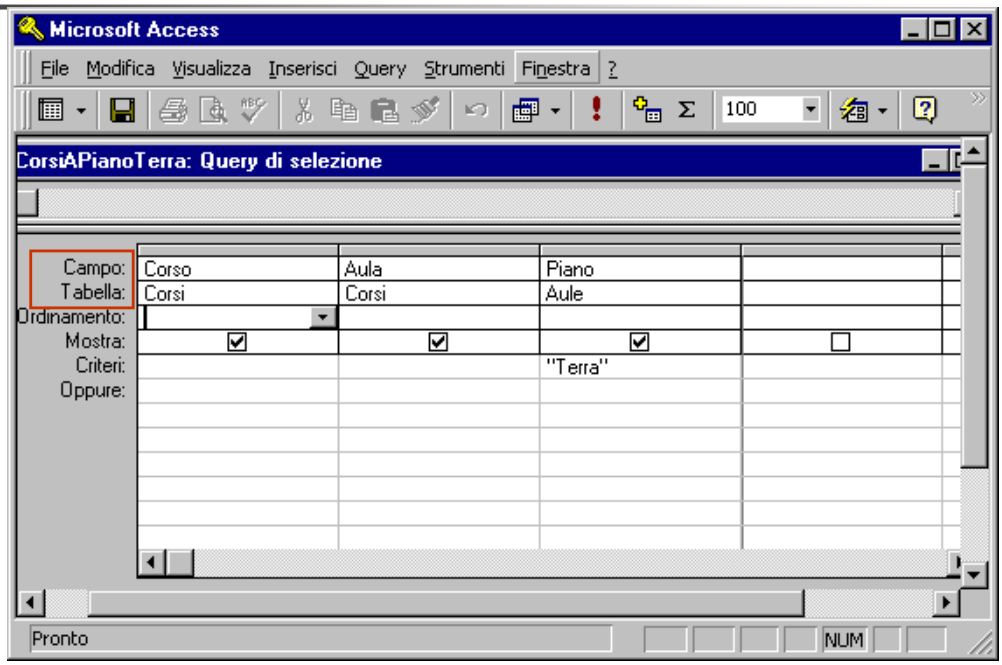
```
//  JDBC
int n;
Statement stmt =
        conn.prepareStatement
       ("INSERT INTO emp VALUES (?)");
stmt.setInt(1,n);
stmt.execute ();
stmt.close();
```

Da SQLJ a Java

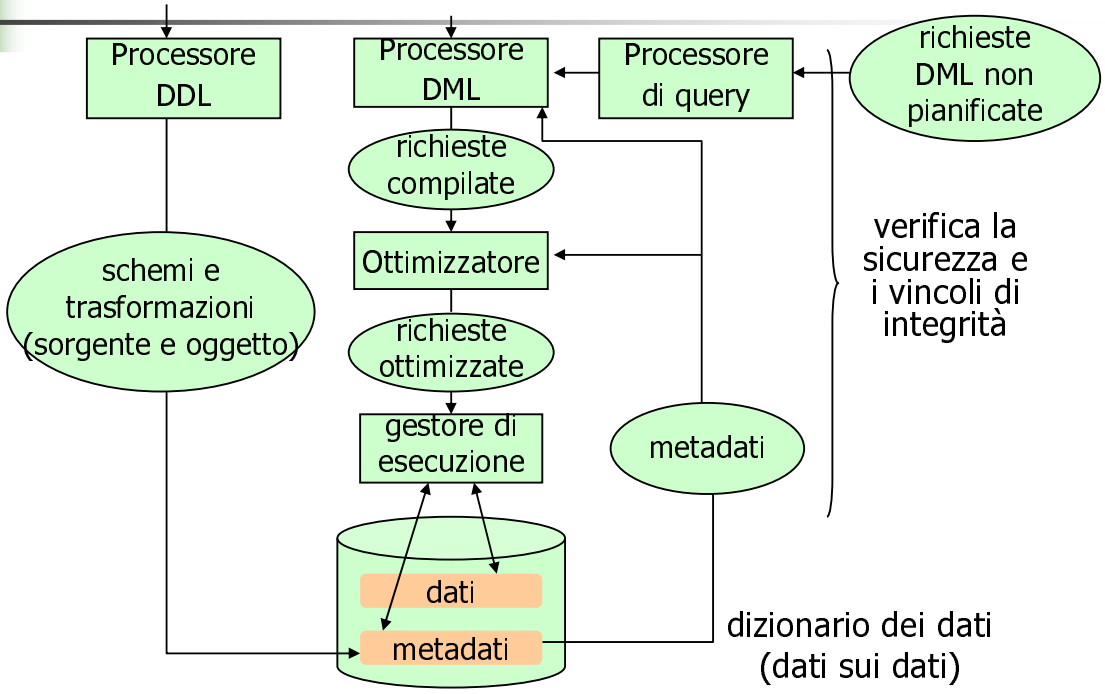
- Il file sorgente SQLJ deve essere tradotto in un sorgente Java puro che può essere compilato in un file class



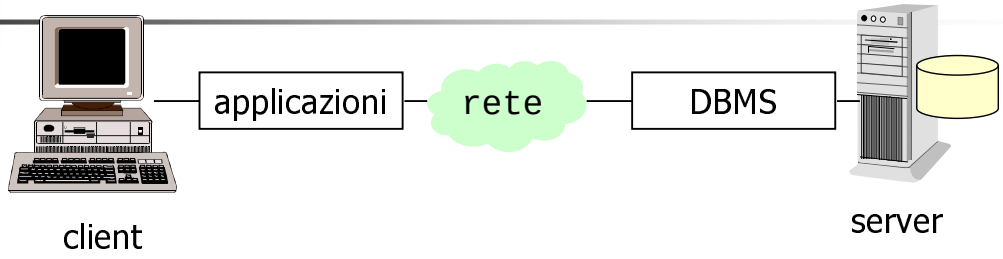
Interazione grafica



Struttura di un DBMS



Organizzazione distribuita



- L'applicazione database può essere organizzata col modello client/server
- Il client e il sever (DBMS) possono essere eseguiti su macchine diverse
- Uno stesso server può servire più client
- Un client può utilizzare più servers (database distribuito)

Vantaggi e svantaggi dei DBMS

Pro

- dati come risorsa comune a disposizione di tutta l'organizzazione
- gestione centralizzata con possibilità di standardizzazione
- disponibilità di servizi integrati
- riduzione di ridondanze e inconsistenze grazie alla condivisione
- indipendenza dei dati (favorisce lo sviluppo e la manutenzione delle applicazioni)

Contro

- costo dei prodotti e della transizione verso di essi (strutture, hardware, personale)
- non scorporabilità di alcuni servizi non utili (con riduzione di efficienza)

Il modello relazionale
Codd, 1970

- Adottato dalla maggior parte dei DBMS in commercio
- Definisce come sono organizzati i dati e non come sono poi memorizzati e gestiti dal sistema informatico

Relazione - Tabella

Il concetto di relazione proviene dalla matematica mentre quello di tabella è intuitivo

Il modello relazionale permette di trattare i dati ad un livello logico senza preoccuparsi del livello fisico ovvero di come i dati sono effettivamente elaborati e memorizzati. Per accedere ai dati non è necessario conoscere le strutture fisiche con cui sono realizzati!

1

1

Implicazioni

- Struttura

  I dati sono visti come tabelle

- Integrità

  Le tabelle soddisfano dei vincoli di integrità

- Manipolazione

  Gli operatori a disposizione per manipolare la tabelle (relazioni) sono operatori che derivano tabelle (relazioni) da tabelle (relazioni)

2

2

Prodotto cartesiano

- Consideriamo l'insieme dei Nomi e dei numeri di telefono dei dipendenti

Mario Rossi	2345
Luca Verdi	2367
Anna Bianchi	2378
	2356
D_1	D_2

Il prodotto cartesiano $D_1 x D_2$ è l'insieme di tutte le coppie ordinate (NOME,TELEFONO)

3

3

Prodotto cartesiano 2

Mario Rossi	2345
Mario Rossi	2367
Mario Rossi	2378
Mario Rossi	2356
Luca Verdi	2345
Luca Verdi	2367
Luca Verdi	2378
Luca Verdi	2356
Anna Bianchi	2345
Anna Bianchi	2367
Anna Bianchi	2378
Anna Bianchi	2356

- Il prodotto cartesiano contiene tutte le possibili associazioni fra gli elementi degli insiemi
- La rubrica dei numeri telefonici contiene solo alcune di tutte le possibili coppie

4

4

Relazioni

- Una relazione matematica sugli insiemi D_1 e D_2 (domini) è un sottoinsieme del prodotto cartesiano $D_1 \times D_2$
- Le relazioni si possono visualizzare efficacemente con una tabella in cui ogni colonna corrisponde ad un dominio e ogni riga a un elemento della relazione

La relazione RUBRICA

Mario Rossi	2345
Luca Verdi	2367
Luca Verdi	2378
Anna Bianchi	2356

La rubrica contiene solo le coppie (NOME,TELEFONO) che esistono

5

5

Definizione generale

- Si possono considerare *n* domini non necessariamente distinti $D_1, D_2, \ldots, D_n$
- Il prodotto cartesiano $D_1 \times D_2 \times \ldots \times D_n$ contiene tutte le possibile *n*-uple $(v_1, v_2, .., v_n)$ dove v_i è uno dei possibili valori presenti nel dominio D_i
- Una relazione R sui domini $D_1, D_2, \ldots, D_n$ è un sottoinsieme del prodotto cartesiano $R \subseteq D_1 \times D_2 \times \ldots \times D_n$
- Il numero delle componenti del prodotto (*n*) è detto grado della relazione; il numero di n-uple della relazione è la cardinalità della relazione.

Un dominio può avere dimensione infinita (es. i numeri naturali, D_i=N). In tal caso il prodotto cartesiano è infinito, mentre possono essere definite relazioni finite.

6

6

Un esempio

D_1 = Squadre di Serie A D_2 = Squadre di Serie A

D_3 = Numero D_4 = Numero

Relazione

Risultati delle partite della II giornata del campionato

Lazio	Inter	1	1
Fiorentina	Roma	3	1
Milan	Bari	0	0
Bologna	Parma	0	1
Udinese	Napoli	2	1
Lecce	Perugia	1	1
Reggina	Juventus	3	0
Atalanta	Brescia	2	2
Verona	Vicenza	0	0

7

7

Proprietà delle relazioni

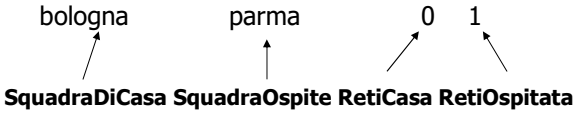
- una relazione matematica è un insieme di n-uple ordinate $(d_1, \ldots, d_n)$ tali che $d_1 \in D_1, \ldots, d_n \in D_n$
- una relazione è un insieme, quindi:
  - non c'è ordinamento fra le n-uple
  - le n-uple sono distinte
  - i valori all'interno di ciascuna n-upla sono ordinati l' i-esimo valore proviene dall' i-esimo dominio
- ciascuno dei domini ha un ruolo diverso, dipendente dalla posizione (struttura posizionale)

8

8

Relazioni con attributi

- Ogni *n*-upla della relazione stabilisce un legame fra i dati (definisce l'esistenza di un fatto)
- Il significato dei valori dipende dall'ordine con cui sono elencati nell'*n*-upla



- Si può associare un nome alle componenti della *n*-upla
- I nomi associati ai domini si dicono attributi e descrivono il ruolo che il dominio rappresenta nella relazione

9

9

Record e campi

- Ciascuna *n*-upla della relazione può essere vista come un record, ovvero una relazione è sostanzialmente un insieme di record omogenei
- Gli attributi definiscono la struttura del record ovvero i suoi campi
- Si può accedere al valore di un attributo di un record usando il nome del campo

tuple

SquadraDiCasa	SquadraOspitata	RetiCasa	RetiOspitata
Fiorentina	Roma	3	1
Milan	Bari	0	0
Bologna	Parma	0	1

Si usa il nome tupla per indicare una riga di *n* valori corrispondenti ad attributi

10

10

Attributi e tuple

- Se X è un insieme di attributi e D un insieme di domini, si stabilisce una corrispondenza fra attributi e domini con una funzione

  $$DOM: X \rightarrow D$$

  che associa a ciascun attributo $A \in X$ un dominio $DOM(A) \in D$

- La funzione DOM corrisponde alla definizione del tipo dati di ogni attributo
- Una tupla su un insieme di attributi X è una funzione t che associa a ciascun attributo $A \in X$ un valore in DOM(A)

11

11

Domini e tipi di dato

- Un dominio è praticamente un tipo di dato
- La definizione di un dominio comprende
  - L'insieme di tutti i valori possibili che può assumere un elemento del dominio (e quindi i valori ammessi per l'attributo)
  - Gli operatori che possono essere applicati ai valori di quel dominio

  Dominio: N (INTEGER)
  Operatori: comparazione (=,>,<), aritmetici (+,-,*,/)
- Non è necessariamente legata alla rappresentazione fisica dei dati

12

12

Le tuple

Data una tupla posso estrarne il valore degli attributi

SquadraDiCasa	**SquadraOspitata**	**RetiCasa**	**RetiOspitata**
Fiorentina	Roma	3	1

t[**SquadraOspitata**] = Roma

t[**SquadraDiCasa**] = Fiorentina

Si possono estrarre anche insiemi di attributi ottenendo tuple

t[**SquadraDiCasa**,**SquadraOspitata**]=

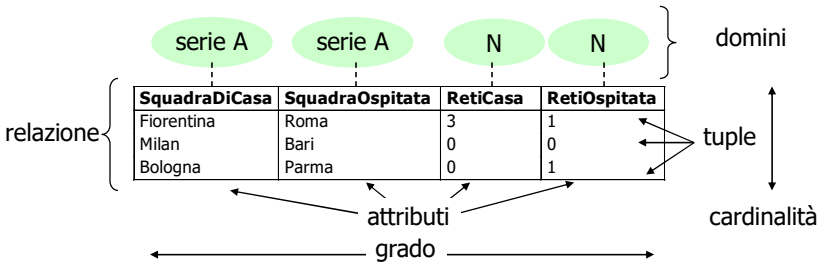
SquadraDiCasa	**SquadraOspitata**
Fiorentina	Roma

13

13

tuple e relazioni

- Una relazione su X è un insieme di tuple su X
- Si usa la definizione di tupla al posto di n-upla
  - in una n-upla gli elementi sono individuati per posizione
  - In una tupla gli elementi sono individuati per attributi



14

14

Tabelle e relazioni

- Una tabella rappresenta una relazione se
  - i valori di ogni colonna sono fra loro omogenei
  - le righe sono diverse fra loro
  - le intestazioni delle colonne sono diverse tra loro
- In una tabella che rappresenta una relazione
  - l'ordinamento tra le righe è irrilevante
  - l'ordinamento tra le colonne è irrilevante

15

15

Relazioni e basi di dati

- Una base di dati è in genere costituita da più relazioni
- Si possono creare corrispondenze fra le tuple di relazioni distinte
- Si ottengono corrispondenze fra tuple di relazioni diverse aventi gli stessi valori su un insieme assegnato di attributi
- Il modello relazionale è basato su valori
  - indipendenza dalle strutture fisiche
  - si rappresenta solo ciò che è rilevante
  - l'utente finale vede gli stessi dati dei programmatori
  - i dati sono portabili più facilmente da un sistema ad un altro
  - i puntatori sono direzionali

16

16

Corrispondenze fra relazioni

Matricola	Cognome	Nome	Data di nascita
A80198760	Bianchi	Anna	22/03/1967
A80293450	Rossi	Andrea	13/04/1968
A80198330	Neri	Luca	04/08/1970
A80295640	Felici	Lorenzo	25/02/1969
A80197456	Melli	Mara	17/10/1966

STUDENTI

Studente	Voto	Corso
A80198760	28	01
A80293450	30	04
A80198330	27	01
A80198330	25	03
A80197456	21	05

ESAMI

Codice	Titolo	Docente
01	Analisi I	Anna Verdi
03	Geometria	Andrea Pitagora
04	Fisica I	Luca Galileo
05	Chimica	Lorenzo Argenti

CORSI

17

17

Schemi per basi di dati

- Uno schema di relazione è costituito dal nome della relazione e da un insieme di attributi $X=\{A_1, A_2,\ldots, A_n\}$

  STUDENTI(Matricola,Cognome,Nome,Data di Nascita)

- Uno schema di base di dati è un insieme di schemi di relazioni con nomi diversi:

$$\mathbf{R}=\{R_1(X_1), R_2(X_2),\ldots, R_m(X_m)\}$$

R = { STUDENTI(Matricola,Cognome,Nome,Data di Nascita),
ESAMI(Studente,Voto,Corso),
CORSI(Codice,Titolo,Docente) }

18

18

Istanze di basi di dati

- Una istanza di relazione su uno schema R(X) è un insieme di r tuple su X
- Un'istanza di base di dati su uno schema

  $\mathbf{R}=\{R_1(X_1), R_2(X_2),\ldots, R_m(X_m)\}$

  è un insieme di relazioni

  $\mathbf{r} = \{r_1, r_2, \ldots, r_n\}$,

  dove ogni r_i, per $1 \leq i \leq n$, è una relazione sullo schema $R_i(X_i)$

19

19

Archivio forniture

Codice	Descrizione	Magazzino	Costo
A0001	Chiodi 35mm	234000	15
A0004	Viti 12mm	102400	25
P0010	Pinza	200	7000
P0230	Cacciavite	600	3500

PRODOTTI

Articolo	Quantita	Data	Cliente
A0001	3000	25/06/2000	0034
A0004	500	25/06/2000	0034
P0010	3	12/07/2000	0001
P0230	2	13/07/2000	0101
A0004	100	15/07/2000	0034

FORNITURE

Codice	Nome	Indirizzo	PIVA
0001	Carlo Berti	Via Roma 6	02332002
0034	BCD Spa	Via Verdi 4	04554303
0101	A&G Srl	Viale Morgagni 16	10920393
0076	Luca Nelli	Piazza Bixio 5	08832822

CLIENTI

20

20

Relazioni con un solo attributo

- Sono ammesse relazioni con un solo attributo
- Può essere utilizzata per selezionare un sottoinsieme delle tuple di un'altra relazione, se contiene valori che appaiono come valori di un attributo dell'altra relazione
  - Se gli elementi del sottoinsieme sono pochi rispetto al totale può essere più efficiente questa soluzione piuttosto che utilizzare un attributo booleano nella relazione

Matricola
A80198760
A80198330

Lavoratori

Matricola	Cognome	Nome	Data di nascita
A80198760	Bianchi	Anna	22/03/1967
A80293450	Rossi	Andrea	13/04/1968
A80198330	Neri	Luca	04/08/1970
A80295640	Felici	Lorenzo	25/02/1969
A80197456	Melli	Mara	17/10/1966

Studenti

21

21

Strutture nidificate

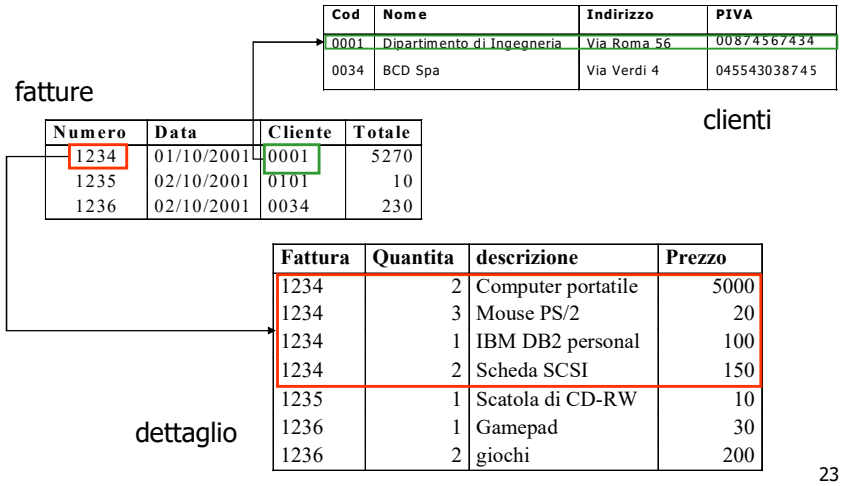
- Rappresentare col modello relazionale un archivio di fatture emesse in seguito ad una fornitura di materiale

Fattura n. 1234 del 1/10/2001		
A: Dipartimento di Ingegneria		
PI: 00874567434		
2	Computer portatile	5000
3	Mouse PS/2	20
1	IBM DB2 personal	100
2	schede SCSI	150
	Totale	5270

22

22

Relazioni per strutture nidificate



Cod	Nome	Indirizzo	PIVA
0001	Dipartimento di Ingegneria	Via Roma 56	00874567434
0034	BCD Spa	Via Verdi 4	045543038745

Numero	Data	Cliente	Totale
1234	01/10/2001	0001	5270
1235	02/10/2001	0101	10
1236	02/10/2001	0034	230

Fattura	Quantita	descrizione	Prezzo
1234	2	Computer portatile	5000
1234	3	Mouse PS/2	20
1234	1	IBM DB2 personal	100
1234	2	Scheda SCSI	150
1235	1	Scatola di CD-RW	10
1236	1	Gamepad	30
1236	2	giochi	200

23

23

Strutture nidificate: note

- La base di dati precedente rappresenta correttamente le fatture purché:
  - Non ci siano righe ripetute in una stessa fattura (le tuple devono essere uniche)
  - Non conti l'ordine con cui devono essere riportate le righe nella fattura
- Si può arricchire la rappresentazione in modo che sia rappresentato l'ordine del dettaglio?
- Il campo Totale della relazione Fatture è un attributo ridondante (potrebbe essere calcolato?)

24

24

Informazione incompleta

- Può accadere che il valore di un attributo non sia disponibile per tutte le tuple
- Non è corretto usare un valore del dominio non utilizzato (es. 0, ”99”, stringa vuota), per rappresentare l’assenza dell’informazione
  - Non è detto che esista un valore del dominio che non sia mai usato come valore valido (es. se l’attributo è una data)
  - L’uso dei valori del dominio può generare confusione: quando si accede ai dati si deve aver chiaro quali sono i valori veri e quelli fittizi
- Si deve estendere il concetto di relazione....

25

25

Il valore nullo

- Si introduce un valore nullo (NULL) che denota l’assenza di informazione sul valore di un attributo per una data tupla
- Il valore di un attributo A può appartenere a DOM(A) o essere NULL
- Il valore di un attributo può mancare perché
  - è sconosciuto
  - è realmente inesistente (l’attributo non è applicabile ad una data tupla)
  - è senza informazione (non si sa se esiste o meno)

Codice	Nome	Indirizzo	PIVA
0001	Carlo Berti	Via Roma 6	NULL
0034	BCD Spa	Via Verdi 4	04554303
0101	A&G Srl	Viale Morgagni 16	NULL

inesistente

sconosciuto

26

26

Tuple non corrette

- Occorre che le tuple rappresentino informazioni corrette per l'applicazione

Ex: Valori nulli

Non ammesso!

Matricola	Cognome	Nome	Data di nascita
NULL	Bianchi	Anna	22/03/1967
A80293450	NULL	NULL	NULL
A80197456	Melli	Mara	NULL

STUDENTI

????

Può essere accettabile se non è un attributo essenziale

Alcuni campi non possono assumere valori nulli!

27

Vincoli di integrità

- Un vincolo di integrità è una proprietà che deve essere soddisfatta dalle istanze della base di dati che rappresentano informazioni corrette per l'applicazione
- Un vincolo può essere visto come un predicato che associa ad un'istanza di base di dati il valore vero o falso
- Se il predicato assume il valore vero, l'istanza soddisfa il vincolo
- Ad uno schema di base di dati si può associare un insieme di vincoli di integrità. Un'istanza è ammissibile se soddisfa tutti i vincoli definiti.

28

Tipi di vincolo

- Vincolo intrarelazionale

  E' definito rispetto ad una singola relazione. Può riguardare:
  - le tuple nel complesso (es. non possono esistere due tuple con uno stesso valore per un dato attributo A - vincoli di chiave)
  - le tuple indipendentemente le une dalle altre (vincolo di tupla)
  - i valori ammessi per un attributo (vincolo di dominio)
- Vincolo interrelazionale

  Coinvolge più relazioni
  - "riferimenti appesi" (vincoli di integrità referenziale)

29

Vincoli intrarelazionali

- Sono vincoli che coinvolgono il valore degli attributi all'interno di una stessa relazione

ESAMI

Studente	Voto	Lode	Corso
A80198760	37		01
A80293450	30	L	04
A80198330	22		01
A80198330	25	L	03

Valore non ammesso

Combinazione non ammessa

Non ci possono essere due studenti con matricola uguale!

Matricola	Cognome	Nome	Data di nascita
A80198760	Bianchi	Anna	22/03/1967
A80293450	Rossi	Andrea	13/04/1968
A80198760	Felici	Lorenzo	25/02/1969
A80197456	Melli	Mara	17/10/1966

STUDENTI

30

Vincoli interrelazionali

- Sono vincoli che coinvolgono più relazioni
- Garantiscono l'integrità dei riferimenti fra tabelle

Studente	Voto	Corso
A80198760	28	01
A80293450	30	04
A80198330	27	01
A80198330	25	03
A80197456	21	05

ESAMI

?

Codice	Titolo	Docente
01	Analisi I	Anna Verdi
03	Geometria	Andrea Pitagora
04	Fisica I	Luca Galileo

CORSI

31

31

Vincoli di tupla

- Esprimono condizioni sui valori degli attributi ciascuna tupla indipendentemente dalle altre
- Sono espressi con predicati che devono essere veri per tutte le tuple

(Voto≥18) AND (Voto≤30)

NOT((Lode='L') AND (Voto≠30))

- Possono anche essere definite da espressioni aritmetiche che legano fra loro i valori degli attributi

PAGAMENTI(Data,Importo,Ritenute,Netto)

Netto = Importo-Ritenute

32

32

Chiavi

- I vincoli di chiave sono fondamentali
- Una chiave è un insieme di attributi utilizzato per identificare univocamente le tuple di una relazione

Matricola	Cognome	Nome	Data di nascita	Corso
A80198760	Rossi	Andrea	22/03/1967	Ingegneria
A80293450	Rossi	Luca	22/03/1967	Ingegneria
A80293856	Rossi	Filippo	25/02/1969	Fisica
A80198777	Bianchi	Filippo	25/02/1969	Lettere
A80197456	Bianchi	Filippo	22/10/1966	Lettere

Matricola
Nome, Cognome, Data di Nascita

SI

NO

Nome, Cognome, Corso
Cognome, Data di Nascita

33

Superchiavi e chiavi

- Un insieme K di attributi è superchiave per una relazione *r* se *r* non contiene due tuple distinte t_1 e t_2 con $t_1[K]=t_2[K]$
- Un insieme di attributi K è chiave per una relazione *r* se è una superchiave minimale (cioè non esiste un'altra superchiave K' che è contenuta propriamente in K)

CHIAVI (Superchiavi Minimali)

{Matricola}
{Nome, Cognome, Data di Nascita}

SUPERCHIAVI

{Matricola, Nome}
{Nome, Cognome, Data di Nascita, Corso}
{Matricola, Corso}
.....

34

Chiavi... per caso!

- Alcune combinazioni di attributi possono essere chiavi "per caso" per una data istanza di relazione

Matricola	Cognome	Nome	Data di nascita	Corso
A80198760	Rossi	Andrea	22/03/1967	Ingegneria
A80293450	Rossi	Luca	22/03/1967	Ingegneria
A80293856	Rossi	Filippo	25/02/1969	Fisica
A80198777	Bianchi	Filippo	25/02/1969	Lettere
A80197456	Bianchi	Filippo	22/10/1966	Lettere

- Niente vieta che prima o poi possano iscriversi allo stesso corso due persone nate lo stesso giorno e con lo stesso nome

35

35

Scelta delle chiavi

- Ogni relazione r(X) ha sempre una chiave
  - L'insieme di tutti gli attributi X è una superchiave dato che non possono esistere 2 tuple uguali
- La scelta della chiave deve tenere conto delle proprietà del mondo reale da cui provengono i dati
  - I vincoli sono definiti a livello di schema e devono essere validi per ogni istanza corretta nel mondo reale
- Può essere aggiunto un campo ad hoc (ad esempio la matricola) da usare come chiave

36

36

Importanza delle chiavi

- l'esistenza delle chiavi garantisce l'accessibilità a ciascun dato della base di dati
- le chiavi permettono di correlare le tuple in relazioni diverse (il modello relazionale è basato su valori e non su puntatori)
- ammettere valori nulli per gli attributi di una chiave può causare ambiguità

Matricola	Cognome	Nome	Data di nascita	Corso
A80198777	Bianchi	Filippo	25/02/1969	Lettere
NULL	Bianchi	Filippo	NULL	Lettere

?? ??

37

Chiavi primarie

- E' una chiave su cui sono vietati i valori nulli
- Si può sempre definire un attributo che ha la proprietà di essere una chiave primaria
  - non è detto che tale attributo abbia un significato per l'applicazione
  - può essere generato in modo automatico all'atto dell'inserimento (es. codice progressivo)

Matricola	Cognome	Nome	Data di nascita	Corso
A80198777	Bianchi	Filippo	25/02/1969	Lettere
A80710111	Bianchi	Filippo	22/09/1971	Lettere

38

Integrità referenziale

- Le corrispondenze fra i dati in relazioni diverse si stabiliscono per mezzo dei valori di chiavi delle tuple

AGENTI

Matricola	Cognome	Nome
672	Rossi	Mario
223	Verdi	Francesco
532	Belli	Carlo

INFRAZIONI

Codice	Data	Agente	Art	Prov	Numero
174655	25-12-1999	672	44	FI	G03635
156732	13-05-2000	223	12	SI	734563
456345	17-08-1999	672	44	GR	234322

AUTO

Prov	Numero	Proprietario	Indirizzo
FI	G03635	Gilli Luca	Via Oro
SI	734563	Tilli Nedo	Via Abete
GR	234322	Billi Aldo	Via Sole

39

39

Vincoli di integrità referenziale

- Un vincolo di integrità referenziale (foreign key) fra un insieme di attributi X di una relazione R_1 e un'altra relazione R_2 è soddisfatto se i valori su X di ciascuna tupla in R_1 compaiono come valori della chiave (primaria) dell'istanza di R_2

INFRAZIONI

Codice	Data	Agente	Art	Prov	Numero
174655	25-12-1999	672	44	FI	G03635
156732	13-05-2000	223	12	SI	734563
456345	17-08-1999	345	44	GR	234322

?

AGENTI

Matricola	Cognome	Nome
672	Rossi	Mario
223	Verdi	Francesco
532	Belli	Carlo

Occorre definire un vincolo di integrità referenziale fra l'attributo Agente della relazione INFRAZIONI e la relazione AGENTI (chiave primaria Matricola)

40

40

Un esempio

Targa	Proprietario	Indirizzo
BE370TZ	Fabio Rossi	Via Abete 13 Siena
FIG04546	Giovanni Neri	Via Moro 7 Firenze
AB234TV	Michele Bessi	Via Sila 4 Terni
TV443AZ	Maria Bianchi	Via Belli 4 Roma

AUTOVEICOLI

INCIDENTI

Codice	Targa1	Targa2	Luogo	Danni
5434	BE370TZ	AB234TV	Siena	Fari
5534	FIG04546	TV443AZ	Monteroni	Paraurti

1) vincolo di integrità referenziale fra l'attributo Targa1 della relazione INCIDENTI e la relazione AUTOVEICOLI
2) vincolo di integrità referenziale fra l'attributo Targa2 della relazione INCIDENTI e la relazione AUTOVEICOLI

41

41

Azioni compensative

- un'operazione causa una violazione di un vincolo di integrità referenziale... cosa accade?
- Es. cancellazione di una tupla referenziata
  - Rifiuto dell'operazione
  - Eliminazione in cascata
  - Introduzione di valori nulli

Matricola	Cognome	Progetto
A007	Rossi	672
A015	Verdi	223
A016	Belli	532

impiegati

progetti

Codice	Inizio	Durata
672	1/10/2001	12
223	1/12/2001	24
532	6/11/2001	6

42

42

Azioni compensative [esempio]

impiegati

Matricola	Cognome	Progetto
A007	Rossi	672
A015	Verdi	NULL
A016	Belli	532

progetti

Codice	Inizio	Durata
672	1/10/2001	12
~~223~~	~~1/12/2001~~	~~24~~
532	6/11/2001	6

sostituzione con valori nulli

impiegati

Matricola	Cognome	Progetto
A007	Rossi	672
~~A015~~	~~Verdi~~	~~223~~
A016	Belli	532

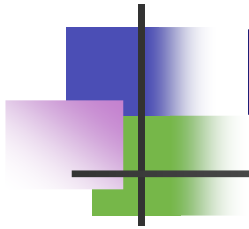
progetti

Codice	Inizio	Durata
672	1/10/2001	12
~~223~~	~~1/12/2001~~	~~24~~
532	6/11/2001	6

cancellazione in cascata

43

43



Linguaggi di interrogazione

- Un'interrogazione è una funzione che data una base di dati produce una relazione su un dato schema
- Linguaggi di interrogazione
  - Procedurali
    - specificano il procedimento di generazione del risultato ("come ottengo il risultato")
    - Algebra relazionale
  - Dichiarativi
    - specificano le proprietà del risultato ("che cosa è il risultato")
    - Calcolo relazionale

Algebra relazionale

- E’ basata su un insieme di operatori che sono definiti su relazioni e che producono relazioni
  - operatori insiemistici (unione, intersezione, differenza)
  - ridenominazione, selezione, proiezione
  - l’operatore di join (join naturale, prodotto cartesiano, theta-join)

- Si possono combinare gli operatori di base per ottenere interrogazioni anche complesse

Unione, Intersezione, differenza

- Questi operatori hanno senso solo se si applicano fra relazioni con lo stesso schema (con gli stessi attributi)

❶L'unione di due relazioni r_1 e r_2 è la relazione $r = r_1 \cup r_2$ che contiene le tuple che appartengono ad r_1 oppure ad r_2, oppure ad entrambe.

Matricola	**Nome**	**Cognome**
A80010012	Mario	Rossi
A80010200	Gianni	Longhi
A80010111	Fabio	Verdi

LAUREATI

Matricola	**Nome**	**Cognome**
A82010007	Marco	Beni
A82010345	Anna	Bianchi

SPECIALIZZATI

Matricola	**Nome**	**Cognome**
A80010012	Mario	Rossi
A80010200	Gianni	Longhi
A80010111	Fabio	Verdi
A82010007	Marco	Beni
A82010345	Anna	Bianchi

STUDENTI

Intersezione e differenza

- L'intersezione di due relazioni $r_1(X)$ e $r_2(X)$ definite su un insieme di attributi X è la relazione $r(X)= r_1(X) \cap r_2(X)$ contenente le tuple che appartengono sia a $r_1(X)$ che a $r_2(X)$
- La differenza di due relazioni $r_1(X)$ e $r_2(X)$ definite su un insieme di attributi X è la relazione $r(X)= r_1(X) - r_2(X)$ contenente le tuple che appartengono a $r_1(X)$ ma non a $r_2(X)$

Matricola	**Nome**	**Cognome**
3456575	Mario	Rossi
3432334	Gianni	Longhi
3223343	Fabio	Verdi

SPECIALIZZATI

Matricola	**Nome**	**Cognome**
3456575	Anna	Verdi
3223342	Isa	Belli
3432334	Gianni	Longhi
3222122	Fabio	Feltri

LAUREATI

Matricola	**Nome**	**Cognome**
3432334	Gianni	Longhi

SPECIALIZZATI ∩ LAUREATI

Matricola	**Nome**	**Cognome**
3456575	Mario	Rossi
3223343	Fabio	Verdi

SPECIALIZZATI - LAUREATI

Ridenominazione

- Modifica il nome di un sottoinsieme degli attributi di una relazione lasciando inalterato il contenuto delle relazione (modifica lo schema)
- L'operatore ha per argomento una relazione r(X) e indica la corrispondenza fra i nomi vecchi e nuovi di un sottoinsieme di attributi di X

$$\rho_{A1,A2,\ldots,An \leftarrow B1,B2,\ldots,Bn}(r)$$

dove $B1,B2,\ldots,Bn \in X$

Un esempio

IMPIEGATI

Matricola	Cognome	Agenzia	Stipendio
2321112	Belli	Milano	55
3432334	Longhi	Roma	45

OPERAI

Matricola	Cognome	Fabbrica	Salario
4434556	Sordi	Monza	35
4568797	Bianchi	Viterbo	33

Matricola	Cognome	Sede	Retribuzione
2321112	Belli	Milano	55
3432334	Longhi	Roma	45

$\rho_{Sede,Retribuzione \leftarrow Agenzia,Stipendio}(IMPIEGATI)$

Matricola	Cognome	Sede	Retribuzione
4434556	Sordi	Monza	35
4568797	Bianchi	Viterbo	33

$\rho_{Sede,Retribuzione \leftarrow Fabbrica,Salario}(OPERAI)$

Matricola	Cognome	Sede	Retribuzione
2321112	Belli	Milano	55
3432334	Longhi	Roma	45
4434556	Sordi	Monza	35
4568797	Bianchi	Viterbo	33

$\rho_{Sede,Retribuzione \leftarrow Agenzia,Stipendio}(IMPIEGATI) \cup \rho_{Sede,Retribuzione \leftarrow Fabbrica,Salario}(OPERAI)$

Selezione

- L'operatore di selezione $\sigma_{cond}(R)$ produce una relazione con lo stesso schema di R, che contiene le tuple di R che soddisfano la condizione *cond*
- Le condizioni di selezione possono prevedere confronti fra attributi e fra attributi e costanti e possono essere costruite combinando condizioni più semplici con i connettivi logici $\vee$ (or), $\wedge$ (and) e $\neg$ (not)

IMPIEGATI(Cognome,Nome,Età,Stipendio)

$\sigma_{Età<30 \text{ AND } Stipendio>4.000.000}(IMPIEGATI)$

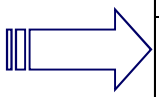
⇩

Vengono selezionati i record relativi ad impiegati con età minore di 30 anni e stipendio superiore a 4 milioni

Esempi di selezione

Cognome	Nome	Età	Stipendio
Rossi	Alberto	25	2.000.000
Verdi	Luca	40	4.500.000
Billi	Paolo	28	4.100.000
Luti	Anna	29	5.000.000

IMPIEGATI

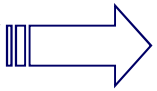


Cognome	Nome	Età	Stipendio
Billi	Paolo	28	4.100.000
Luti	Anna	29	5.000.000

$\sigma_{\text{Età}<30 \text{ AND } \text{Stipendio}>4.000.000}(\text{IMPIEGATI})$

Cognome	Nome	Città di nascita	Residenza
Rossi	Alberto	Firenze	Firenze
Verdi	Luca	Siena	Pisa
Billi	Paolo	Pisa	Lucca
Luti	Anna	Prato	Prato

CITTADINI



Cognome	Nome	Città di nascita	Residenza
Rossi	Alberto	Firenze	Firenze
Luti	Anna	Prato	Prato

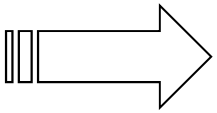
$\sigma_{\text{Città di Nascita}=\text{Residenza}}(\text{CITTADINI})$

Proiezione

- Dati una relazione R(X) e un sottoinsieme Y degli attributi in X, la proiezione di R su Y $\pi_Y(R)$ è l'insieme delle tuple su Y ottenute dalle tuple di R considerando solo i valori su Y

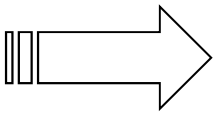
Cognome	Nome	Cat	Stipendio
Rossi	Alberto	Ric	2.000.000
Verdi	Luca	PA	3.500.000
Billi	Paolo	PA	3.500.000
Luti	Anna	PO	5.000.000

DOCENTI



Cognome	Nome
Rossi	Alberto
Verdi	Luca
Billi	Paolo
Luti	Anna

$\pi_{Cognome,Nome}(DOCENTI)$



Cat	Stipendio
Ric	2.000.000
PA	3.500.000
PO	5.000.000

$\pi_{Cat,Stipendio}(DOCENTI)$

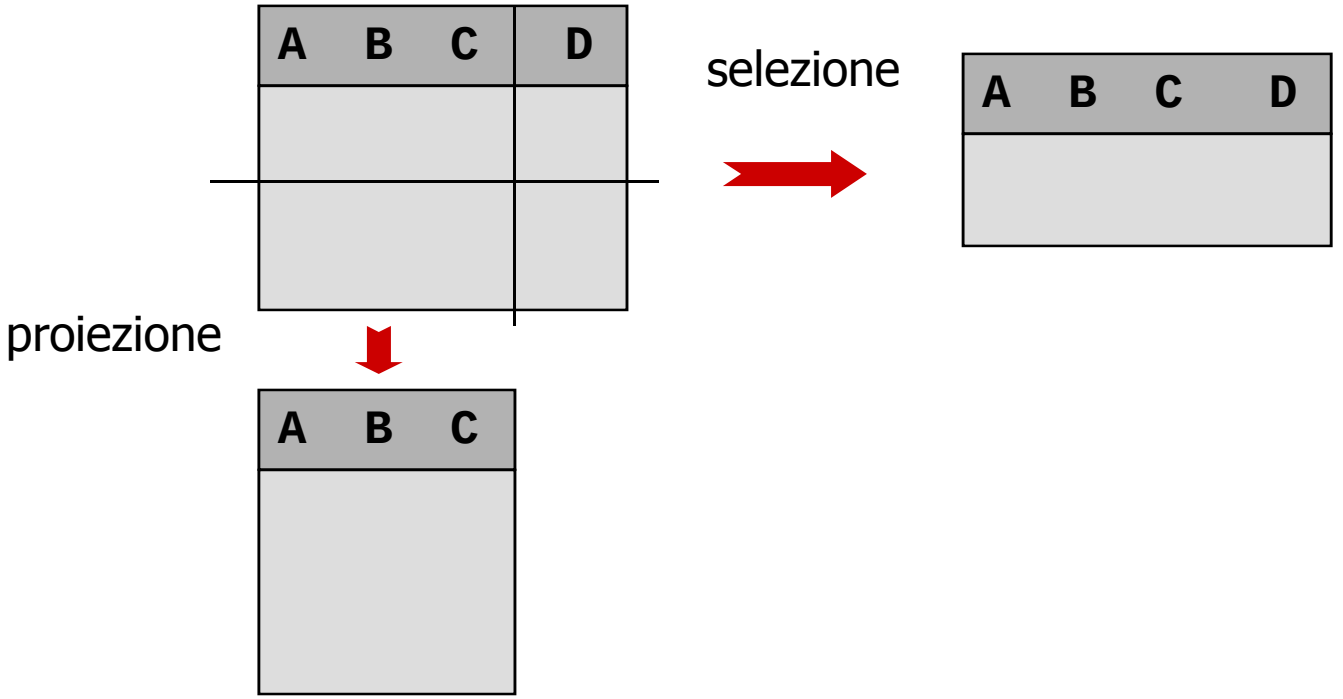
Cardinalità della proiezione

- una proiezione
  - contiene al più tante tuple quante l'operando
  - può contenerne di meno (si eliminano i duplicati)

- se X è una superchiave di R, allora $\pi_X(R)$ contiene esattamente tante tuple quante R

Selezione e proiezione

- Sono operatori "ortogonali"
  - selezione - decomposizione orizzontale
  - proiezione - decomposizione verticale



Combinazione di σ e π

- Combinando selezione e proiezione si possono estrarre informazioni da una relazione

Cognome	Nome	Cat	Stipendio
Rossi	Alberto	Ric	2.000.000
Verdi	Luca	PA	3.500.000
Billi	Paolo	PA	3.500.000
Luti	Anna	PO	5.000.000

Docenti

seleziona Nome e Cognome dei Professori Associati

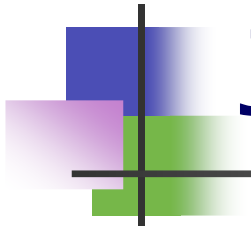
$\pi_{Cognome,Nome}(\sigma_{Cat='PA'}(docenti))$

Cognome	Nome
Verdi	Luca
Billi	Paolo

Professori Associati

Limitazione di σ e π

- Combinando selezione e proiezione, possiamo estrarre informazioni da una sola relazione
- Sono infatti operatori con un solo arogomento
- non possiamo però correlare informazioni presenti in relazioni diverse
- Si introduce l'operatore di JOIN che permette di correlare dati in relazioni diverse



Join

- E’ l’operatore che permette di correlare dati contenuti in relazioni diverse confrontando i valori contenuti in esse
- Utilizza il fatto che il modello relazionale è basato su valori
- Esistono due varianti principali dell’operatore
  - Join naturale
  - theta-join

Join naturale

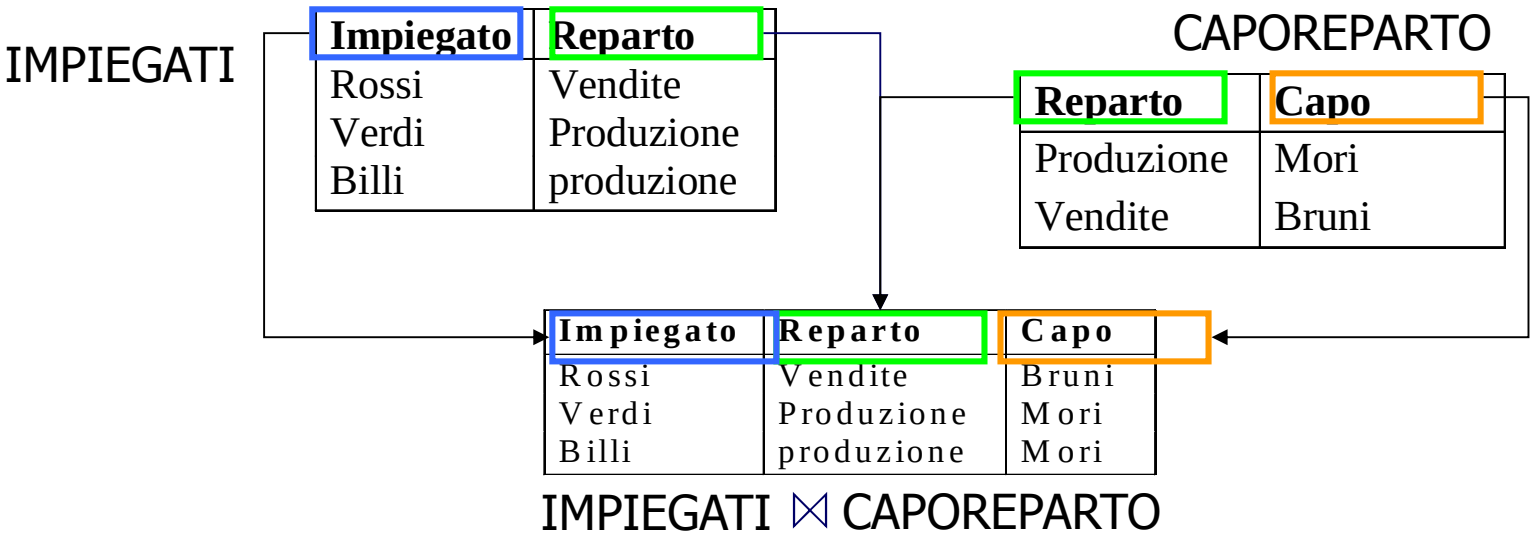
- E' un operatore che correla dati in relazioni diverse sulla base di

  valori uguali in attributi con lo stesso nome

- La relazione risultante
  - ha per attributi l'unione degli attributi delle relazioni di partenza
  - le sue tuple sono ottenute combinando le tuple delle relazioni con valori uguali sugli attributi comuni

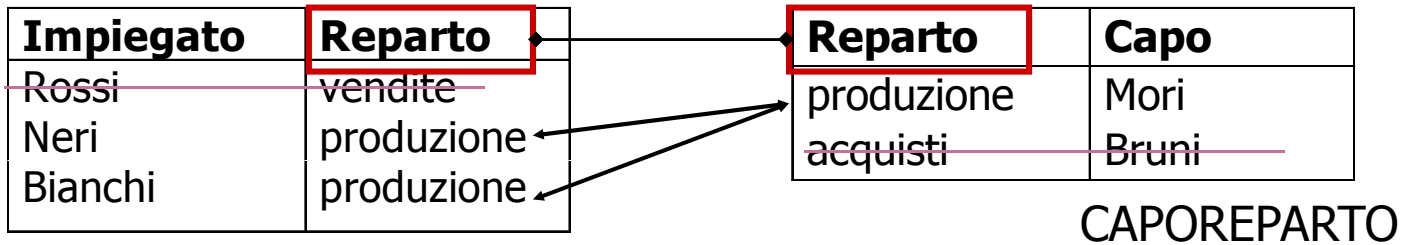
$$r_1(X_1) \bowtie r_2(X_2) = \{t \text{ su } X_1 \cup X_2 \mid t[X_1] \in r_1 \text{ e } t[X_2] \in r_2\}$$

Esempio di join naturale



- In questo caso si ha un join completo
  - $\forall\ t_1 \in r_1\ \exists\ t \in r_1 \bowtie r_2$ tale che $t[X_1]=t_1$
  - $\forall\ t_2 \in r_2\ \exists\ t \in r_1 \bowtie r_2$ tale che $t[X_2]=t_2$

Join con tuple "pendenti"



Impiegato	Reparto
Rossi	vendite
Neri	produzione
Bianchi	produzione

DIPENDENTI

Reparto	Capo
produzione	Mori
acquisti	Bruni

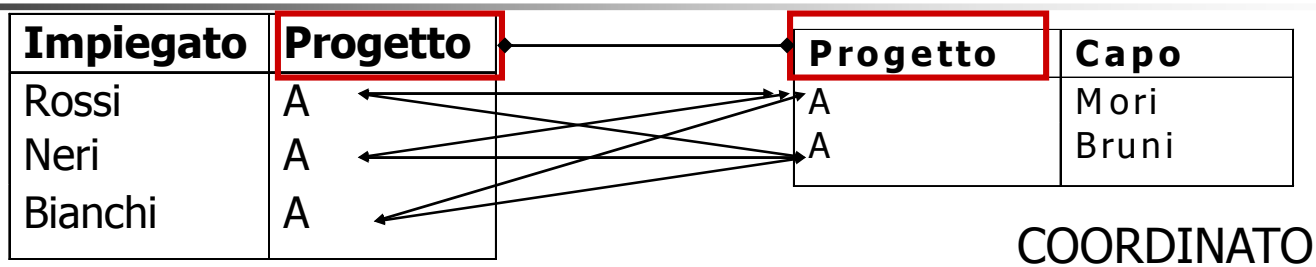
CAPOREPARTO

Impiegato	Reparto	Capo
Neri	produzione	Mori
Bianchi	produzione	Mori

CAPOREPARTO ⋈ DIPENDENTI

La prima tupla della relazione DIPENDENTI e la seconda della relazione CAPOREPARTO non generano nessuna tupla nel join

Un esempio di join



Impiegato	Progetto
Rossi	A
Neri	A
Bianchi	A

RICERCATORI

Progetto	Capo
A	Mori
A	Bruni

COORDINATORI

Impiegato	Progetto	Capo
Rossi	A	Mori
Rossi	A	Bruni
Neri	A	Mori
Neri	A	Bruni
Bianchi	A	Mori
Bianchi	A	Bruni

RICERCATORI ⋈ COORDINATORI

Ciascuna tupla della prima relazione è combinabile con tutte le tuple dell'altra. Il risultato contiene un numero di tuple pari al prodotto dei numeri di tuple nelle due relazioni originarie.

Infrazioni e auto

Codice	Data	Agente	Art	Prov	Numero
343535	12/05/99	567	44	FI	B04546
343344	14/05/99	456	22	SI	345678
342233	22/06/99	456	44	FI	B04546
332112	27/07/99	456	53	PI	768930
334545	07/08/99	567	44	GR	546788

INFRAZIONI

è una chiave

AUTOVEICOLI

Prov	Numero	Proprietario	Indirizzo
FI	B04546	Paolo Rossi	Via Bixio 5 Prato
SI	345678	Piero Berti	Via Abete 6 Siena
PI	768930	Luca Bianchi	Piazza Po 16 Pisa
GR	546788	Anna Verdi	Viale Europa 4 Firenze

Codice	Data	Agente	Art	Prov	Numero	Proprietario	Indirizzo
343535	12/05/99	567	44	FI	B04546	Paolo Rossi	Via Bixio 5 Prato
343344	14/05/99	456	22	SI	345678	Piero Berti	Via Abete 6 Siena
342233	22/06/99	456	44	FI	B04546	Paolo Rossi	Via Bixio 5 Prato
332112	27/07/99	456	53	PI	768930	Luca Bianchi	Piazza Po 16 Pisa
334545	07/08/99	567	44	GR	546788	Anna Verdi	Viale Europa 4 Firenze

INFRAZIONI $\bowtie$ AUTOVEICOLI

Il join è basato su una chiave della relazione AUTOVEICOLI, quindi ogni tupla di INFRAZIONI si combina con una sola tupla di AUTOVEICOLI

Cardinalità del join

- In generale

$$0 \leq |r_1 \bowtie r_2| \leq |r_1| \times |r_2|$$

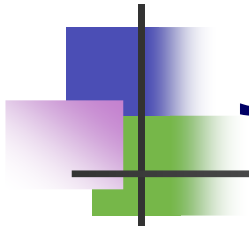
- Se il join è completo $|r_1 \bowtie r_2| \geq \max(|r_1|,|r_2|)$
- Se $X_1 \cap X_2$ contiene una chiave di r_2 allora

$$|r_1 \bowtie r_2| \leq |r_1|$$

Ogni tupla di r_1 si combina al più con una tupla di r_2

- Se $X_1 \cap X_2$ contiene una chiave di r_2 e sussiste il vincolo di integrità referenziale fra $X_1 \cap X_2$ in r_1 e la chiave di r_2, allora

$$|r_1 \bowtie r_2| = |r_1|$$



Join esterno

- Il join tralascia le tuple di una relazione che non hanno corrispondenza nell'altra relazione
- Il join esterno (outer join) estende, con valori nulli, le tuple che verrebbero tagliate fuori da un join (interno)
- esiste in tre versioni:
  - sinistro - mantiene tutte le tuple del primo operando estendendole con valori nulli se necessario
  - destro - estende le tuple del secondo operando
  - completo - estende le tuple di entrambi gli operandi
- E' previsto da SQL

Esempi di join esterno

Impiegato	Reparto
Rossi	vendite
Neri	produzione
Bianchi	produzione

Dipendenti

Reparto	Capo
produzione	Mori
acquisti	Bruni

Caporeparto

Dipendenti ⟕ Caporeparto (LEFT)

Impiegato	Reparto	Capo
Rossi	vendite	NULL
Neri	produzione	Mori
Bianchi	produzione	Mori

Dipendenti ⟖ Caporeparto (RIGHT)

Impiegato	Reparto	Capo
Neri	produzione	Mori
Bianchi	produzione	Mori
NULL	acquisti	Bruni

Dipendenti ⟗ Caporeparto (FULL)

Impiegato	Reparto	Capo
Rossi	vendite	NULL
Neri	produzione	Mori
Bianchi	produzione	Mori
NULL	acquisti	Bruni

Proprietà del Join naturale

- Il join è commutativo e associativo

$$r_1 \bowtie r_2 = r_2 \bowtie r_1$$

$$r_1 \bowtie (r_2 \bowtie r_3) = (r_1 \bowtie r_2) \bowtie r_3$$

- Se r_1 e r_2 sono definite sullo stesso insieme di attributi, il join coincide con l'intersezione

$$r_1(X) \bowtie r_2(X) = r_1(X) \cap r_2(X)$$

- Se r_1 e r_2 sono definite su insiemi di attributi disgiunti ($X_1 \cap X_2 = \emptyset$), il join diventa un "prodotto cartesiano"

  (concatenazione di tutte le tuple in r_1 e con tutte le tuple in r_2)

Join e *prodotto cartesiano*

Impiegato	Progetto
Rossi	A
Neri	A
Neri	B

RICERCATORI

Codice	Nome
A	Venere
B	Marte

PROGETTI

RICERCATORI ⋈ PROGETTI

Impiegato	Progetto	Codice	Nome
Rossi	A	A	Venere
Neri	A	A	Venere
Neri	B	A	Venere
Rossi	A	B	Marte
Neri	A	B	Marte
Neri	B	B	Marte

Il theta-join

- E' un operatore derivato. Corrisponde ad effettuare il prodotto cartesiano delle due relazioni seguito da una selezione basata su una condizione F.
- Se r_1 e r_2 non hanno attributi in comune

$$r_1 \bowtie_F r_2 = \sigma_F(r_1 \bowtie r_2)$$

Selezione delle tuple che soddisfano F

Tutte le possibili combinazioni delle tuple

- E' importante dal punto di vista pratico perché è utilizzato nei sistemi di basi di dati esistenti

Perchè theta-join?

- La condizione F è spesso una espressione che lega con operatori logici (AND, OR, NOT) atomi di confronto

$$A_1 \vartheta\ A_2$$

dove ϑ è uno degli operatori di confronto (=, >, <, …)

- se l'operatore è sempre l'uguaglianza (=) allora si parla di equi-join
- Il join naturale può essere ottenuto combinando ridenominazione equi-join e proiezione

$$\pi_{ABC}(r_1 \bowtie_{B=B'} (\rho_{B' \leftarrow B}(r_2)))$$

con $r_1(AB)$ e $r_2(BC)$

Un esempio di theta-join

Impiegato	Progetto
Rossi	A
Neri	A
Neri	B

RICERCATORI

Codice	Nome
A	Venere
B	Marte

PROGETTI

RICERCATORI

$\bowtie$

PROGETTI

Impiegato	Progetto	Codice	Nome
Rossi	A	A	Venere
Neri	A	A	Venere
Neri	B	A	Venere
Rossi	A	B	Marte
Neri	A	B	Marte
Neri	B	B	Marte

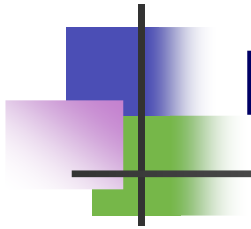
Impiegato	Progetto	Codice	Nome
Rossi	A	A	Venere
Neri	A	A	Venere
Neri	B	B	Marte

RICERCATORI

$\bowtie_{\text{Progetto=Codice}}$

PROGETTI

Interrogazioni in algebra relazionale



- Un'interrogazione a una base di dati produce una serie di record che soddisfano i criteri richiesti
- E' una funzione che applicata ad un'istanza di basi di dati produce una relazione ovvero dei dati organizzati come tuple di una relazione
- Le interrogazioni si possono rappresentare con gli operatori dell'algebra relazionale che forniscono una procedura per calcolare il risultato

Un esempio di interrogazione

Matricola	**Nome**	**Età**	**Stipendio**
101	Marco Rossi	23	1.500
103	Paolo Bianchi	34	2.680
105	Anna Falchi	33	1.700
110	Gaia Belli	36	2.500
134	Luca Forti	27	2.500
145	Sonia Melli	23	1.500
149	Mario Mori	33	1.800
153	Bruno Bruni	35	1.500
155	Filippo Mei	30	2.500

IMPIEGATI

Capo	**Impiegato**
103	101
103	105
103	145
110	103
110	149
134	153

SUPERVISIONE

Trovare nome e stipendio dei capi degli impiegati che guadagnano più di 1.700

I passi dell'interrogazione [1]

- Si selezionano gli impiegati che guadagnano più di 1.700

$$\sigma_{\text{Stipendio}>1.700}(\text{IMPIEGATI})$$

Matricola	**Nome**	**Età**	**Stipendio**
103	Paolo Bianchi	34	2.680
110	Gaia Belli	36	2.500
134	Luca Forti	27	2.500
149	Mario Mori	33	1.800
155	Filippo Mei	30	2.500

- Si associano gli impiegati trovati con i rispettivi capi

$$\text{SUPERVISIONE} \bowtie_{\text{Impiegato=Matricola}} \sigma_{\text{Stipendio}>1.700}(\text{IMPIEGATI})$$

Matricola	**Nome**	**Età**	**Stipendio**	**Capo**	**Impiegato**
103	Paolo Bianchi	34	2.680	110	103
149	Mario Mori	33	1.800	110	149

I passi dell'interrogazione [2]

- Si estrae la matricola dei capi

$$\pi_{Capo}\ (SUPERVISIONE \bowtie_{Impiegato=Matricola} \sigma_{Stipendio>1.700}(IMPIEGATI))$$

Capo
110

- Si associa la matricola con la relazione IMPIEGATI per ottenere le informazioni sui capi e poi si proietta sugli attributi richiesti (Nome, Stipendio)

$$\pi_{Nome,Stipendio}\ (IMPIEGATI \bowtie_{Matricola=Capo} (\pi_{Capo}\ (SUPERVISIONE \bowtie_{Impiegato=Matricola} \sigma_{Stipendio>1.700}(IMPIEGATI))))$$

Nome	Stipendio
Gaia Belli	2.500

Un'altra interrogazione [1]

Trovare gli impiegati che guadagnano più del rispettivo capo, mostrando matricola, nome e stipendio di entrambi

- Si associa la matricola del capo ad ogni impiegato

$$\text{SUPERVISIONE} \bowtie_{\text{Impiegato=Matricola}} \text{IMPIEGATI}$$

- Si associa la matricola del capo con la riga corrispondente della relazione IMPIEGATI (ridenominando i campi)

$$\rho_{\text{MatrCapo,NomeCapo,StipCapo,EtàCapo} \leftarrow \text{Matricola,Nome,Stipendio,Età}}(\text{IMPIEGATI}) \bowtie_{\text{MatrCapo=Capo}} (\text{SUPERVISIONE} \bowtie_{\text{Impiegato=Matricola}} \text{IMPIEGATI})$$

Un'altra interrogazione [2]

IMPIEGATI

Nome	Età	Stipendio	Matricola
Marco Rossi	23	1.500	101
Paolo Bianchi	34	2.680	103
Anna Falchi	33	1.700	105
Gaia Belli	36	2.500	110
Luca Forti	27	2.500	134
Sonia Melli	23	1.500	145
Mario Mori	33	1.800	149
Bruno Bruni	35	1.500	153
Filippo Mei	30	2.500	155

SUPERVISIONE

Impiegato	Capo
101	103
105	103
145	103
103	110
149	110
153	134

Matricola	Nome	Età	Stipendio	Impiegato	Capo
101	Marco Rossi	23	1.500	101	103
103	Paolo Bianchi	34	2.680	103	110
105	Anna Falchi	33	1.700	105	103
145	Sonia Melli	23	1.500	145	103
149	Mario Mori	33	1.800	149	110
153	Bruno Bruni	35	1.500	153	134

SUPERVISIONE $\bowtie_{\text{Impiegato=Matricola}}$ IMPIEGATI

Un'altra interrogazione [3]

$\rho_{MatrCapo,NomeCapo,StipCapo,EtàCapo \leftarrow Matricola,Nome,Stipendio,Età}(IMPIEGATI)$

NomeCapo	EtàCapo	StipCapo	MatrCapo
Marco Rossi	23	1.500	101
Paolo Bianchi	34	2.680	103
Anna Falchi	33	1.700	105
Gaia Belli	36	2.500	110
Luca Forti	27	2.500	134
Sonia Melli	23	1.500	145
Mario Mori	33	1.800	149
Bruno Bruni	35	1.500	153
Filippo Mei	30	2.500	155

Capo	Matricola	Nome	Età	Stipendio	Impiegato
103	101	Marco Rossi	23	1.500	101
110	103	Paolo Bianchi	34	2.680	103
103	105	Anna Falchi	33	1.700	105
103	145	Sonia Melli	23	1.500	145
110	149	Mario Mori	33	1.800	149
134	153	Bruno Bruni	35	1.500	153

SUPERVISIONE $\bowtie_{Impiegato=Matricola}$ IMPIEGATI

NomeCapo	EtàCapo	StipCapo	MatrCapo	Capo	Matricola	Nome	Età	Stipendio	Impiegato
Paolo Bianchi	34	2.680	103	103	101	Marco Rossi	23	1.500	101
Gaia Belli	36	2.500	110	110	103	Paolo Bianchi	34	2.680	103
Paolo Bianchi	34	2.680	103	103	105	Anna Falchi	33	1.700	105
Paolo Bianchi	34	2.680	103	103	145	Sonia Melli	23	1.500	145
Gaia Belli	36	2.500	110	110	149	Mario Mori	33	1.800	149
Luca Forti	27	2.500	134	134	153	Bruno Bruni	35	1.500	153

$\rho_{MatrCapo,NomeCapo,StipCapo,EtàCapo \leftarrow Matricola,Nome,Stipendio,Età}(IMPIEGATI)$

$\bowtie_{MatrCapo=Capo}$ (SUPERVISIONE $\bowtie_{Impiegato=Matricola}$ IMPIEGATI)

Un'altra interrogazione [4]

Trovare gli impiegati che guadagnano più del rispettivo capo, mostrando matricola, nome e stipendio di entrambi

- Si selezionano le righe che soddisfano il criterio Stipendio>StipCapo

$$\sigma_{\text{Stipendio}>\text{StipCapo}}(\rho_{\text{MatrCapo,NomeCapo,StipCapo,EtàCapo}\leftarrow\text{Matricola,Nome,Stipendio,Età}}(\text{IMPIEGATI}) \bowtie_{\text{MatrCapo=Capo}} (\text{SUPERVISIONE} \bowtie_{\text{Impiegato=Matricola}} \text{IMPIEGATI}))$$

- Si estraggono gli attributi che interessano con la proiezione

$$\pi_{\text{Matricola,Nome,Stipendio,MatrCapo,NomeCapo,StipCapo}}()$$

Un'altra interrogazione [5]

$\rho_{MatrCapo,NomeCapo,StipCapo,EtàCapo \leftarrow Matricola,Nome,Stipendio,Età}(IMPIEGATI)$

$\bowtie_{MatrCapo=Capo}$ (SUPERVISIONE $\bowtie_{Impiegato=Matricola}$ IMPIEGATI)

NomeCapo	EtàCapo	StipCapo	MatrCapo	Capo	Matricola	Nome	Età	Stipendio	Impiegato
Paolo Bianchi	34	2.680	103	103	101	Marco Rossi	23	1.500	101
Gaia Belli	36	2.500	110	110	103	Paolo Bianchi	34	2.680	103
Paolo Bianchi	34	2.680	103	103	105	Anna Falchi	33	1.700	105
Paolo Bianchi	34	2.680	103	103	145	Sonia Melli	23	1.500	145
Gaia Belli	36	2.500	110	110	149	Mario Mori	33	1.800	149
Luca Forti	27	2.500	134	134	153	Bruno Bruni	35	1.500	153

$\sigma_{Stipendio>StipCapo}()$

NomeCapo	EtàCapo	StipCapo	MatrCapo	Capo	Matricola	Nome	Età	Stipendio	Impiegato
Gaia Belli	36	2.500	110	110	103	Paolo Bianchi	34	2.680	103

Equivalenza di espressioni

- Due espressioni sono equivalenti se producono lo stesso risultato
  - $E_1 \equiv_R E_2$ se $E_1(r) = E_2(r)$ per ogni istanza r della basi di dati con schema R
  - $E_1 \equiv E_2$ se $E_1 \equiv_R E_2$ per ogni schema R

- Le trasformazioni di equivalenza sono utilizzate per ottimizzare le operazioni da effettuare per produrre il risultato di una interrogazione
  - Ottimizzazione della dimensione dei risultati intermedi

Trasformazioni di equivalenza 1

- Atomizzazione delle selezioni

$$\sigma_{F1 \wedge F2} \equiv \sigma_{F1}(\sigma_{F2}(E))$$

- Idempotenza delle proiezioni

$$\pi_X(E) \equiv \pi_X(\pi_{XY}(E))$$

- Anticipazione della selezione rispetto al join

$$\sigma_F(E_1 \bowtie E_2) \equiv E_1 \bowtie \sigma_F(E_2)$$

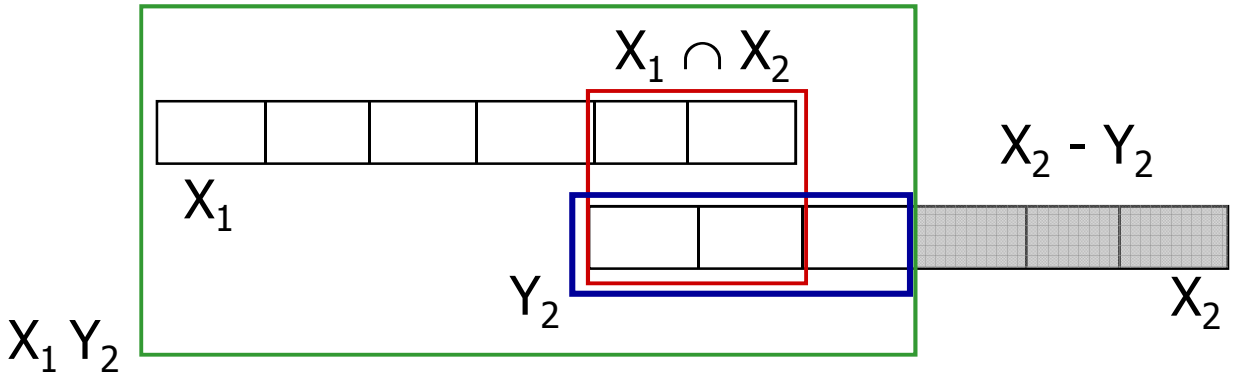
se la condizione F fa riferimento solo ad attributi di E_2

Trasformazioni di equivalenza 2

- Anticipazione della proiezione rispetto al join

$$\pi_{X1Y2}(E_1 \bowtie E_2) \equiv E_1 \bowtie \pi_{Y2}(E_2)$$

  - E_1 è definita sugli attributi X_1 e E_2 su X_2
  - $Y_2 \subseteq X_2$
  - $(X_1 \cap X_2) \subseteq Y_2$ - gli attributi coinvolti nel join sono tutti in Y_2



Trasformazioni di equivalenza 3

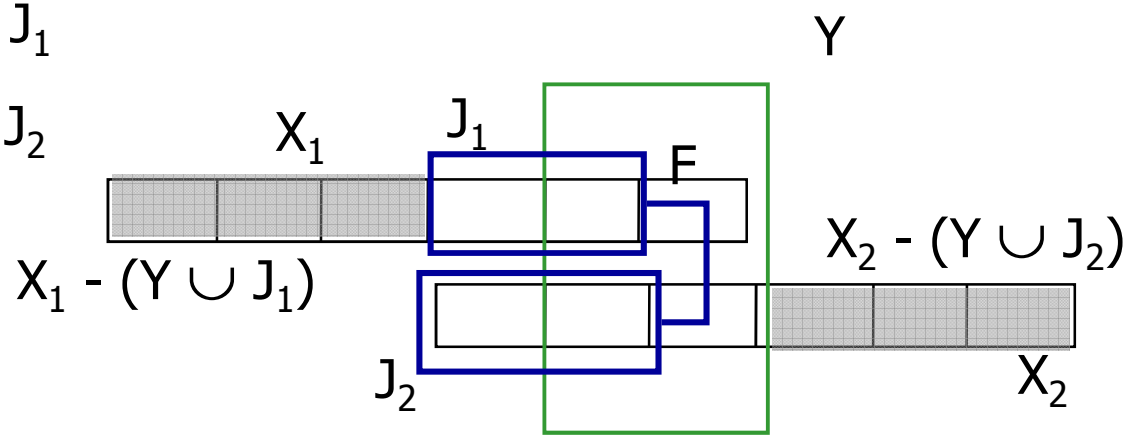
- Eliminazione degli attributi "superflui"

  Si possono eliminare subito da ciascuna relazione gli attributi che non compaiono nel risultato finale e non sono coinvolti nel join

$$\pi_Y(E_1 \bowtie_F E_2) \equiv \pi_Y(\pi_{Y1}(E_1) \bowtie_F \pi_{Y2}(E_2))$$

  Detti $J_1 \subseteq X_1$ e $J_2 \subseteq X_2$ gli attributi coinvolti nel join

  - $Y_1=(X_1 \cap Y) \cup J_1$
  - $Y_2=(X_2 \cap Y) \cup J_2$



Trasformazioni di equivalenza 4

- Selezione in un prodotto cartesiano

$$\sigma_F(E_1 \bowtie E_2) \equiv E_1 \bowtie_F E_2$$

- Distributività della selezione

$$\sigma_F(E_1 \cup E_2) \equiv \sigma_F(E_1) \cup \sigma_F(E_2)$$

$$\sigma_F(E_1 - E_2) \equiv \sigma_F(E_1) - \sigma_F(E_2)$$

- Distributività della proiezione

$$\pi_X(E_1 \cup E_2) \equiv \pi_X(E_1) \cup \pi_X(E_2)$$

Trasformazioni di equivalenza 5

- Distributività del join

$$E \bowtie (E_1 \cup E_2) \equiv (E \bowtie E_1) \cup (E \bowtie E_2)$$

- Selezioni con espressioni composte

$$\sigma_{F1 \vee F2}(R) \equiv \sigma_{F1}(R) \cup \sigma_{F2}(R)$$

$$\sigma_{F1 \wedge F2}(R) \equiv \sigma_{F1}(R) \cap \sigma_{F2}(R) \equiv \sigma_{F1}(R) \bowtie \sigma_{F2}(R)$$

$$\sigma_{F1 \wedge \neg F2}(R) \equiv \sigma_{F1}(R) - \sigma_{F2}(R)$$

Algebra con valori nulli

- La presenza di valori nulli crea problemi nel calcolo dei predicati applicati sugli attributi
- Una proposizione può essere vera (V) , falsa (F) o sconosciuta (U) - logica a 3 valori

Età>33

??

Matricola	Nome	Età	
103	Paolo Bianchi	34	V
110	Gaia Belli	36	V
134	Luca Forti	NULL	U
149	Mario Mori	33	F
155	Filippo Mei	30	F

Impiegati

Matricola	Nome	Età
103	Paolo Bianchi	34
110	Gaia Belli	36

$\sigma_{Età>33}(Impiegati)$

Logica a 3 valori

- Occorre estendere gli operatori AND, OR e NOT alla logica a 3 valori

NOT	
F	V
U	U
V	F

AND	**F**	**U**	**V**
F	F	F	F
U	F	U	U
V	F	U	V

OR	**F**	**U**	**V**
F	F	U	V
U	U	U	V
V	V	V	V

- Tuttavia ci sono dei problemi....

Un risultato non desiderabile

??

Matricola	Nome	Età
103	Paolo Bianchi	34
110	Gaia Belli	36
134	Luca Forti	NULL
149	Mario Mori	33
155	Filippo Mei	30

Impiegati

Matricola	Nome	Età
103	Paolo Bianchi	34
110	Gaia Belli	36
149	Mario Mori	33
155	Filippo Mei	30

$\sigma_{Età>33}(Impiegati) \cup \sigma_{Età\leq 33}(Impiegati)$

Matricola	Nome	Età
103	Paolo Bianchi	34
110	Gaia Belli	36
149	Mario Mori	33
155	Filippo Mei	30

$\sigma_{Età>33 \vee Età\leq 33}(Impiegati)$

$\neq$ Impiegati

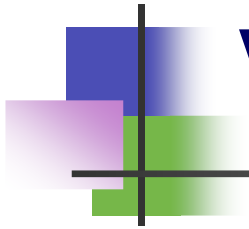
Due nuove condizioni atomiche

- Si introducono le condizioni atomiche
  - A IS NULL

    vero per una tupla t se t[A]=NULL
  - A IS NOT NULL

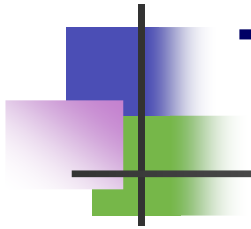
    vero per una tupla t se t[A]≠NULL
- Questa soluzione è disponibile in SQL
- Il problema scompare....

$$\sigma_{\text{Età}>33 \vee \text{Età}\leq 33 \vee \text{Età IS NULL}}(\text{Impiegati}) = \text{Impiegati}$$



Viste

- Le relazioni derivate permettono rappresentazioni diverse per gli stessi dati
- Si fa riferimento allo schema esterno dell'architettura dei DBMS
  - Relazioni derivate
    Relazioni il cui contenuto dipende dal contenuto di altre relazioni
  - Relazioni di base
    Relazioni il cui contenuto è autonomo
- Le relazioni derivate possono dipendere anche da altre relazioni derivate a patto che non ci sia circolarità



Tipologie di viste

- Viste materializzate

  Sono relazioni derivate effettivamente memorizzate nella base di dati
  - immediatamente disponibili per le interrogazioni
  - ridondanti
  - appesantiscono gli aggiornamenti
  - non sono supportate dai DBMS
- Relazioni virtuali (viste)

  Relazioni definite per mezzo di funzioni (espressioni del linguaggio di interrogazione)

  Vengono ricalcolate all'occorrenza

Viste: un esempio

Impiegato	Reparto
Paolo Bianchi	A
Gaia Belli	B
Mario Mori	B
Filippo Mei	C

Afferenza

Reparto	Capo
A	Gino Bruni
B	Cesare Teo
C	Anna Livio

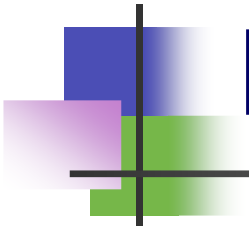
Direzione

$$\text{Supervisione} = \pi_{\text{Impiegato,Capo}}(\text{Afferenza} \bowtie \text{Direzione})$$

Le interrogazioni sono effettuate sostituendo la definizione della vista nell'espressione che rappresenta l'interrogazione

$$\sigma_{\text{Capo='Anna Livio'}}(\text{Supervisione}) =$$

$$\sigma_{\text{Capo='Anna Livio'}}\ (\pi_{\text{Impiegato,Capo}}(\text{Afferenza} \bowtie \text{Direzione}))$$



Motivazioni delle viste

- Un utente *vede* solo le parti rilevanti della base di dati
- L'utente vede solo ciò che è autorizzato a vedere
- Utilizzando le viste si può semplificare la scrittura di interrogazioni (espressioni complesse e sottoespressioni ripetute)
- Un vista può permettere l'utilizzo di programmi esistenti su schemi ristrutturati
- L'utilizzo di viste non influisce sull'efficienza delle interrogazioni

Programmazione e viste

Trovare gli impiegati che hanno lo stesso capo di Mario Mori

- Senza vista

$$\pi_{\text{Impiegato}}((\text{Afferenza} \bowtie \text{Direzione}) \bowtie \rho_{\text{ImpR,RepR} \leftarrow \text{Impiegato,Reparto}}(\sigma_{\text{Impiegato='Mario Mori'}}(\text{Afferenza} \bowtie \text{Direzione})))$$

capo di Mario Mori

- Con la vista Supervisione = $\pi_{\text{Impiegato,Capo}}(\text{Afferenza} \bowtie \text{Direzione})$ diviene

$$\pi_{\text{Impiegato}}(\text{Supervisione} \bowtie \rho_{\text{ImpR} \leftarrow \text{Impiegato}}(\sigma_{\text{Impiegato='Mario Mori'}}(\text{Supervisione})))$$

Viste e aggiornamenti

- "Aggiornare una vista"
  - modificare le relazioni di base in modo che la vista, "ricalcolata" rispecchi l'aggiornamento
- L'aggiornamento sulle relazioni di base corrispondente a quello specificato sulla vista deve essere univoco
- In generale però non è univoco....

  Se aggiorno Supervisione inserendo una coppia (Impiegato,Capo) come posso aggiornare Afferenza e Direzione se non specifico il reparto? Dipende dai dati già presenti...
- Ben pochi aggiornamenti sono ammissibili sulle viste

Calcolo relazionale

- Fa riferimento ad una famiglia di linguaggi dichiarativi, basati sul calcolo dei predicati del primo ordine
- Specifica le proprietà del risultato piuttosto che indicare la procedura per calcolarlo
- Ne esistono versioni con potenzialità espressive diverse
  - calcolo relazionale su domini
  - calcolo su tuple con dichiarazioni di range
    (In questo corso vedremo il secondo, che è la base di molti costrutti del linguaggio SQL)

Calcolo su tuple con dichiarazioni di range

- Le espressioni del calcolo su tuple con dichiarazioni di range hanno la forma

  { target list | range list | f }

  - La target list è la lista degli obiettivi dell'interrogazione
  - La range list elenca le variabili libere della formula f
  - f è una formula con valore vero o falso

Esempio

- Si considera il seguente schema

  Studenti(Matricola,Nome,DataNascita,CittaResidenza)

  Citta(CodiceCitta,Nome,NumeroAbitanti)

Q: *Trovare matricola e nome degli studenti che abitano in città con più di 1000 abitanti*

Target list — Range list

{ s.Matricola ,s.Nome | c(Citta), s(Studenti) |

s.CittaResidenza = c.CodiceCitta $\wedge$ c.NumeroAbitanti >1000

formula

Target list

- La target list indica gli attributi della tabella che si intende ricavare
- è composta da elementi del tipo
  - Y: *x*.Z
    - x è una variabile che rappresenta una tupla il cui range è una relazione definita su un insieme di attributi X
    - Z $\subseteq$ X
    - Y è una lista di attributi e |Y|=|Z|
  - *x*.Z
    abbreviazione per Z: *x*.Z
  - *x*.*
    abbreviazione per X: *x*.X (si prelevano tutti gli attributi della tupla *x*)

Target list: esempio

- *i* è una variabile associata ad una tupla della relazione Impiegati(Matricola,Nome,Età,Stipendio)
  - *i*.*
    definisce come risultato una tupla che contiene tutti gli attributi di Impiegati, ovvero {Matricola,Nome,Età,Stipendio}
  - *i*.(Nome,Età)
    rappresenta una tupla definita sulla relazione con i due attributi {Nome,Età}
  - NomeCapo,StipCapo: *i*.(Nome,Stipendio)
    definisce una tupla con i due attributi {NomeCapo,StipCapo}

Range list

- Elenca le variabili libere della formula f con i relativi campi di variabilità (relazione di appartenenza R)

$$x(R)$$

Ad esempio...

- *i* (Impiegati)
  i può assumere i valori delle tuple contenute nella relazione Impiegati
- *i* (Impiegati), *s* (Supervisione)
- *i* (Impiegati), *s* (Supervisione), *i1* (Impiegati)

La formula f

- Formule atomiche
  - $x.A \vartheta c$ - confronto del valore dell'attributo A della tupla x con la costante c
  - $x_1.A_1 \vartheta x_2.A_2$ - confronto fra il valore dell'attributo A_1 della tupla x_1 e quello dell'attributo A_2 della tupla x_2
  - ad es.
    - S.nome='pippo'
    - E.matricola=s.matricola
- Operatori logici
  - Se f_1 e f_2 sono formule allora sono formule
    - $f_1 \vee f_2$ (or)
    - $f_1 \wedge f_2$ (and)
    - $\neg f_1$ (not)
    - ad es. S.nome='pippo' $\wedge$ e.matricola =s.matricola

Quantificatori

- Quantificatore esistenziale
  - Data una formula f, una variabile x e una relazione R si definisce la formula

    $\exists$ x(R) f
  - La formula è vera se esiste almeno una tupla di R che sostituito alla variabile x rende vera f
- Quantificatore universale
  - Data una formula f e una variabile x e una relazione R si definisce la formula

    $\forall$ x(R) f
  - La formula è vera se per ogni possibile tupla x di R la formula f è vera

Relazione fra ∀ e ∃

- Valgono le leggi di De Morgan

$$\forall x \neg P \equiv \neg \exists x \; P$$
$$\neg \forall x \; P \equiv \exists x \neg P$$
$$\forall x \; P \equiv \neg \exists x \neg P$$
$$\exists x \; P \equiv \neg \forall x \neg P$$

Ad esempio.....

Non esiste uno studente nato il '1/1/1900'

¬ ∃ s(Studenti) s.dataNascita= '1/1/1900'

Ogni studente non è nato il '1/1/1900'

∀ s(Studenti) ¬ s.dataNascita= '1/1/1900'

- Quindi... uno solo dei due operatori sarebbe sufficiente...

Esempio [1]

- Si considera il seguente schema

Impiegati(Matricola,Nome,età,Stipendio)

Supervisione(Capo,Impiegato)

Q: *Trovare matricola, nome, età e stipendio degli impiegati che guadagnano più di 1.700*

{ i.* | i(Impiegati) | i.Stipendio > 1700 }

Target list | Range list | condizione

Q: *Trovare matricola, nome, età di tutti gli impiegati*

{ i.(Matricola,Nome,Età) | i(Impiegati) | }

Target list | Range list | condizione (vuota)

Esempio [2]

Q: *Trovare nome e stipendio dei capi degli impiegati che guadagnano più di 1.700*

Target list

Range list

{NomeCapo, StipCapo: c.(Nome,Stipendio) |

c(Impiegati), s(Supervisione), i(Impiegati) |

c.Matricola = s.Capo ∧ s.Impiegato=i.Matricola

∧ i.Stipendio > 1700

condizione di join

condizione di selezione

Esempio [3]

Q: *Trovare nome e stipendio dei capi degli impiegati che guadagnano più di 1700*

Target list

{NomeCapo, StipCapo: c.(Nome,Stipendio) |

Range list

c(Impiegati), s(Supervisione), i(Impiegati) |

c.Matricola = s.Capo ∧ s.Impiegato=i.Matricola

∧ i.Stipendio > 1700}

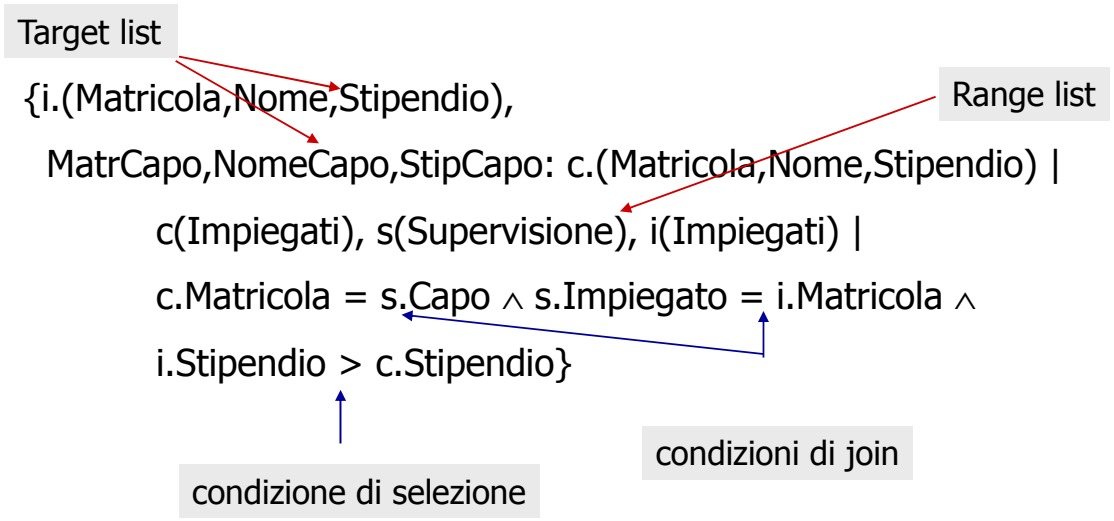
condizioni di join

condizione di selezione

Le due variabili c e i accedono ai valori della stessa relazione in modo indipendente

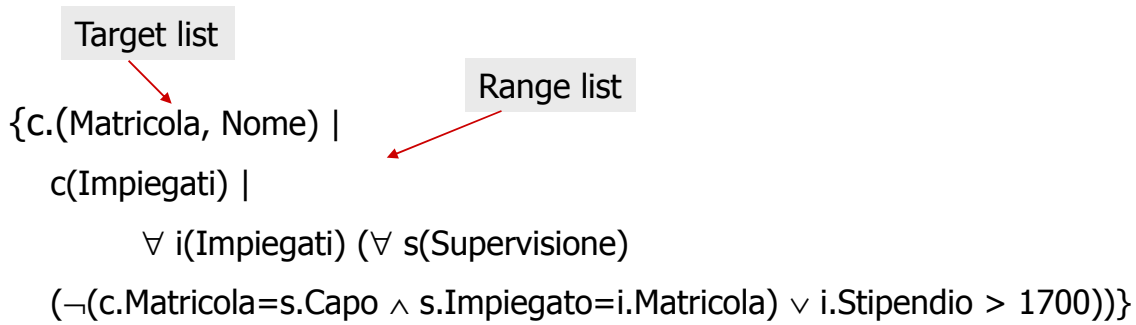
Esempio [4]

Q: *Trovare gli impiegati che guadagnano più del rispettivo capo, mostrando matricola, nome e stipendio di entrambi*



Esempio [5]

Q: *Trovare matricola e nome dei capi i cui impiegati guadagnano tutti più di 1700*



Esempio [6]

Q: *Trovare matricola e nome dei capi i cui impiegati guadagnano tutti più di 1700*

Target list

Range list

{c.(Matricola, Nome) |

c(Impiegati)|

¬ (∃ i(Impiegati) (∃ s(Supervisione)

(c.Matricola=s.Capo ∧ s.Impiegato=i.Matricola ∧ i.Stipendio ≤ 1700))}

condizioni di join rispettivamente per trovare i capi in Supervisione e accedere alla tupla con le informazioni di ogni impegato dipendente

Limitazioni: unione

Il calcolo su tuple con dichiarazioni di range, rispetto all'algebra relazionale, ha una limitazione:

- non permette di esprimere l'unione fra relazioni !
  - I risultati sono costruiti a partire dalle variabili libere
  - Ogni variabile ha come range una sola relazione

  es.

$$R_1(A) \cup R_2(A)$$

  - Se si ha una sola variabile libera si fa riferimento ad una sola relazione
  - Se si hanno due variabili si può far riferimento alle due relazioni ma non si riesce a combinarle per produrre il risultato in modo corretto
- SQL che è basato sul calcolo su tuple con dichiarazione di range prevede un operatore esplicito per l'unione

Operazioni insiemistiche

- Gli operatori di intersezione e differenza sono esprimibili!
  - Intersezione

$$\{ x_1.* \mid x_1(r_1) \mid \exists\, x_2(r_2)\ (x_1.A_1 = x_2.A_1 \wedge \ldots . x_1.A_k = x_2.A_k)\}$$

preleva le tuple di r_1 che hanno una tupla uguale in r_2

  - Differenza

$$\{ x_1.* \mid x_1(r_1) \mid \neg\exists\, x_2(r_2)\ (x_1.A_1 = x_2.A_1 \wedge \ldots . x_1.A_k = x_2.A_k)\}$$

preleva le tuple di r_1 che non hanno una tupla uguale in r_2

Altri limiti

- calcolo di valori derivati
  - possiamo solo estrarre valori, non calcolarne di nuovi
  - calcoli di interesse
    - a livello di tupla o di singolo valore (conversioni, somme, ecc.)
    - su insiemi di tuple (somme, medie, ecc.)
  - In SQL ci sono estensioni ad hoc
- interrogazioni inerentemente ricorsive, come la chiusura transitiva
  - Soluzione esterna a SQL.. con programma...

Chiusura transitiva

- Accesso ricorsivo alla stessa relazione per un numero non predeterminato di volte
  - Esempio

Supervisione(Impiegato, Capo)

Q: *Per ogni impiegato, trovare tutti i superiori (cioè il capo, il capo del capo, e cosi' via)*

Impiegato	Capo
Paolo Bianchi	Gaia Belli
Gaia Belli	Mario Mori
Mario Mei	Filippo Verdi

Impiegato	Superiore
Paolo Bianchi	Gaia Belli
Paolo Bianchi	Mario Mori
Gaia Belli	Mario Mori
Mario Mei	Filippo Verdi

Soluzione col join?

- Se la gerarchia è a 2 livelli si può pensare a una soluzione con un join di Supervisione con se stessa

{Impiegato: i, Superiore: s |

Supervisione(Impiegato: i,Capo: s) $\vee$

(Supervisione(Impiegato: i,Capo: c) $\wedge$

Supervisione(Impiegato: c, Capo: s)) }

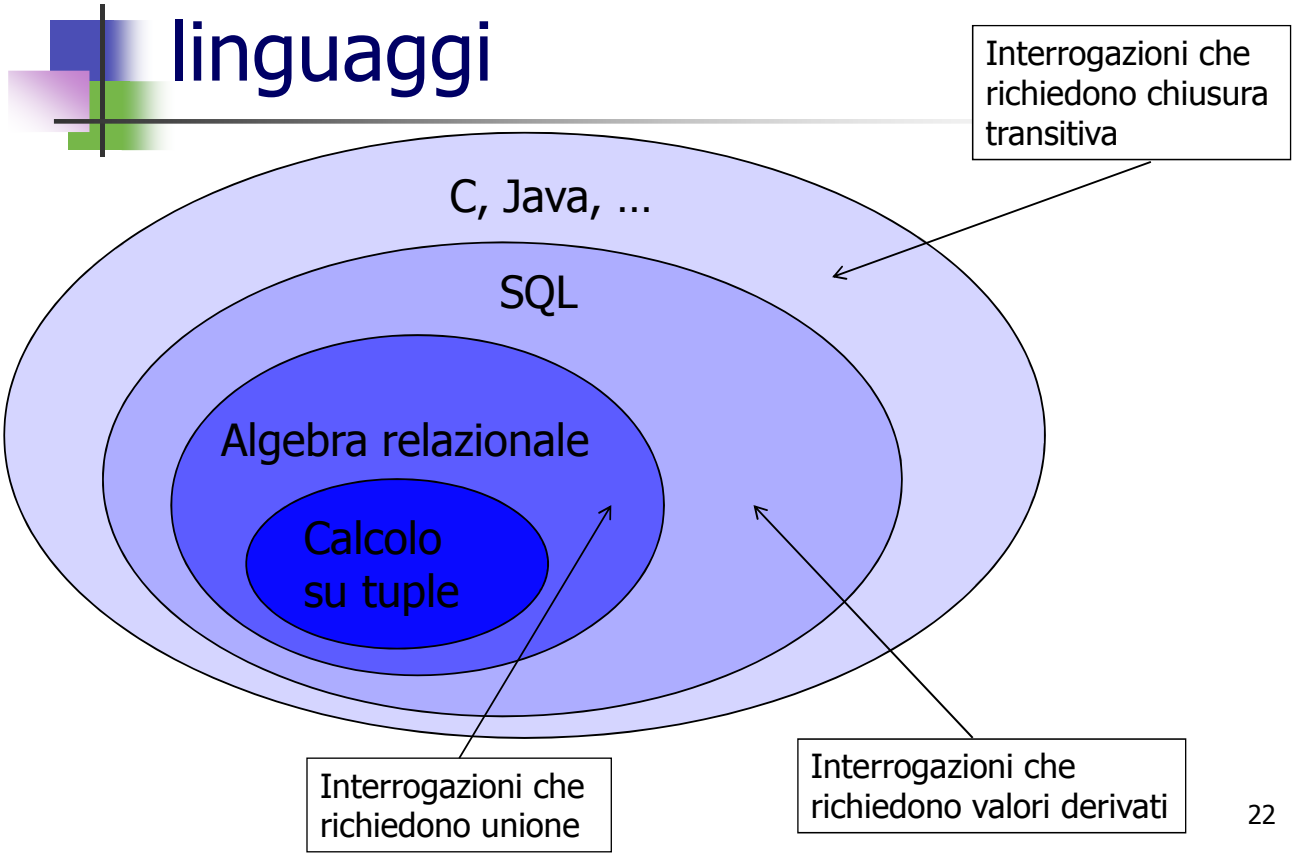
join per trovare i capi dei capi

- Se si vuole estendere a gerarchie più profonde si devono aggiungere join a più termini

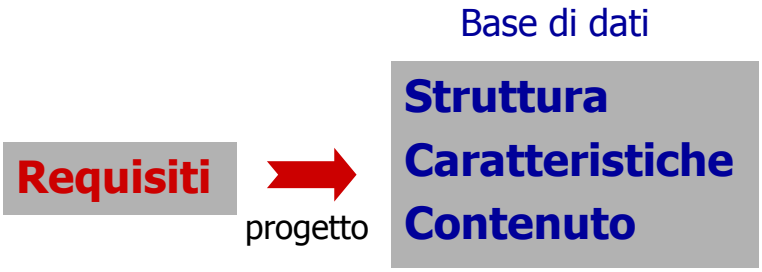
Conclusione

- Non esiste in algebra e calcolo relazionale la possibilità di esprimere l'interrogazione che, per ogni relazione binaria, ne calcoli la chiusura transitiva
- Per ciascuna relazione, è possibile calcolare la chiusura transitiva, ma con un'espressione ogni volta diversa:
  - quanti join servono?
  - non c'è limite!

Potere espressivo dei linguaggi



Progettazione di basi di dati



Metodologia in 3 fasi

- Progettazione concettuale
- Progettazione logica
- Progettazione fisica

Ciclo di vita dei sistemi informativi

- Comprende tutte le attività svolte da analisti, progettisti, utenti nello sviluppo e utilizzazione dei sistemi informativi
- E' un'attività iterativa... un ciclo di operazioni
  - spesso durante l'esecuzione di una attività occorre rivedere decisioni prese nelle attività precedenti
- Le basi di dati sono solo una delle componenti di un sistema informativo

Fasi del ciclo di vita [1]

- Studio di fattibilità
  - definizione dei costi delle alternative
  - priorità di realizzazione delle componenti del sistema
- Raccolta ed analisi dei requisiti
  - Studio delle proprietà e delle funzionalità del sistema
    - Interazioni con gli utenti (a livelli diversi)
    - Analisi delle realizzazione esistenti
    - Studio della normativa (es. riservatezza dei dati)
  - Descrizione informale dei dati e delle operazioni
  - Requisiti hardware e software

Fasi del ciclo di vita [2]

- Progettazione
  - Progettazione dei dati
  - Progettazione delle applicazioni

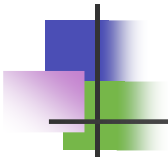
  Descrizioni formali per mezzo di modelli per dati e applicazioni
- Implementazione
  - Realizzazione del sistema informativo secondo la struttura prevista nella progettazione
    - Costruzione e popolazione della base di dati
    - Sviluppo del codice delle applicazioni

Fasi del ciclo di vita [3]

- Validazione e collaudo
  - Verifica del corretto funzionamento e della qualità
  - Si devono considerare tutte le possibili condizioni operative
- Funzionamento
  - Utilizzo operativo del sistema informativo. Possono essere richieste
    - Correzioni per malfunzionamenti
    - Revisioni delle funzionalità del sistema
    - Installazione, gestione e manutenzione

Analisi e progettazione

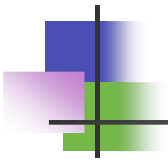
- L'analisi descrive cosa rappresentare in una base di dati
  - costruzione di un modello astratto rispetto alle realizzazioni
  - analisi delle procedure aziendali
  - riguarda i dati (specifiche sui dati) e le funzioni (specifiche sulle operazioni)
- La progettazione definisce come automatizzare le procedure
  - organizzazione dei dati in ingresso e uscita
  - architettura hardware e software
  - organizzazione dei moduli software



Progettazione di basi di dati [1]

Progettazione concettuale

- Rappresentare le specifiche della realtà di interesse (specifiche sui dati) con una descrizione formale
- Si utilizza un modello concettuale dei dati
- Il prodotto è uno **schema concettuale** che rappresenta il contenuto informativo della base di dati ed è utile anche a scopo documentativo
- La rappresentazione è indipendente dai modelli utilizzati per rappresentare i dati nei DBMS
- Non ci si interessa a come le informazioni verranno codificate e all'efficienza dei programmi che utilizzeranno queste informazioni



Progettazione di basi di dati [2]

Progettazione logica

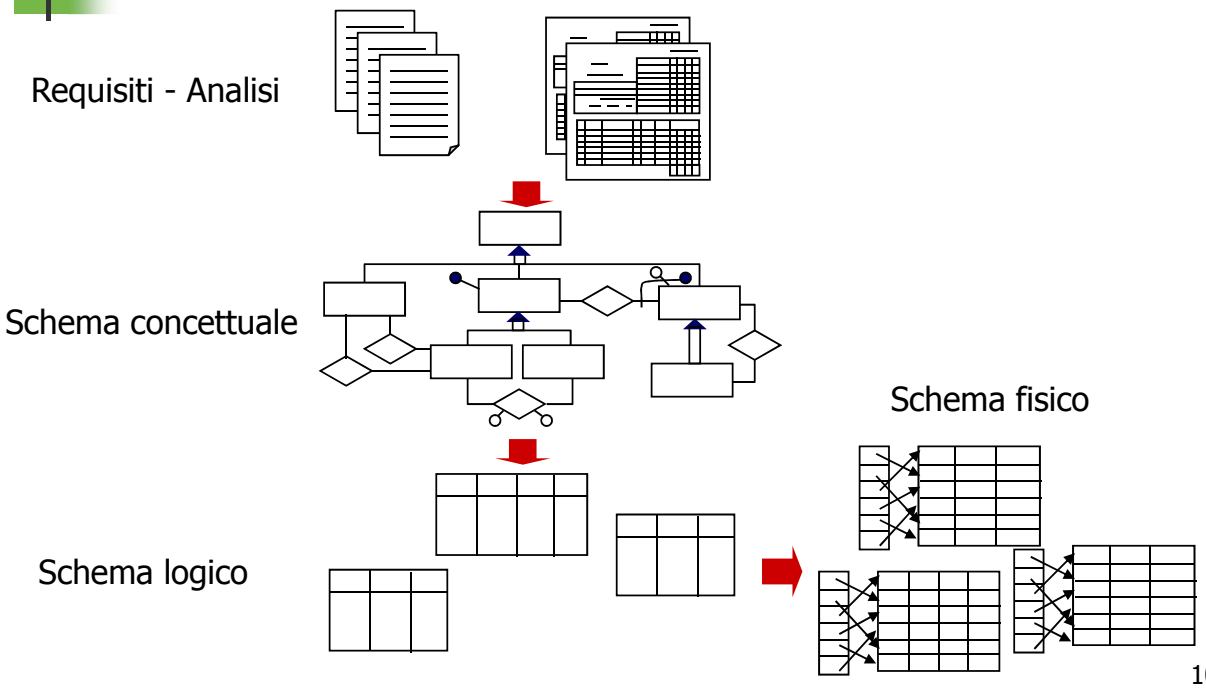
- Lo schema concettuale viene tradotto nel modello di rappresentazione dei dati utilizzato dal DBMS (es. modello relazionale)
- Si utilizza un modello logico dei dati
- Il prodotto è uno **schema logico** dei dati che fornisce una rappresentazione concreta del contenuto della base di dati
- Si effettuano le scelte progettuali per ottimizzare le operazioni che saranno effettuate sui dati
- Per il modello relazionale sono definite tecniche di normalizzazione per validare lo schema logico prodotto

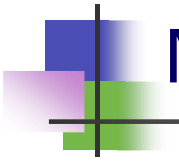
Progettazione di basi di dati [3]

Progettazione fisica

- Si scelgono i parametri fisici di memorizzazione dei dati (organizzazione dei file, definizione degli indici, tipo e dimensioni dei campi)
- Si tiene conto del modello logico e delle specifiche sulle operazioni per ottimizzare le prestazioni del sistema
- Si fa riferimento ad un modello fisico dei dati
- Il prodotto è uno schema fisico che dipende dal particolare DBMS utilizzato

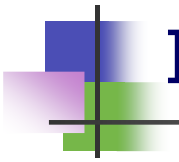
Fasi della progettazione e materiale prodotto





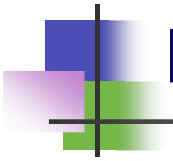
Modelli concettuali

- Rappresentazione dei dati in modo indipendente da ogni sistema e rappresentazione dei dati
- Descrizione dei concetti del mondo reale
- Rappresentazione mediante costrutti di
  - classi dei dati di interesse
  - correlazioni fra le classi di dati
- I costrutti definiscono schemi che descrivono l'organizzazione e la struttura delle istanze dei dati
- Per ogni costrutto esiste una sua rappresentazione grafica



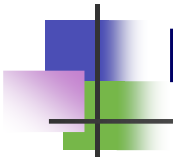
Il modello E-R

- Il modello Entità-Relazione è uno dei più diffusi modelli concettuali
- I costrutti del modello E-R sono
  - Entità
  - Relazioni
  - Attributi
  - Identificatori
  - Generalizzazioni e sottoinsiemi
- Uno schema E-R è rappresentato con un diagramma grafico



Entità

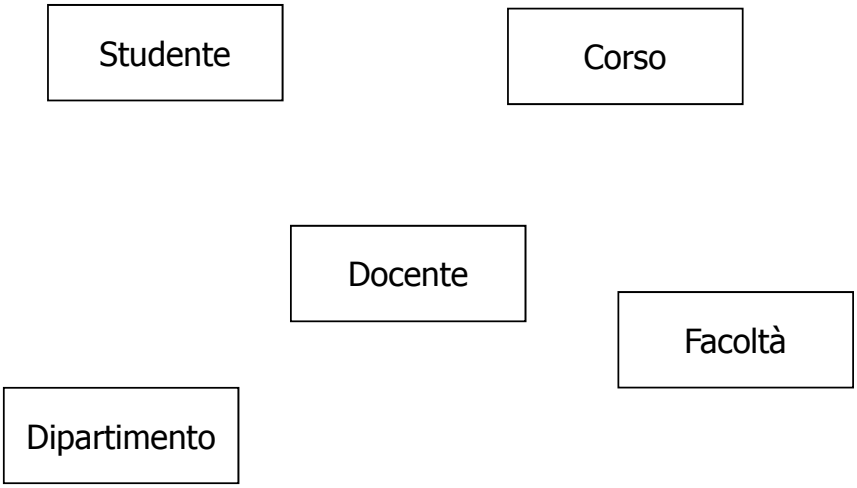
- Classe di oggetti che hanno proprietà comuni ed esistenza *autonoma*
  - Studente
  - Docente
  - Corso
  - Dipartimento
  - Facoltà
- Una istanza di una entità è un oggetto della classe rappresentata dall'entità
  - Facoltà di Ingegneria



Entità e occorrenze

- Una occorrenza di entità non è un valore (o valori) che identifica un oggetto ma l'oggetto in se stesso
- Una occorrenza di entità esiste indipendentemente dalle proprietà ad essa associate
- Ogni entità ha un nome che la identifica univocamente nello schema concettuale
  - Uso di nomi significativi ed espressivi
  - Rispetto di alcune convenzioni (es. usare il singolare)
- Graficamente le entità sono rappresentate da rettangoli

Rappresentazione delle Entità



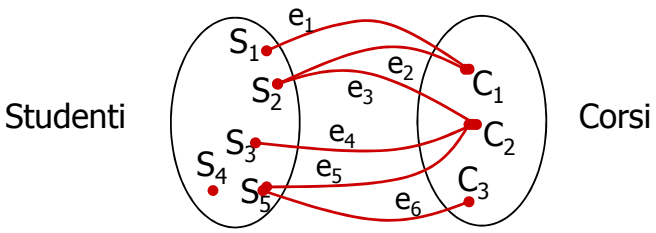
Relazione

- Legame logico fra due o più entità
  - Esame lega l'entità Studente con l'entità Corso
- Da non confondere col termine relazione del modello relazionale dei dati
- Una istanza di relazione è una n-upla di istanze di entità, una per ciascuna entità coinvolta
  - La coppia (*Mario Rossi*, *Basi di dati*) è un'istanza della relazione binaria Esame fra le entità Studente e Corso se
    - *Mario Rossi* è un'istanza dell'entità Studente
    - *Basi di dati* è un'istanza dell'entità Corso

Rappresentazione delle relazioni

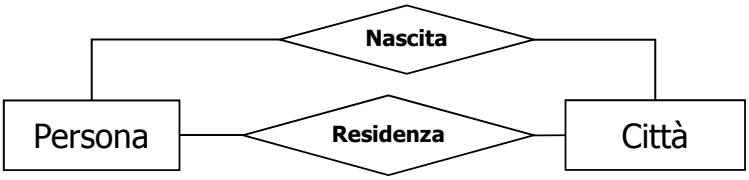
- In uno schema E-R ogni relazione ha un nome ed è rappresentata graficamente con un rombo e da linee che connettono la relazione con le sue componenti

- Esempio di istanze della relazione ESAME

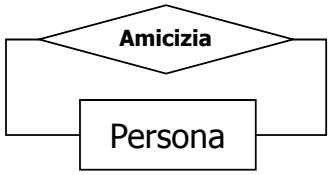


Esempi di relazioni

- Due entità possono essere coinvolte in più relazioni

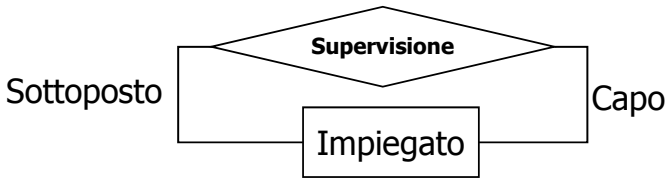


- Una relazione può essere ricorsiva ovvero una relazione fra un'entità e se stessa



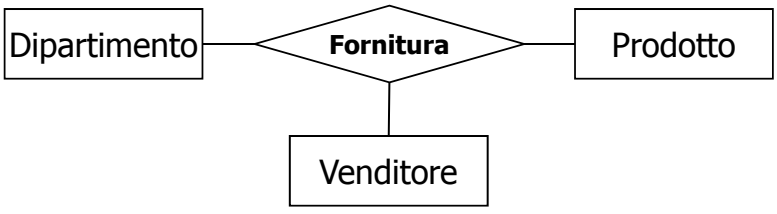
Relazioni con ruoli

- Una relazione ricorsiva può essere non simmetrica, ovvero nella coppia le due istanze dell'entità giocano ruoli diversi
- Si associano degli identificatori alle linee uscenti dalla relazione



Relazioni con più di due entità

- Una relazione può coinvolgere più di due entità

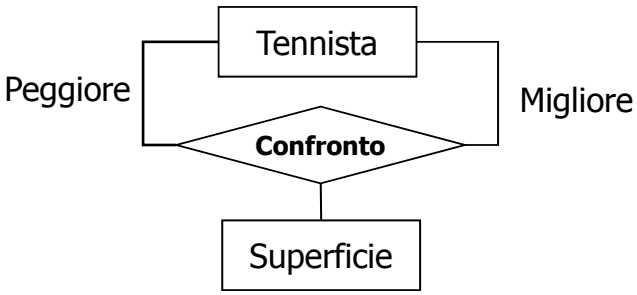


- Le istanze sono triple (Venditore,Prodotto,Dipartimento)

A	fornisce	stampanti	al Dip.	Ingegneria
A	....	calcolatori	.....	Fisica
B	....	calcolatori	...	Ingegneria

Relazioni con più di due entità e ricorsive

- Possono esistere relazioni ternarie ricorsive



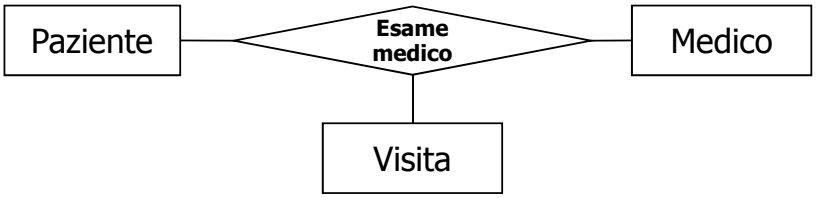
- Le istanze sono triple (Tennista,Tennista,Superficie)

Migliore		**Peggiore**		**Superficie**
Agassi	Migliore di	Becker	su	erba
Chang	.....	Noa	..	terra rossa
Becker	.....	Agassi	..	cemento

Relazioni e valori multipli

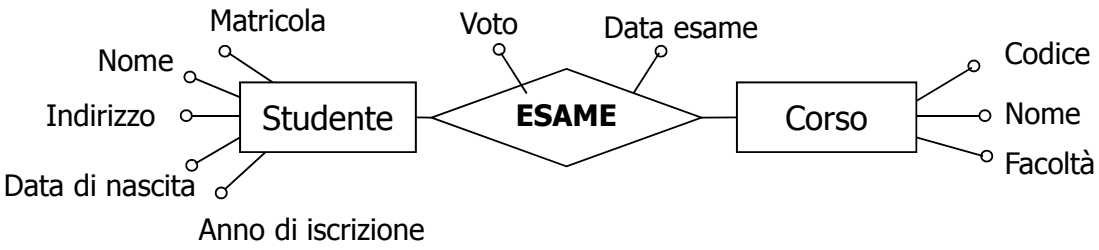
- Una relazione è un insieme e come tale può contenere solo istanze distinte (non ci possono essere *n*-uple ripetute)
  - Sottoinsieme del prodotto cartesiano degli insiemi di istanze coinvolte

- Un paziente può avere più visite con lo stesso medico
- Occorre trasformare visita in una entità e definire una relazione multipla



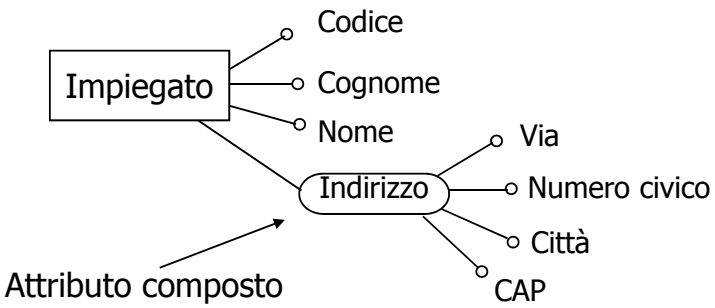
Attributi

- Descrivono proprietà elementari di entità o relazioni
- Ogni attributo associa a ciascuna istanza un valore appartenente al dominio dell'attributo
  - I domini non sono riportati nello schema ma sono descritti nella documentazione associata

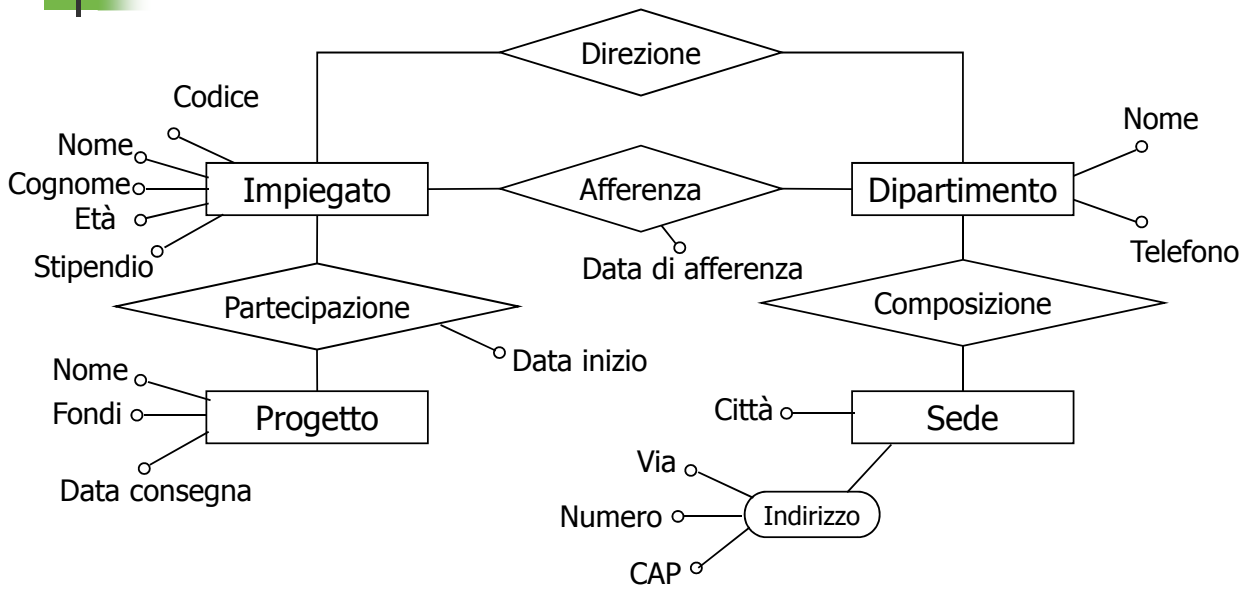


Attributi composti

- Si possono raggruppare gli attributi in base ad una affinità nel loro significato o uso
- L'attributo composto ha un nome che lo individua e un insieme di attributi atomici che lo compongono

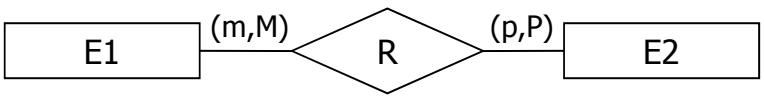


Uno schema ER



Cardinalità delle relazioni

- Per ogni partecipazione di una entità ad una relazione si specificano il numero minimo e massimo cui un'istanza dell'entità può partecipare
- Rappresenta un vincolo



- ogni istanza di E1 partecipa ad almeno m e al più a M istanze di R
- ogni istanza di E2 partecipa ad almeno p e al più P istanze di R

Esempio di cardinalità

- Ad ogni Impiegato è assegnato almeno un incarico
- Ogni Impiegato ha al più 5 incarichi
- Un Incarico può anche non essere ricoperto
- Ad un Incarico possono essere assegnati al massimo 50 Impiegati

Tipi di cardinalità

- In genere si specificano solo tre valori per i vincoli di cardinalità: 0, 1, N (intero maggiore di 1)

Cardinalità minima

- 0 - la partecipazione alla relazione è opzionale
- 1 - la partecipazione alla relazione è obbligatoria

Cardinalità massima

- 1 - rappresenta una funzione (parziale se la cardinalità minima è 0) che associa una sola istanza dell'altra entità
- N - rappresenta una associazione con un numero arbitrario di instanze dell'altra entità

Relazioni "molti a molti"

- Uno studente può non aver fatto esami o un certo numero di esami
- Nessun studente o più studenti possono aver sostenuto l'esame di un corso

- Una montagna può non essere stata scalata o scalata da molti alpinisti
- Un alpinista ha scalato almeno una montagna....

- Un autista ha almeno un turno
- Ogni autobus è usato almeno in un turno

Relazioni "uno a molti"

- Ogni Persona ha una sola Città di residenza
- Una Città può non avere Cittadini residenti

- Una Persona è disoccupata o ha un impiego
- Una Società ha almeno un impiegato

- Ogni Comune è in una sola Provincia
- Ogni Provincia ha almeno un Comune

Relazioni "uno a uno"

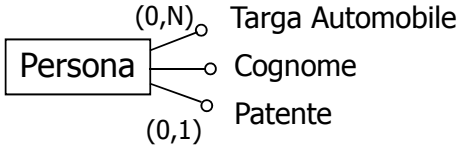
- Un uomo è o meno sposato con una donna
- ... e viceversa

- Ogni Patente ha un unico titolare
- Ogni persona può o meno avere la Patente

- Ogni Spettatore acquista un solo Biglietto

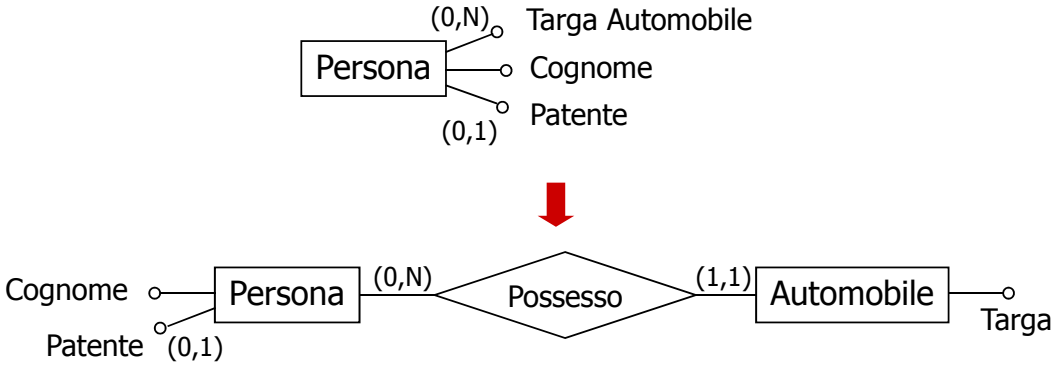
Cardinalità di attributi

- Per gli attributi si possono specificare il numero massimo e minimo di valori dell'attributo associati all'istanza
  - Nella maggior parte dei casi è (1,1) e viene omessa
  - Se la cardinalità è (0,1) l'attributo è opzionale (può assumere il valore NULL)
  - Se la cardinalità massima è N l'attributo può avere più valori per una istanza (attributo multivalore)



Cardinalità di attributi

- Gli attributi multivalore possono spesso essere sostituiti con entità legate da relazioni con l'entità a cui si riferiscono

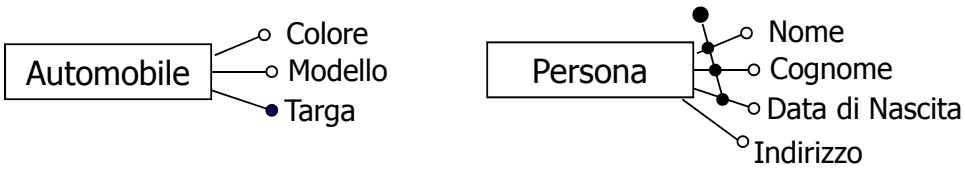


Identificatori delle entità

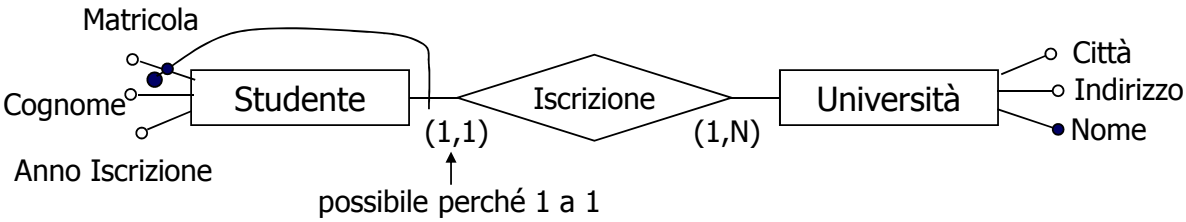
- Permettono di identificare in modo univoco le istanze dell'entità
- Può essere costituito da
  - identificatore interno (chiave)
    attributi (uno o più) dell'entità con cardinalità (1,1)
  - identificatore esterno
    entità esterne attraverso relazioni con cardinalità (1,1) ed eventualmente attributi
- Ogni entità deve avere almeno un identificatore
- Le relazioni sono identificate dagli attributi delle entità coinvolte

Esempi di identificatori

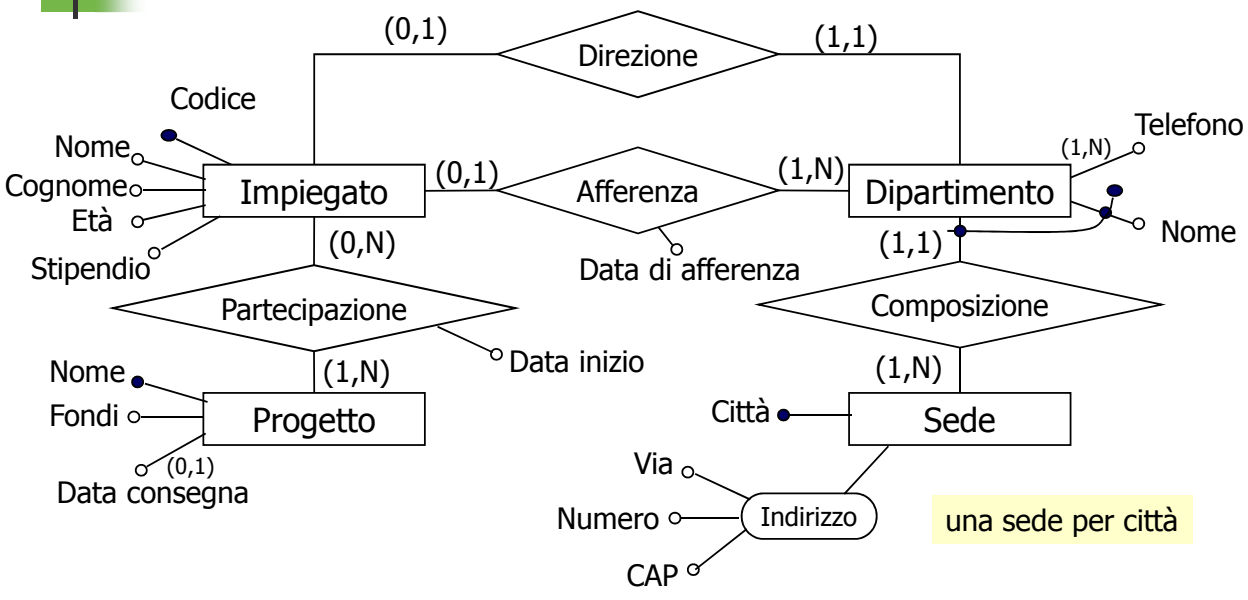
Identificatori interni



Identificatori esterni



Uno schema ER più completo



Generalizzazione

- Rappresenta un legame fra
  - Entità padre E
  - Entità figlie E_1, E_2, ..., E_n (specializzazioni o sottotipi)

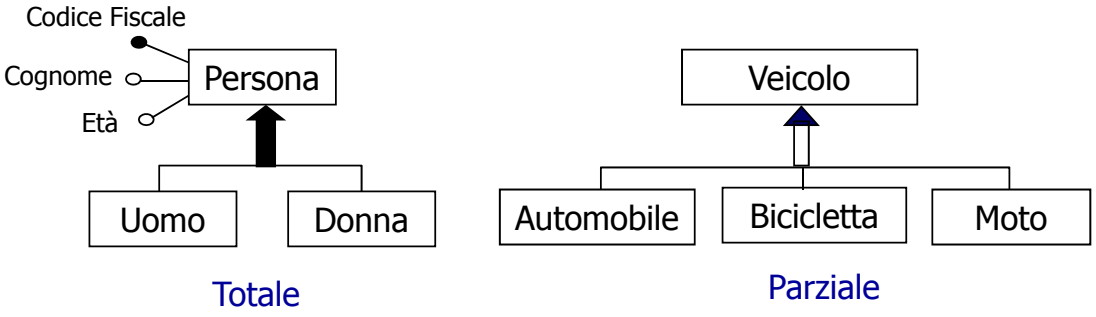
  E è un'entità più generale delle entità figlie, nel senso che le comprende come casi particolari
- Ogni istanza di E_i è anche istanza di E
- Ogni istanza di E è istanza al più di una E_i
- Ereditarietà

  Ogni proprietà di E (attributi,identificatori, relazioni e altre generalizzazioni) è ereditata anche dalle E_i

Esempi di generalizzazione

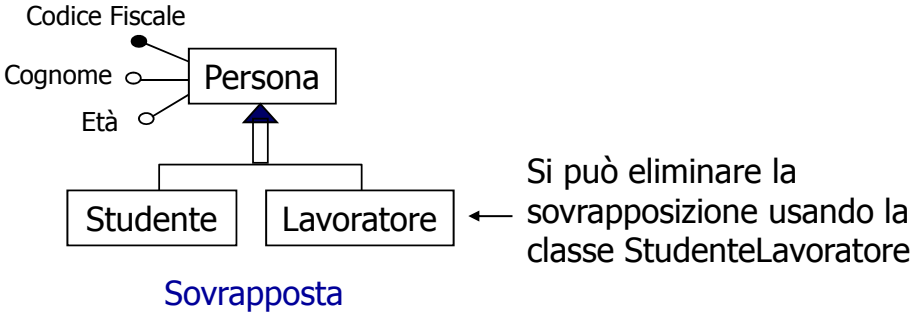
Generalizzazione totale

- La generalizzazione è totale se ogni istanza della classe padre è una istanza di almeno una delle classi figlie
- Altrimenti si dice parziale



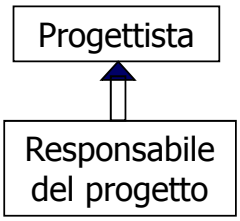
Generalizzazione esclusiva

- La generalizzazione è esclusiva se ogni istanza della classe padre è una istanza di al più una delle classi figlie
- Altrimenti si dice sovrapposta
  - Si possono eliminare le classi sovrapposte introducendo leclassi intersezione

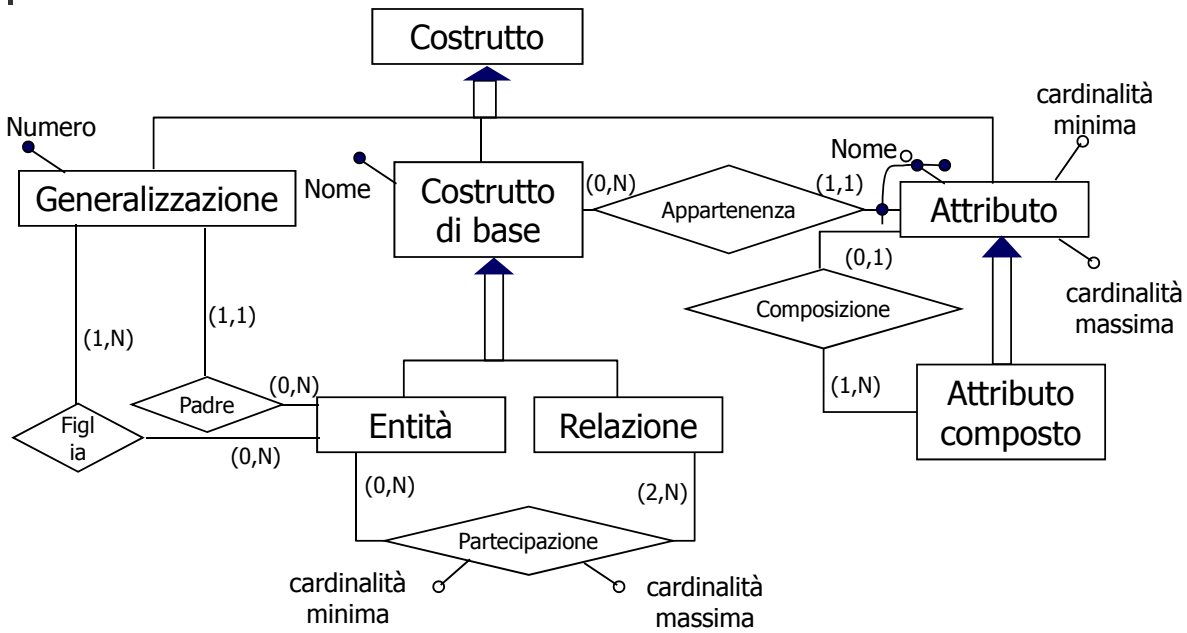


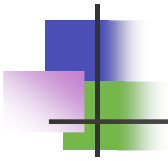
Generalizzazione "is-a"

- Nel caso di generalizzazione con un solo figlio si parla di sottoinsieme (gerarchia is-a)
- In questo caso la generalizzazione non può essere totale (le due entità sarebbero coincidenti)



Il modello ER in... ER





Documentazione

Dizionario dei dati

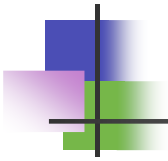
Spiegazione del significato dei concetti... oltre il nome

- Entità
- Relazioni
- Attributi
- Gerarchie

Vincoli non esprimibili

Impossibilità di rappresentare alcune proprietà dei dati

- Vincoli di integrità dei dati



Regole aziendali [1]

- Descrizione delle proprietà che non si riescono a rappresentare con il modello concettuale
- Regole del particolare dominio applicativo

Descrizioni dei concetti

- Si utilizza il linguaggio naturale
- Si costruisce un glossario (per entità e relazioni)

Regole aziendali [2]

Vincoli di integrità

- Vincoli espressi con un costrutto del diagramma ER (es. cardinalità delle relazioni)
- Vincoli non esprimibili con i costrutti del modello
- Definizione formale o con linguaggio naturale

Derivazioni

- Possibilità di ricavare un concetto con una inferenza o un calcolo aritmetico da altri concetti dello schema

Asserzioni

- Regole per descrivere i vincoli di integrità
- Sono affermazioni che devono essere verificate nella base di dati
- Le affermazioni sono atomiche
- Sono enunciate in modo dichiarativo

  <concetto> **deve**/**non deve** *<espressione sui concetti>*

  I concetti a cui si fa riferimento possono essere presenti direttamente nello schema ER o derivabili

Derivazioni

- Le regole specificano le operazioni che permettono di ottenere il concetto derivato

  <*concetto*> **`si ottiene`** <*operazione sui concetti*>

Esempio

(RD1) il numero di studenti di una facoltà si ottiene contando gli studenti che vi sono iscritti

(RD2) lo stipendio netto si ottiene sottraendo allo stipendio lordo il suo 33%

Regole aziendali e DBMS

- Le regole che esprimono vincoli e derivazioni sono implementate nella base di dati
  - Uso di clausole di SQL
    (**`assertion`**, **`foreign key`** e **`references`** , **`check`**, espressioni)
  - Uso di trigger o regole attive (basi di dati attive)
    Regole *Evento-Condizione-Azione*
    (when *Evento* if *Condizione* then *Azione*)
  - Uso di procedure scritte in un linguaggio di programmazione

Dizionario dei dati

- E' composto da due tabelle
  - Tabella di descrizione delle entità

Entità	**Descrizione**	**Attributi**	**Identificatore**
Nome dell'entità	Definizione informale	Elenco di tutti gli attributi con eventuali descrizioni	Possibili identificatori

  - Tabella di descrizione delle relazioni

Relazione	**Descrizione**	**Entità coinvolte**	**Attributi**
Nome della relazione	Definizione informale	Elenco delle entità coinvolte con cardinalità	Elenco di tutti gli attributi con eventuali descrizioni

Uno schema ER

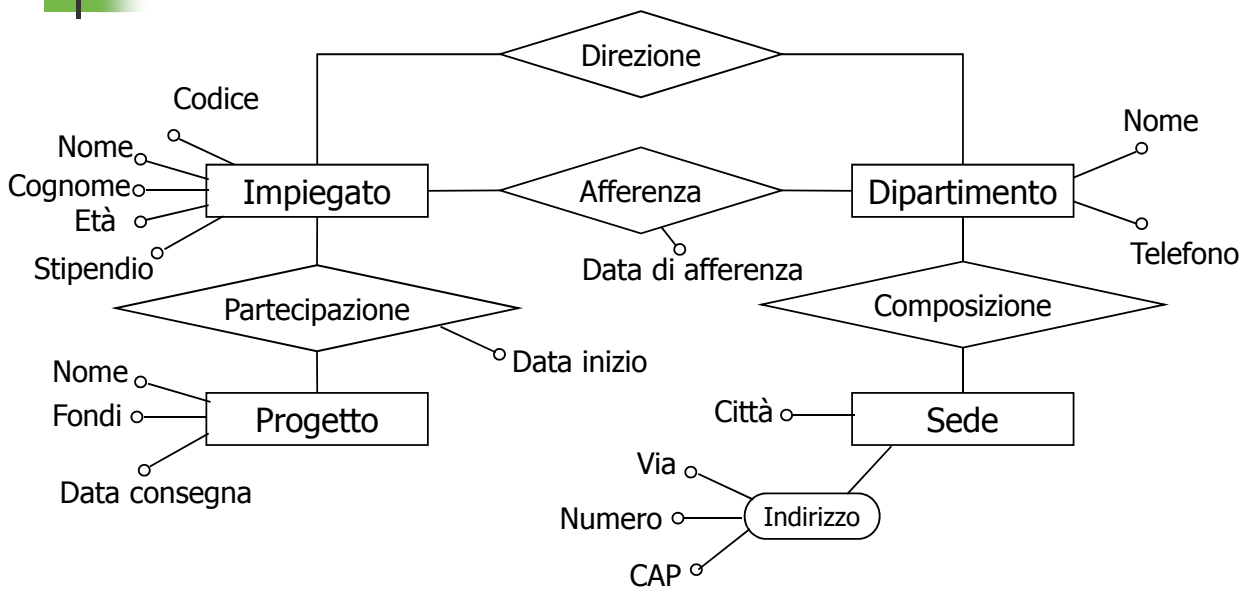


Tabella delle entità

Entità	Descrizione	Attributi	Identificatore
Impiegato	Impiegato che lavora nell'azienda	Codice, Cognome, Nome, Stipendio, Età	Codice
Progetto	Progetti aziendali sui quali lavorano gli impiegati	Nome, Fondi, Data consegna	Nome
Dipartimento	Dipartimenti delle sedi dell'azienda	Telefono, Nome	Nome, Sede
Sede	Sede dell'azienda in una certa città	Città, Indirizzo (Numero, Via e CAP)	Città

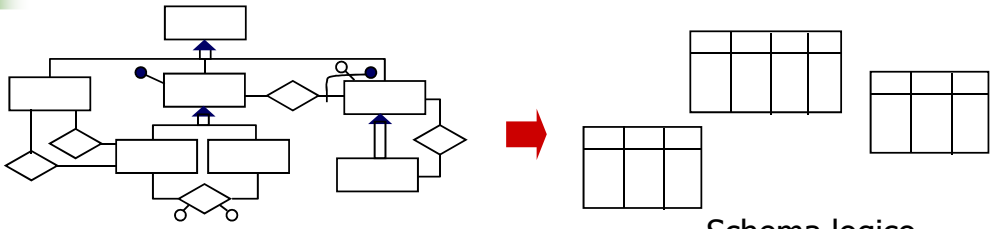
Tabella delle relazioni

Relazioni	Descrizione	Entità	Attributi
Direzione	Associa un dipartimento al suo direttore	Impiegato (0,1) Dipartimento (1,1)	
Afferenza	Associa un impiegato al suo dipartimento	Impiegato (0,1) Dipartimento (1,N)	Data afferenza
Partecipazione	Associa gli impiegati ai progetti sui quali lavorano	Impiegato (0,N) Progetto (1,N)	Data inizio
Composizione	Associa una sede ai dipartimenti di cui è composta	Dipartimento (1,1) Sede (1,N)	

Regole di vincolo e derivazione

Regole di vincolo
(RV1) Il Direttore di un dipartimento deve afferire a tale dipartimento (RV2) Un impiegato non deve avere uno stipendio maggiore del direttore del dipartimento al quale afferisce (RV3) Un dipartimento con sede a Roma deve essere diretto da un impiegato con più di dieci anni di anzianità (RV4) Un impiegato che non afferisce a nessun dipartimento non deve partecipare a nessun progetto
Regole di derivazione
(RD1) I fondi di un progetto si ottengono moltiplicando per 3 la somma degli stipendi degli impiegati che vi partecipano

Progettazione logica



Schema concettuale
Informazioni sul carico applicativo

Schema logico
Vincoli
Documentazione

- Costruzione di uno schema logico che descrive in modo corretto e efficiente tutte le informazioni presenti nello schema ER
  - Input: schema concettuale + informazioni sul carico applicativo + modello logico prescelto
  - Output: schema logico con vincoli e documentazione

Verso il modello logico

Riorganizzazione dello schema ER

- semplificazione della traduzione
  - non tutti i costrutti del modello ER hanno una traduzione naturale nel modello logico desiderato
    - Entità → relazione del modello relazionale con gli stessi attributi
    - Generalizzazione → ?
- ottimizzazione del progetto
  - si deve scegliere la soluzione che garantisce le migliori prestazioni

Traduzione verso il modello logico

- Ottimizzazioni mediante tecniche proprie del modello logico (normalizzazione per il modello relazionale)

Analisi delle prestazioni su schemi ER

- Lo schema ER può essere modificato per ottimizzare alcuni indici di prestazione
  - Spazio: dimensione dei dati
  - Tempo: costo di un'operazione valutato come numero di entità e relazioni che mediamente devono essere visitate
- Le prestazioni effettive non sono valutabili perché dipendono da parametri fisici (caratteristiche del DBMS)
- Per studiare le prestazioni occorre conoscere la dimensione dei dati e le caratteristiche delle operazioni

Volume dei dati

Volume dei dati

- numero di istanze per ogni entità e relazione dello schema
- dimensione di ogni attributo
- Nella tavola dei volumi si riportano per tutti i concetti (entità e associazioni) i volumi previsti a regime

Concetto	Tipo	Volume
<Nome>	E/R	<dim>

- Per le relazioni il volume dipende da
  - numero di istanze coinvolte nella relazione
  - il numero (medio) di partecipazioni di una istanza di entità alle istanze di relazione (dipende dalla cardinalità delle relazioni)

Caratteristiche delle operazioni

Caratteristiche delle operazioni

- tipo dell'operazione (interattiva/batch)
- frequenza (numero medio/unità di tempo)
- dati coinvolti (entità/relazioni)

- Nella tavola delle operazioni si riporta per ogni operazione la frequenza prevista e se l'operazione è interattiva (I) o batch (B)

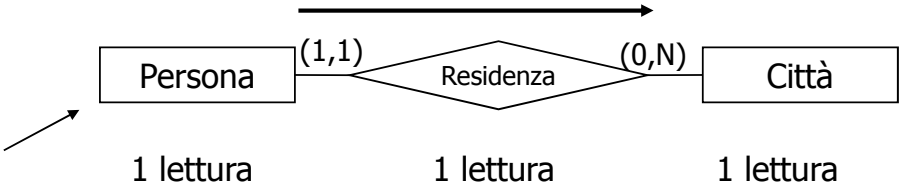
Operazione	Tipo	Frequenza
<Nome>	I/B	<freq>

- Si considerano le operazioni principali previste secondo la regola dell'80-20: si suppone che l'80% del tempo dipenda dal 20% delle operazioni più frequenti

Schema di operazione

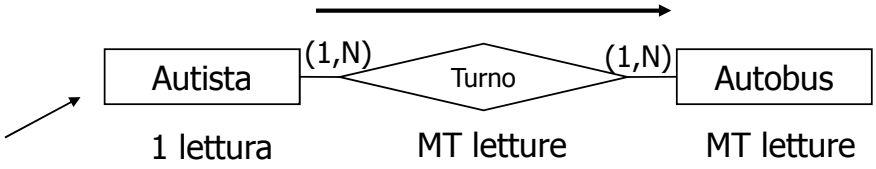
- Lo schema di operazione descrive i dati coinvolti in un'operazione
- Corrisponde al frammento dello schema ER interessato all'operazione sul quale viene disegnato il cammino logico per accedere alle informazioni di interesse

Operazione: data una Persona trovare la città di Residenza



Esempio 1

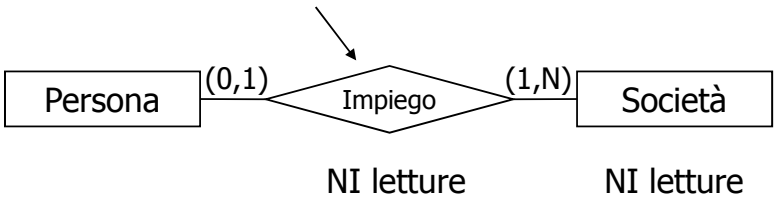
Operazione: dato un Autista trovare gli autobus dei suoi turni



- Per calcolare il numero di accessi a Turno e Autobus occorre conoscere il numero medio di Turni per Autista (MT)

Esempio 2

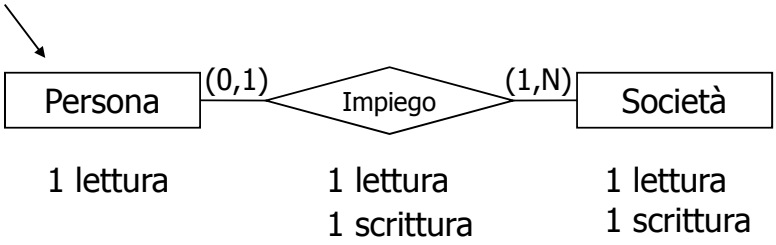
Operazione: Trovare le Società delle Persone che hanno un Impiego



- Per calcolare il numero di accessi a Persona e Società occorre conoscere il numero di Persone occupate (NI)

Esempio 3

Operazione: Assegnare ad una Persona nota un Impiego in una Società



- Si deve accedere alla Persona
- Verificare se la Società non è presente e nel caso scriverla
- Verificare se la Persona aveva già un Impiego e (sovra)scrivere la nuova associazione

Tavola degli accessi

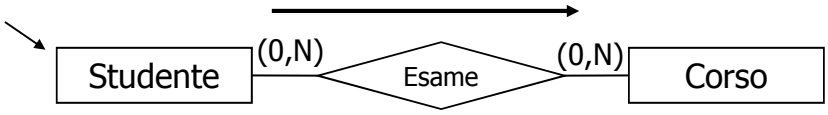
- Con lo schema di operazione si può fare una stima del costo di un'operazione contando il numero di accessi alle istanze di entità e relazioni
- Il risultato può essere riassunto in una tavola degli accessi

Concetto	Costrutto	Accessi	Tipo

- Il tipo distingue gli accessi in scrittura (S) e in lettura (L)
- Le operazioni di scrittura sono in genere più onerose (esecuzione in modo esclusivo, aggiornamento degli indici)

Esempio 1

Operazione: Stampare il curriculum di uno Studente

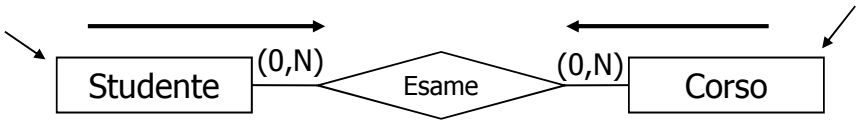


- Si suppone che la media degli esami sostenuti dagli studenti sia 10

Concetto	Costrutto	Accessi	Tipo
Studente	E	1	L
Esame	R	10	L
Corso	E	10	L

Esempio 2

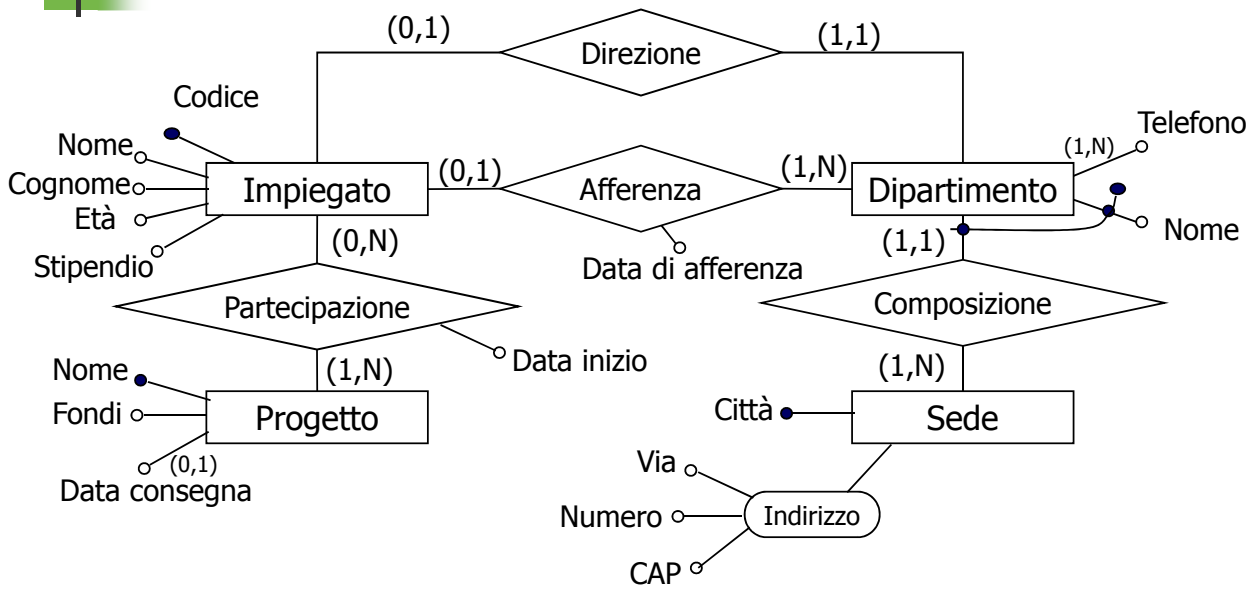
Operazione: Registrare un Esame di un Corso ad uno Studente



- Si suppone che si inseriscano solo Esami nuovi

Concetto	Costrutto	Accessi	Tipo
Studente	E	1	L
Esame	R	1	S
Corso	E	1	L

Schema ER



Operazioni

- Operazione 1: assegna un impiegato a un progetto
- Operazione 2: trova i dati di un impiegato, del dipartimento nel quale lavora e dei progetti ai quali partecipa
- Operazione 3: trova i dati di tutti gli impiegati di un certo dipartimento
- Operazione 4: per ogni sede, trova i suoi dipartimenti con il cognome del direttore e l'elenco degli impiegati

Operazione	Tipo	Frequenza
Op. 1	I	50/giorno
Op. 2	I	100/giorno
Op. 3	I	10/giorno
Op. 4	B	2/settimana

Volumi

Concetto	Tipo	Volume
Sede	E	10
Dipartimento	E	80
Impiegato	E	2000
Progetto	E	500
Composizione	R	80
Afferenza	R	1900
Direzione	R	80
Partecipazione	R	6000

- Composizione: è pari al numero di Dipartimenti (1 Dipartimento - 1 Sede)
- Afferenza: è paragonabile (leggermente inferiore) al numero di Impiegati
- Partecipazione: si assume che in media un Impiegato partecipi a 3 Progetti
- Direzione: ogni Dipartimento ha un Direttore

Schema di operazione

- Operazione 2: trova i dati di un impiegato, del dipartimento nel quale lavora e dei progetti ai quali partecipa

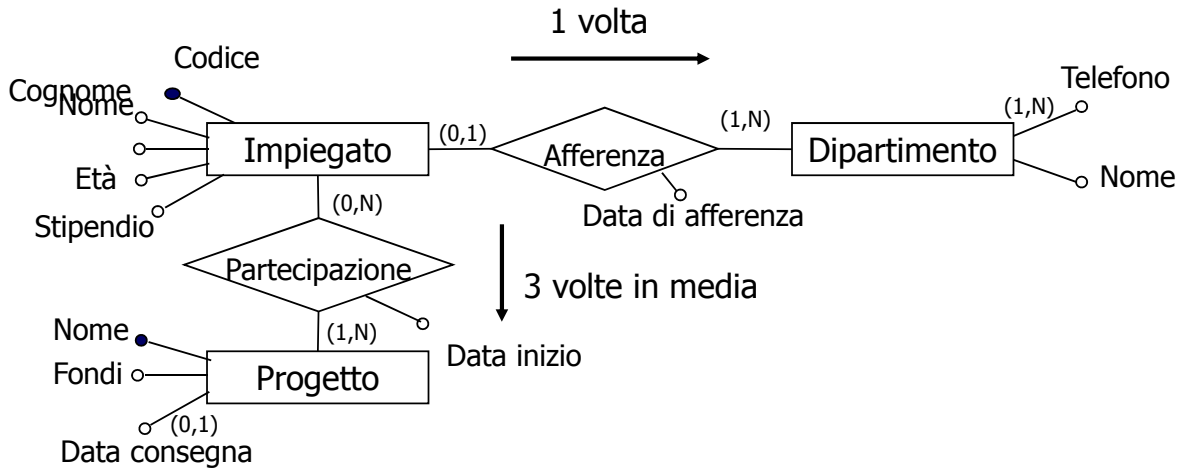


Tavola degli accessi

- Le informazioni richieste sono prelevate da
  - Impiegato: 1 accesso
  - Afferenza: 1 accesso (ogni Impiegato afferisce al più a un Dipartimento)
  - Dipartimento: 1 accesso
  - Partecipazione: 3 accessi (in media 1 Impiegato partecipa a 3 Progetti)
  - Progetto: 1 accesso
- Tutti gli accessi sono in lettura

Concetto	Costrutto	Accessi	Tipo
Impiegato	E	1	L
Afferenza	R	1	L
Dipartimento	E	1	L
Partecipazione	R	3	L
Progetto	E	3	L

Ristrutturazione degli schemi ER

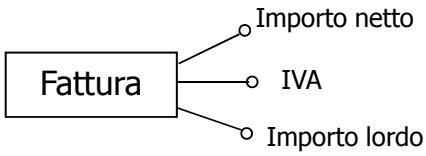
- Schema ER iniziale + specifiche sul carico applicativo
  - Analisi delle ridondanze
    - Mantenere/rimuovere le ridondanze dello schema iniziale
  - Eliminazione delle generalizzazioni
    - Le generalizzazioni sono analizzate e sostituite da altri costrutti
  - Partizionamento/accorpamento di entità e associazioni
    - Suddivisione di un concetto (entità o relazione) in più concetti
    - Accorpamento di più concetti separati in un solo concetto
  - Scelta degli identificatori primari
    - Per le entità che hanno più di un identificatore

Analisi delle ridondanze

- Ridondanza = presenza di un dato che può essere derivato (ottenuto attraverso una serie di operazioni) da altri dati
  - Vantaggi
    - riduzione degli accessi necessari per calcolare il dato
  - Svantaggi
    - maggiore occupazione di memoria (spesso trascurabile)
    - meno semplice da implementare: necessità di operazioni aggiuntive per mantenere il dato aggiornato
- La decisione viene presa confrontando i costi delle operazioni nel caso di presenza e assenza della ridondanza

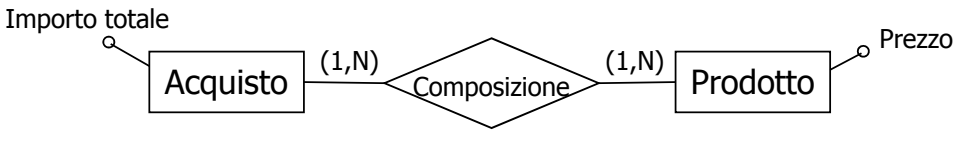
Esempi di ridondanze 1

- Attributi derivati da altri attributi della stessa istanza di entità (relazione)



Importo lordo =
(1+IVA/100) * Importo netto

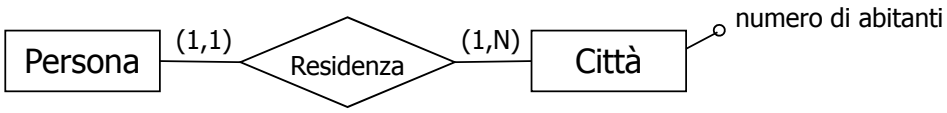
- Attributi derivati da attributi di altre entità (relazione) di solito attraverso funzioni aggregate



Acquisto.Importo totale = Σ Prodotto.Prezzo

Esempi di ridondanze 2

- Attributi derivati da operazioni di conteggio di occorrenze

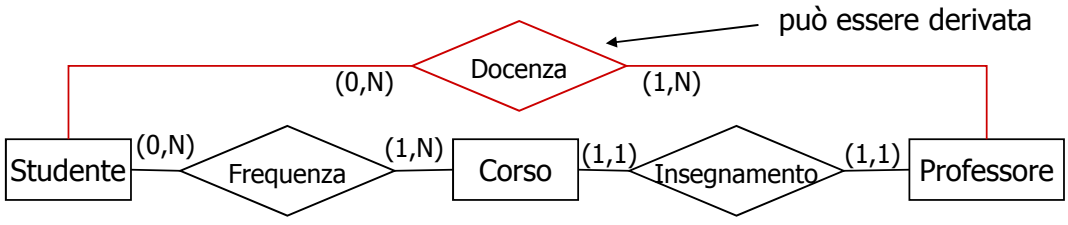


Città.numero di abitanti = Σ Residenza

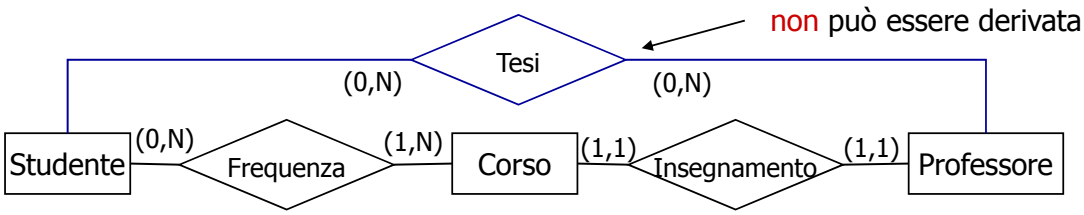
Data una città si devono contare le istanze della relazione Residenza in cui essa è coinvolta

Esempi di ridondanze 3

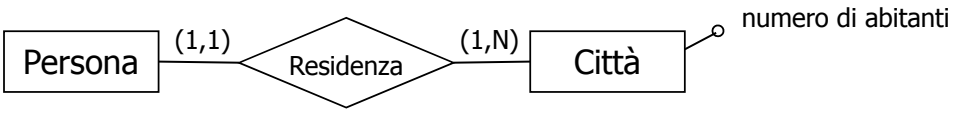
- Associazioni derivate dalla composizione di altre associazioni in presenza di cicli



- Occorre però considerare la semantica della relazione coinvolta nel ciclo



Esempio di analisi 1



- Operazione 1: memorizza una persona con la relativa città di residenza
- Operazione 2: stampa tutti i dati di una città

Concetto	Tipo	Volume
Città	E	200
Persona	E	1000000
Residenza	R	1000000

Tavola dei volumi

Operazione	Tipo	Frquenza
Operazione 1	I	500/giorno
Operazione 2	I	10/giorno

Tavola delle operazioni

Esempio di analisi 2

- Occupazione di memoria
  - 1 integer per città (4 byte)
  - 4 x 200 = 800 byte < 1Kb
- Costo delle operazioni

Operazione 1: memorizza una persona con la relativa città di residenza

con ridondanza

Concetto	Costrutto	Accessi	Tipo
Persona	E	1	S
Residenza	R	1	S
Città	E	1	L
Citta	E	1	S

senza ridondanza

Concetto	Costrutto	Accessi	Tipo
Persona	E	1	S
Residenza	R	1	S

legge il numero di abitanti precedente
e scrive il numero aggiornato

Esempio di analisi 3

- Costo delle operazioni

Operazione 2: stampa tutti i dati di una città

con ridondanza

Concetto	Costrutto	Accessi	Tipo
Città	E	1	L

senza ridondanza

Concetto	Costrutto	Accessi	Tipo
Città	E	1	L
Residenza	R	5000	L

conta in media 1000000/200 abitanti per città

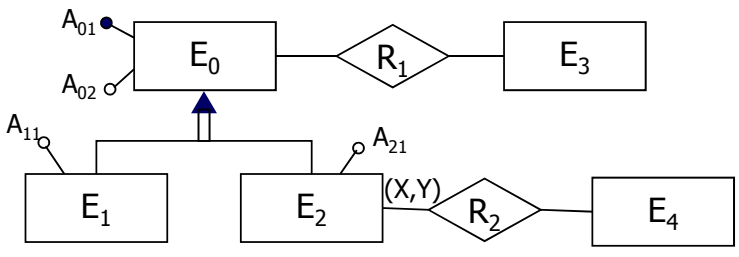
Totali

Operazione	L	S
Operazione 1	1 x 500/giorno	3 x 500/giorno
Operazione 2	1 x 10/giorno	0

Operazione	L	S
Operazione 1	1 x 500/giorno	1 x 500/giorno
Operazione 2	5000 x 10/giorno	0

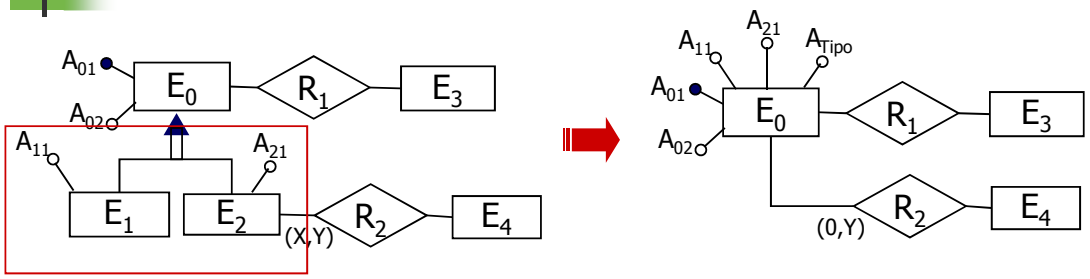
Eliminazione delle gerarchie

- Il modello relazionale non permette di rappresentare direttamente la generalizzazione



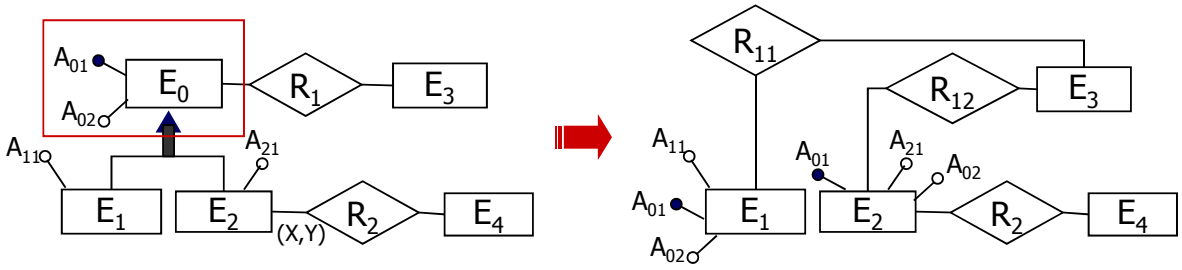
- Occorre trasformare le generalizzazioni in entità e relazioni
  - accorpamento delle figlie della generalizzazione nel genitore
  - accorpamento del genitore della generalizzazione nelle figlie
  - sostituzione della generalizzazione con relazioni

Accorpamento delle figlie



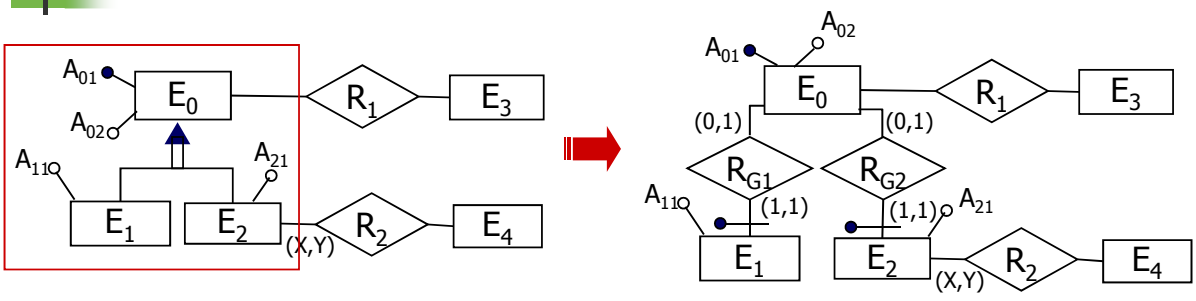
- Le entità figlie sono eliminate
- Gli attributi delle figlie sono aggiunte al padre
- Si aggiunge un attributo per distinguere il tipo (quale figlia è - o nessuna)
- Gli attributi che provengono da una figlia possono essere nulli
- Le relazioni (es. R_1) che provengono da una sola figlia hanno cardinalità minima pari a 0

Accorpamento del padre



- L'entità padre è eliminata
- Gli attributi, le relazioni e l'identificatore del padre sono aggiunti alle figlie
- Ogni relazione definita sul padre genera una relazione distinta per ogni figlia

Sostituzione con relazioni



- Si introduce una relazione uno-a-uno fra l'entità padre e ciascuna entità figlia
- Occorre inserire il vincolo che ogni istanza dell'entità padre può partecipare solo ad una relazione di legame con le entità figlie
- Se la generalizzazione è totale ogni istanza dell'entità padre partecipa necessariamente ad una (sola) delle relazioni di legame con le figlie

Scelta delle alternative 1

Accorpamento delle figlie nel padre

- Contro
  - Si ha spreco di memoria per i valori nulli
- Pro
  - Accesso contestuale agli attributi del padre e della figlia
  - E' la soluzione più semplice

Accorpamento del padre nelle figlie

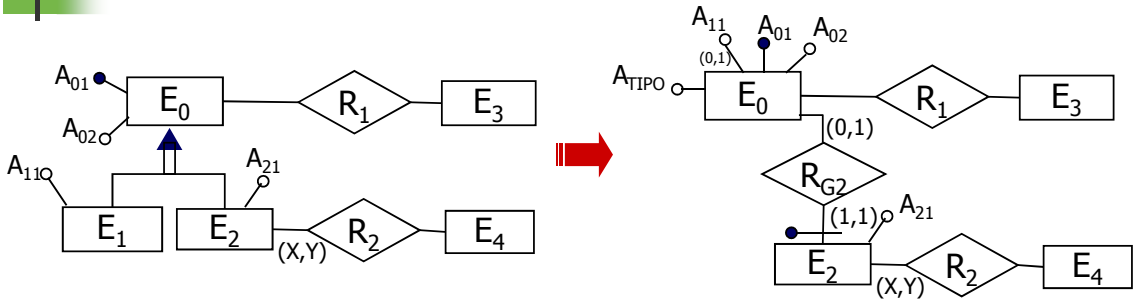
- Contro
  - Possibile solo se la generalizzazione è totale (non si possono rappresentare istanze che non sono in nessuna delle classi figlie)
- Pro
  - Migliore se si effettuano prevalentemente operazioni solo su istanze di una classe figlia
  - Non ci sono di principio valori nulli

Scelta delle alternative 2

Sostituzione con relazioni

- Contro
  - Si incrementa il numero degli accessi per mantenere la consistenza delle istanze rispetto ai vincoli introdotti
- Pro
  - Conviene quando la generalizzazione non è totale e ci sono operazioni che fanno distinzione fra le entità padre e le entità figlie
  - Non è necessario introdurre valori nulli
  - Genera entità con pochi attributi (le tabelle corrispondenti sono più piccole e più tuple possono essere gestite in memoria principale)

Soluzioni miste

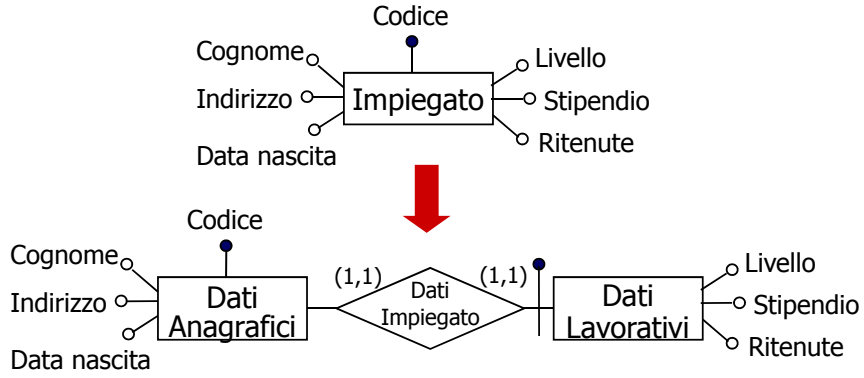


- Si è accorpata E_1 col padre E_0
- L'attributo A_{TIPO} serve per distinguere le istanze di E_1 da quelle di E_0
- Per E_2 invece si è introdotta una relazione R_{G2} che rappresenta la generalizzazione

Partizionamento/accorpamento

- Motivazioni
  - gli accessi si riducono separando attributi di uno stesso concetto che vengono considerati in operazioni diverse
  - gli accessi si riducono raggruppando attributi di concetti diversi a cui si accede contemporaneamente
- Tipologie
  - Partizionamento di entità
    - Verticale - opera su attributi
    - Orizzontale - opera sulle tuple. Corrisponde a introdurre una generalizzazione
  - Partizionamento (orizzontale) di relazioni
  - Accorpamento di entità

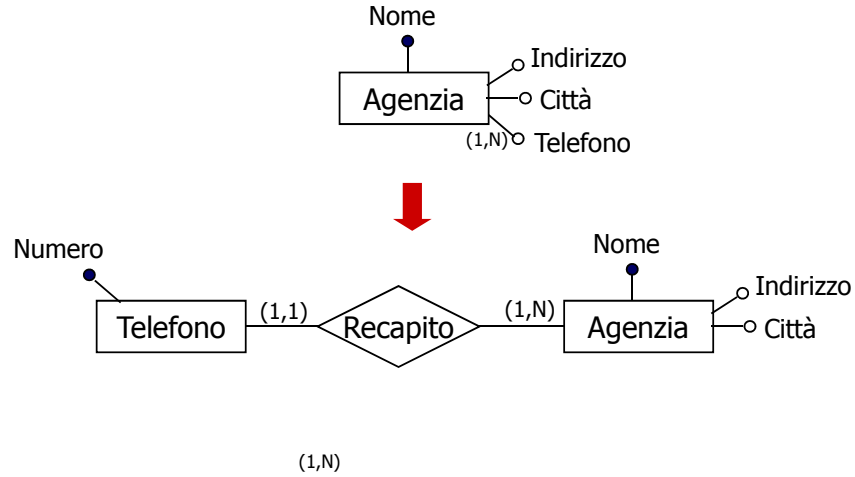
Partizionamento di entità



- Conviene se le operazioni richiedono solo informazioni di carattere anagrafico o lavorativo

Eliminazione di attributi multivalore

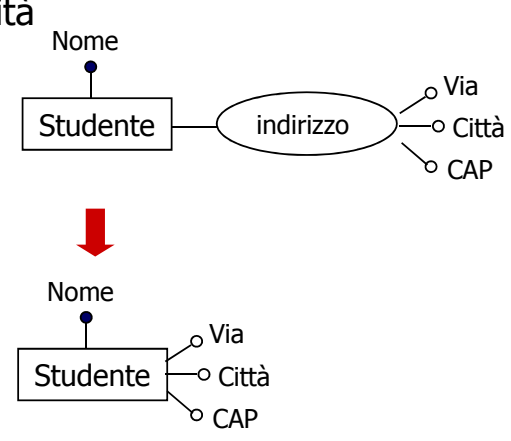
- Il modello relazionale non permette di rappresentare direttamente attributi multivalore
- Si può eliminare un attributo multivalore introducendo una entità e una relazione



(1,N)

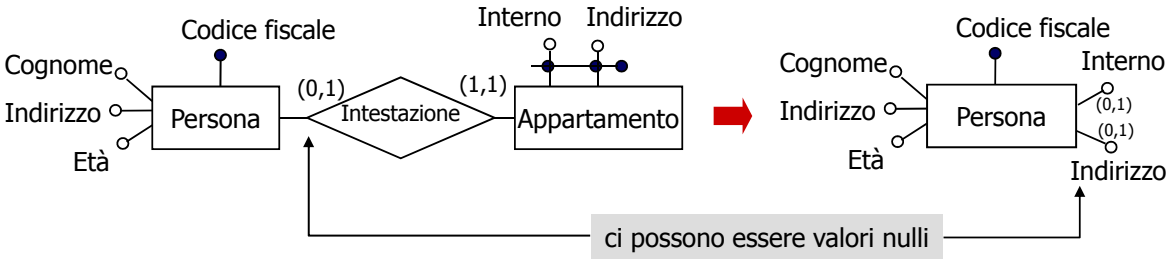
Eliminazione di attributi composti

- Il modello relazionale non permette di rappresentare direttamente attributi composto
- Si può eliminare un attributo composto associando gli attributi direttamente all'entità



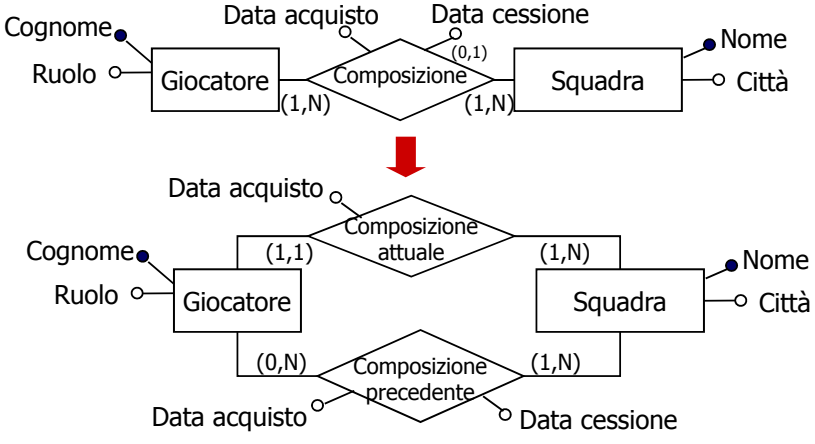
Accorpamento di entità

- Due entità legate da una relazione possono essere fuse in un'unica entità contenente gli attributi di entrambe quando le operazioni fanno sempre riferimento a tutti gli attributi delle due entità
- In questo modo si risparmiano gli accessi per recuperare i dati attraverso la relazione che lega le due entità
- Si effettuano per relazioni uno-a-uno (raramente per uno-a-molti dato che si generano ridondanze: gli attributi delle istanze della prima entità sono ripetuti in N tuple)



Partizionamento/accorpamento di relazioni

- Si può decomporre una relazione fra due entità in due o più relazioni fra le medesime entità per separare istanze della relazione originale a cui si accede sempre separatamente
- Viceversa si possono accorpare due relazioni tra le medesime entità



Scelta degli identificatori principali

- Le chiavi sono essenziali nel modello relazionale
  - permettono il collegamento fra relazioni
  - la chiave primaria permette la definizione di indici
- Quando un'entità ha più identificatori occorre scegliere la chiave principale (o introdurne una ad hoc → Codice)
  - assenza di valori nulli
  - identificatori con uno o pochi attributi
    - dimensioni indici ridotte, join più semplici
  - preferenza per gli identificatori interni
    - rispetto a identificatori esterni che coinvolgono molte entità
  - identificatori usati nelle operazioni principali

Ottimizzazione degli schemi ER: riassumendo

L'ottimizzazione dello schema ER può avere obiettivi diversi a seconda dell'applicazione

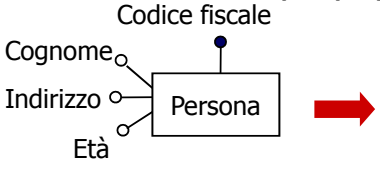
1. minimizzare l'occupazione della memoria
2. migliorare l'efficienza nell'esecuzione delle operazioni, minimizzando il numero di accesso alla memoria secondaria
3. semplificare lo schema del database e rendere più semplice il lavoro dei programmatore

Si osservi che con il migliorare dell'efficienza dell'hardware per molte applicazioni i punti 1 e 2 hanno un importanza limitata. In tali casi la semplificazione dello schema del database diventa l'obiettivo primario!!

Traduzione verso il modello relazionale

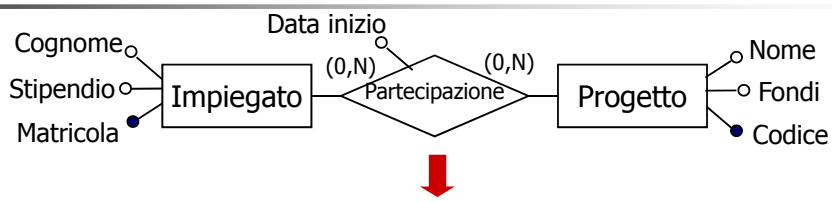
Traduzione verso il modello relazionale

- Si traduce lo schema ER ristrutturato in uno schema relazionale equivalente
- Trasformazioni di base
  - Una entità diviene una relazione definita sugli stessi attributi e con chiave uguale all'identificatore
  - Una relazione ER diventa una relazione sulle chiavi delle relazioni che rappresentano le entità coinvolte (più gli attributi propri)



Persona(<u>CodiceFiscale</u>,Cognome,Indirizzo,Età)

Relazioni molti-a-molti 1



Impiegato(<u>Matricola</u>,Stipendio,Cognome)
Partecipazione(<u>Impiegato</u>,<u>Progetto</u>,DataInizio)
Progetto(<u>Codice</u>,Nome,Fondi)

- La coppia di attributi Impiegato,Progetto identifica una istanza della relazione ER Partecipazione (è la chiave)
- Esiste un vincolo di integrità referenziale fra
  - Partecipazione.Impiegato e Impiegato.Matricola
  - Partecipazione.Progetto e Progetto.Codice

Relazioni molti-a-molti 2

Impiegato(<u>Matricola</u>,Stipendio,Cognome)
Partecipazione(<u>Impiegato</u>,<u>Progetto</u>,DataInizio)
Progetto(<u>Codice</u>,Nome,Fondi)

Impiegato

Cognome	Stipendio	Matricola
Rossi	500	0801
Verdi	800	0802
Bianchi	400	0823

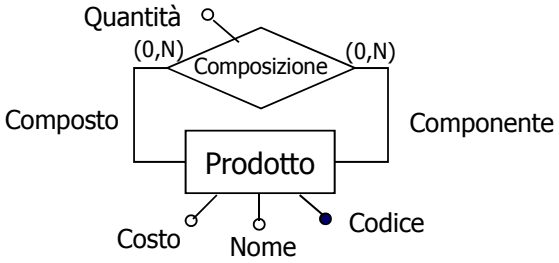
Codice	Fondi	Nome
P0032	3500	Alpha
P0054	7800	Beta
P0012	2400	Gamma

Progetto

Impiegato	DataInizio	Progetto
0801	22/3/2001	P0032
0801	12/11/2001	P0012
0802	10/7/2001	P0054

Partecipazione

Relazioni molti-a-molti 3



Prodotto(Codice,Nome,Costo)
Composizione(Composto,Componente,Quantità)

- Esiste un vincolo di integrità referenziale fra
  - Composizione.Composto e Prodotto.Codice
  - Composizione.Componente e Prodotto.Codice

Relazioni molti-a-molti 4

Prodotto(Codice,Nome,Costo)
Composizione(Composto,Componente,Quantità)

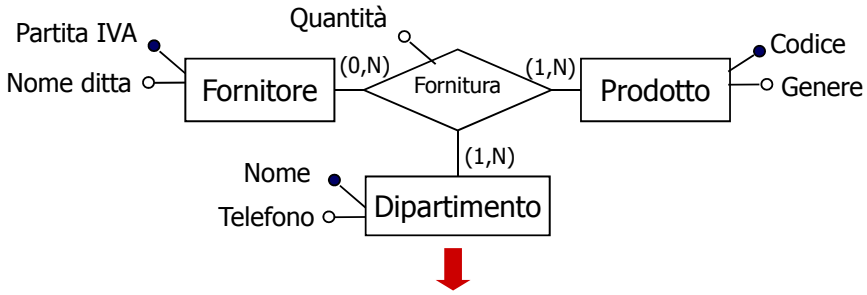
Prodotto

Nome	Costo	Codice
Cilindro	500	C003
Cuscinetto	800	C023
Sfera	300	V823

Composto	Componente	Quantità
C023	C003	2
C023	V823	20

Composizione

Relazioni molti-a-molti 5



Fornitore(PartitaIVA,NomeDitta)
Prodotto(Codice,Genere), Dipartimento(Nome,Telefono)
Fornitura(Fornitore,Prodotto,Dipartimento,Quantità)

- Esiste un vincolo di integrità referenziale fra
  - Fornitura.Fornitore e Fornitore.PartitaIVA
  - Fornitura.Prodotto e Prodotto.Codice
  - Fornitura.Dipartimento e Dipartimento.Nome

Relazioni molti-a-molti 6

Fornitore(PartitaIVA,NomeDitta)
Prodotto(Codice,Genere), Dipartimento(Nome,Telefono)
Fornitura(Fornitore,Prodotto,Dipartimento,Quantità)

NomeDitta	PartitaIVA
Rossi	08009382
Verdi	07092913
Bianchi	04563281

Fornitore

Codice	Genere
A0034	Computer
B3456	Stampante
V0567	Video

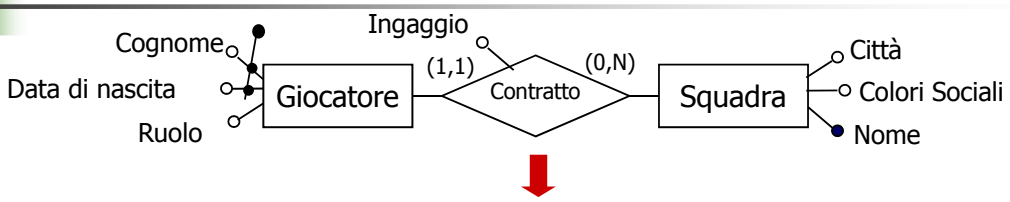
Prodotto

Fornitore	Prodotto	Dipartimento	Quantità
07092913	A0034	Fisica	5

Nome	Telefono
Ingegneria	0577233601
Fisica	0577232471
Biologia	0577234632

Dipartimento

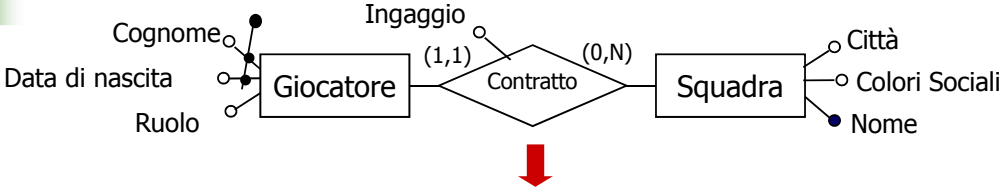
Relazioni uno-a-molti 1



Giocatore(Cognome,DataNascita,Ruolo)
Contratto(Giocatore,DataNascitaGiocatore,NomeSquadra,Ingaggio)
Squadra(Nome,Città,ColoriSociali)

- La chiave per la relazione Contratto è solo l'identificatore dell'entità Giocatore perché la cardinalità verso la relazione è (1,1)
- Le relazioni Giocatore e Contratto hanno un'unica chiave e quindi possono essere fuse insieme

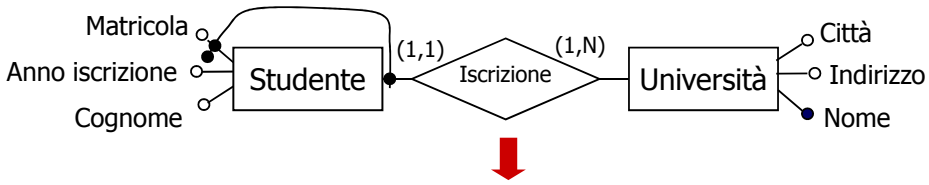
Relazioni uno-a-molti 2



Giocatore(Cognome,DataNascita,Ruolo,NomeSquadra,Ingaggio)
Squadra(Nome,Città,ColoriSociali)

- Esiste un vincolo di integrità referenziale fra
  - Giocatore.NomeSquadra e Squadra.Nome
- La soluzione è valida anche se la cardinalità della relazione Contratto verso Giocatore è (0,1)
  - Si hanno giocatori con l'attributo NomeSquadra uguale a NULL

Relazioni con identificatore esterno 1



Studente(<u>Matricola</u>,<u>NomeUniversità</u>,Cognome, AnnoIscrizione)
Università(<u>Nome</u>,Città,Indirizzo)

- Esiste un vincolo di integrità referenziale fra
  - Studente.NomeUniversità e Università.Nome
- Rappresentando l'identificatore esterno viene rappresentata anche la relazione fra le due entità

Relazioni con identificatore esterno 2

Studente(<u>Matricola</u>,<u>NomeUniversità</u>,Cognome, AnnoIscrizione)
Università(<u>Nome</u>,Città,Indirizzo)

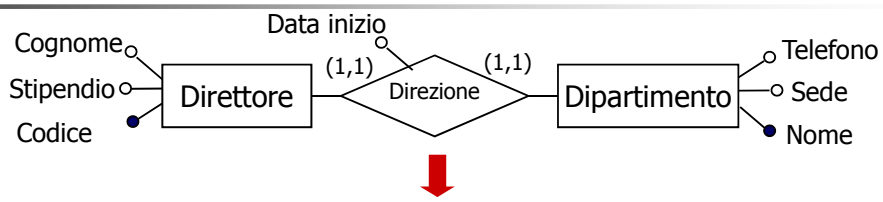
Studente

Matricola	**NomeUniversità**	**Cognome**	**AnnoIscrizione**
A80023633	UNISI	Rossi	1993
A80023633	UNIFI	Verdi	1996
A80023635	UNISI	Bianchi	2000

Nome	**Città**	**Indirizzo**
UNISI	Siena	Via Banchi di sotto
UNIFI	Firenze	P.zza S. Marco

Università

Relazioni uno-a-uno 1



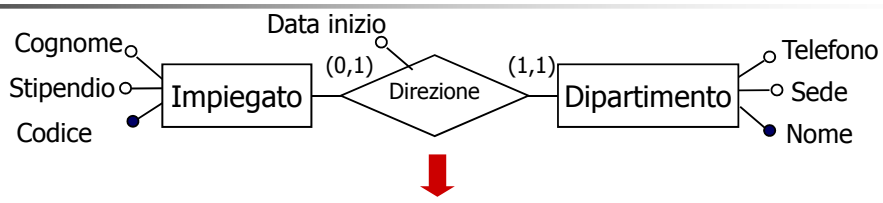
Direttore(Codice,Cognome,Stipendio,DipartimentoDiretto,InizioDirezione)
Dipartimento(Nome,Telefono,Sede)

oppure

Direttore(Codice,Cognome,Stipendio)
Dipartimento(Nome,Telefono,Sede,Direttore,InizioDirezione)

- La relazione è biunivoca e quindi può essere rappresentata in una qualsiasi delle relazioni che rappresentano le entità
- Si potrebbe anche definire un'unica relazione che fonde entrambe ma si perde la distinzione fra le due entità

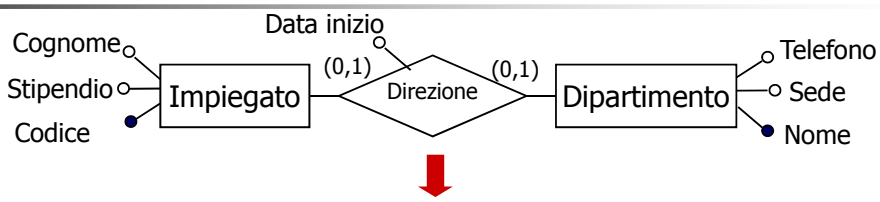
Relazioni uno-a-uno 2



Impiegato(Codice,Cognome,Stipendio)
Dipartimento(Nome,Telefono,Sede,Direttore,InizioDirezione)

- In questo caso la partecipazione della prima entità è opzionale
- Rappresentare la relazione Direzione in Dipartimento piuttosto che Impiegato è preferibile perché nell'altro caso si avrebbero molti valori nulli

Relazioni uno-a-uno 3

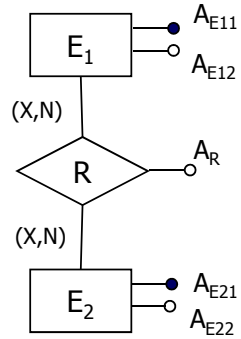


Impiegato(Codice,Cognome,Stipendio)
Direzione(Direttore,Dipartimento,DataInizioDirezione)
Dipartimento(Nome,Telefono,Sede)

- In questo caso la partecipazione di entrambe le entità è opzionale
- Questa soluzione ha il vantaggio di non presentare valori nulli sugli attributi
- Occorre però introdurre una relazione in più
- Conviene se il numero di istanze che partecipano alla relazione sono poche

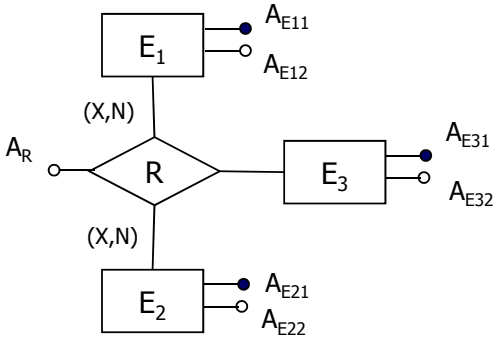
Traduzione delle relazioni 1

Relazione binaria molti-a-molti



$E_1(\underline{A_{E11}},A_{E12})$
$R(\underline{A_{E11}},\underline{A_{E21}},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

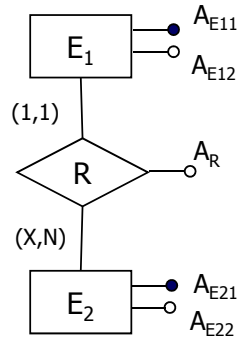
Relazione ternaria molti-a-molti



$E_1(\underline{A_{E11}},A_{E12})$
$E_2(\underline{A_{E21}},A_{E22})$
$E_3(\underline{A_{E31}},A_{E32})$
$R(\underline{A_{E11}},\underline{A_{E21}},\underline{A_{E31}}\ A_R)$

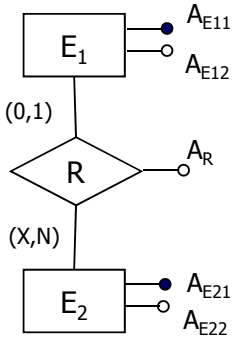
Traduzione delle relazioni 2

Relazione uno-a-molti con partecipazione obbligatoria



$E_1(\underline{A_{E11}},A_{E12},A_{E21},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

Relazione uno-a-molti con partecipazione opzionale



possono essere nulli

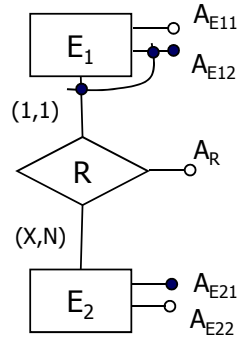
$E_1(\underline{A_{E11}},A_{E12})$
$R(\underline{A_{E11}},\underline{A_{E21}},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

o

$E_1(\underline{A_{E11}},A_{E12},A_{E21},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

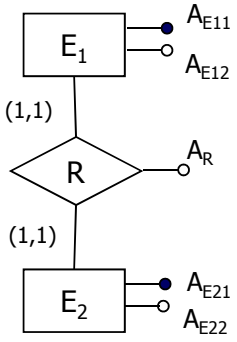
Traduzione delle relazioni 3

Relazione con identificatore esterno



$E_1(\underline{A_{E12}},\underline{A_{E21}},A_{E11},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

Relazione uno-a-uno con partecipazione obbligatoria di entrambe le entità



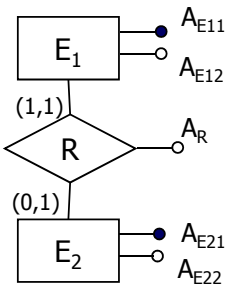
$E_1(\underline{A_{E11}},A_{E12},A_{E21},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

o

$E_2(\underline{A_{E21}},A_{E22},A_{E11},A_R)$
$E_1(\underline{A_{E11}},A_{E12})$

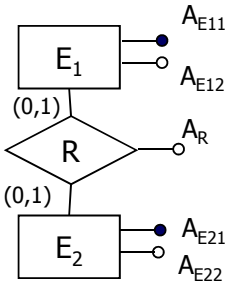
Traduzione delle relazioni 4

Relazione uno-a-uno con partecipazione opzionale di una entità



$E_1(\underline{A_{E11}},A_{E12},A_{E21},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

Relazione uno-a-unoi con partecipazione opzionale di entrambe le entità



possono essere nulli

$E_1(\underline{A_{E11}},A_{E12},A_{E21},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

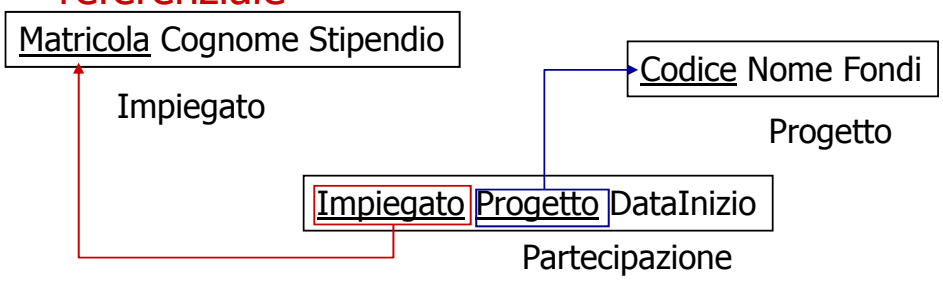
$E_2(\underline{A_{E21}},A_{E22},A_{E11},A_R)$
$E_1(\underline{A_{E11}},A_{E12})$

o

$E_1(\underline{A_{E11}},A_{E12})$
$R(\underline{A_{E11}},A_{E21},A_R)$
$E_2(\underline{A_{E21}},A_{E22})$

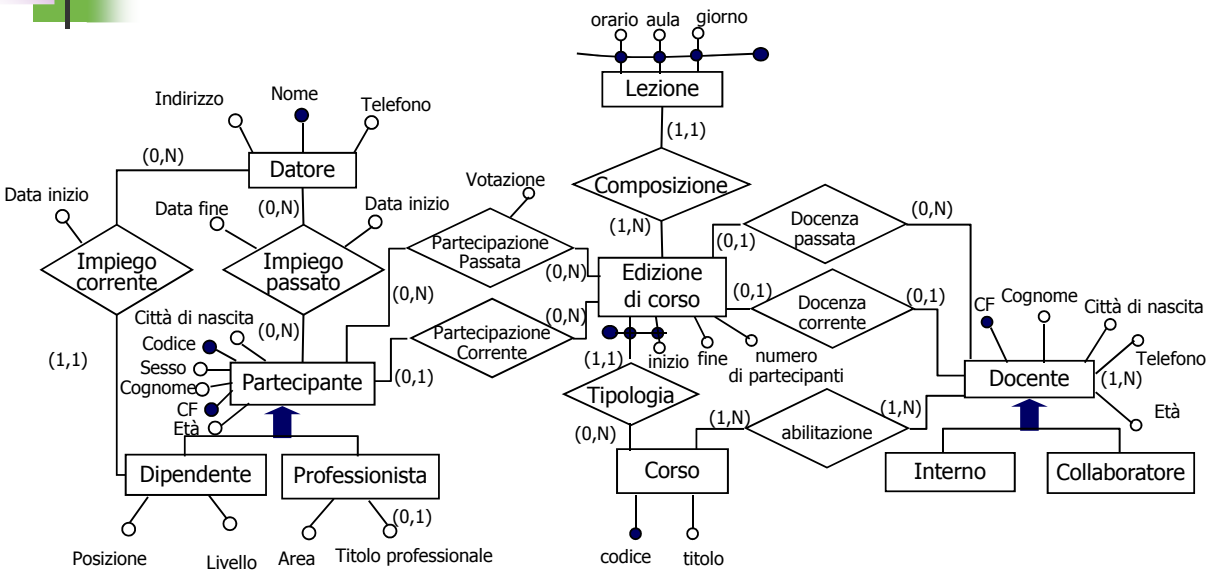
Documentazione degli schemi logici

- Si devono documentare i vincoli di integrità referenziale



- Il diagramma è utile anche per visualizzare i cammini di join

Esempio: società di formazione



Esempio: operazioni

- Operazione 1: inserisci un nuovo partecipante indicando tutti i suoi dati
- Operazione 2: assegna un partecipante ad una edizione di un corso
- Operazione 3: inserisci un nuovo docente indicando tutti o suoi dati e i corsi che può insegnare
- Operazione 4: assegna un docente abilitato a una edizione di corso
- Operazione 5: stampa tutte le informazioni sulle edizioni passate di un corso con titolo, orari delle lezioni e numero dei partecipanti
- Operazione 6: stampa tutti i corsi offerti, con informazioni sui docenti che possono insegnarli
- Operazione 7: per ogni docente, trova tutti i partecipanti a tutti i corsi da lui insegnati
- Operazione 8: effettua una statistica su tutti i partecipanti a un corso, con tutte le informazioni su di essi, sulla edizione alla quale hanno partecipato e la rispettiva votazione

Esempio: il carico

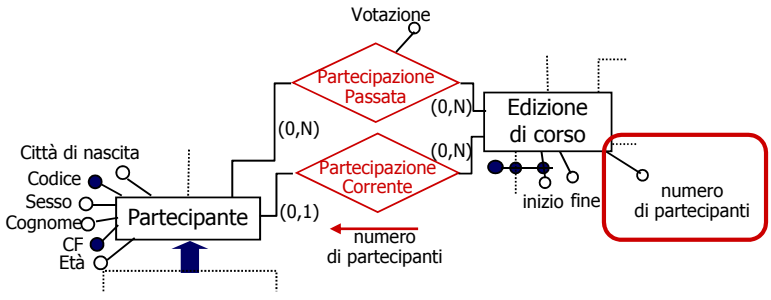
Concetto	Tipo	Volume
Lezione	E	8000
Edizione corso	E	1000
Corso	E	200
Docente	E	300
Collaboratore	E	250
Interno	E	50
Partecipante	E	5000
Dipendente	E	4000
Professionista	E	1000
Datore	E	8000
Part. Passata	R	10000
Part. Corrente	R	500
Composizione	R	8000
Tipologia	R	1000
Doc. passata	R	900
Doc. corrente	R	100
Abilitazione	R	500
Impiego corrente	R	4000
Impiego passato	R	1000

Tavola dei volumi

Operazione	Tipo	Frequenza
Op. 1	I	40/giorno
Op. 2	I	50/giorno
Op. 3	I	2/giorno
Op. 4	I	15/giorno
Op. 5	I	10/giorno
Op. 6	I	20/giorno
Op. 7	I	5/sett.
Op. 8	B	10/mese

Tavola delle operazioni

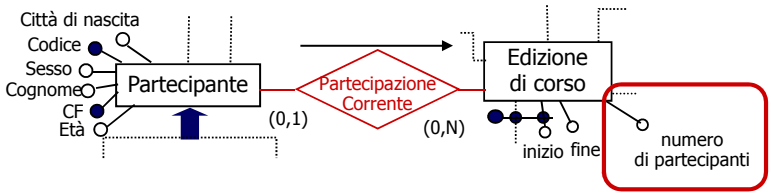
Esempio: ridondanze



- Occupazione di memoria: Volume Edizione Corso x 4 byte = 4000 byte
- Operazioni coinvolte:
  - Op. 2: assegna un partecipante ad una edizione di un corso
  - Op. 5: stampa tutte le informazioni sulle edizioni passate di un corso
  - Op. 8: può essere trascurata perché poco frequente e batch

Esempio: operazione 2

Op. 2: assegna un partecipante ad una edizione di un corso



con ridondanza

Concetto	Costrutto	Accessi	Tipo
Partecipante	E	1	L
Par. corrente	R	1	S
Edizione corso	E	1	L
Edizione corso	E	1	S

2 x 50 S + 2 x 50 L /giorno

senza ridondanza

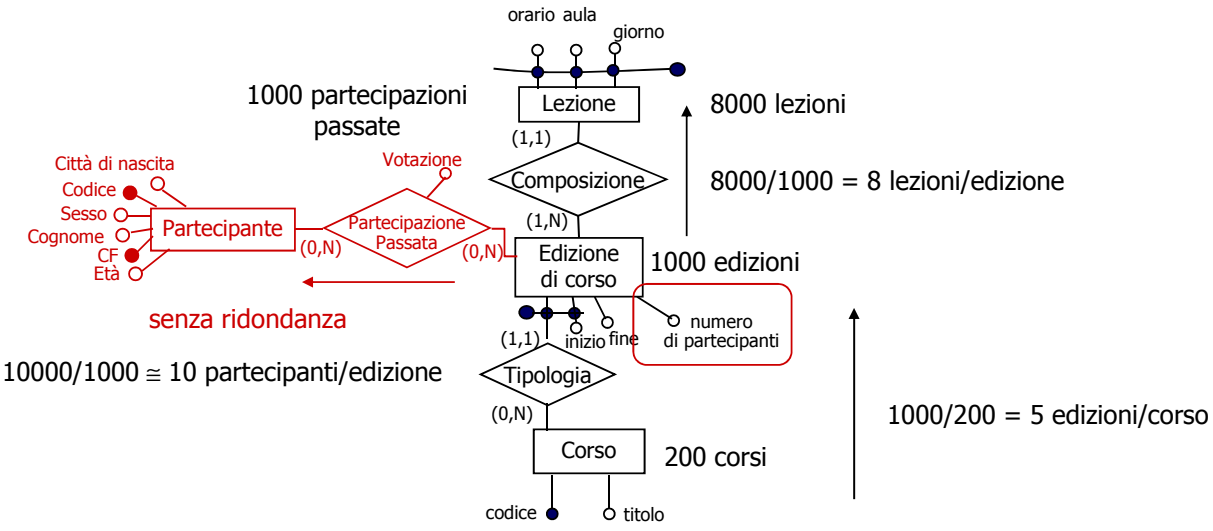
Concetto	Costrutto	Accessi	Tipo
Partecipante	E	1	L
Par. corrente	R	1	S

aggiornamento di "numero di partecipanti"

1 x 50 S + 1 x 50 L /giorno

Esempio: operazione 5/a

Op. 5: stampa tutte le informazioni sulle edizioni passate di un corso con titolo, orari delle lezioni e numero dei partecipanti



Esempio: operazione 5/b

con ridondanza

Concetto	Costrutto	Accessi	Tipo
Corso	E	1	L
Tipologia	R	1	L
Edizione corso	E	5	L
Composizione	R	5x8=40	L
Lezione	E	40	L

87 x 10 L /giorno

senza ridondanza

Concetto	Costrutto	Accessi	Tipo
Corso	E	1	L
Tipologia	R	1	L
Edizione corso	E	5	L
Part. passata	R	5x10=50	L
Composizione	R	5x8=40	L
Lezione	E	40	L

137 x 10 L /giorno

conteggio dei partecipanti alle edizioni passate

Esempio: scelta ridondanza

Totali

Operazione	L	S
Operazione 2	100/giorno	100/giorno
Operazione 5	870/giorno	0

con ridondanza

Operazione	L	S
Operazione 2	50/giorno	50/giorno
Operazione 5	1370/giorno	0

senza ridondanza

- Considerando doppio il costo per le operazioni di scrittura
  - totale con ridondanza: 1170
  - totale senza ridondanza: 1520
- Si decide quindi di mantenere la ridondanza

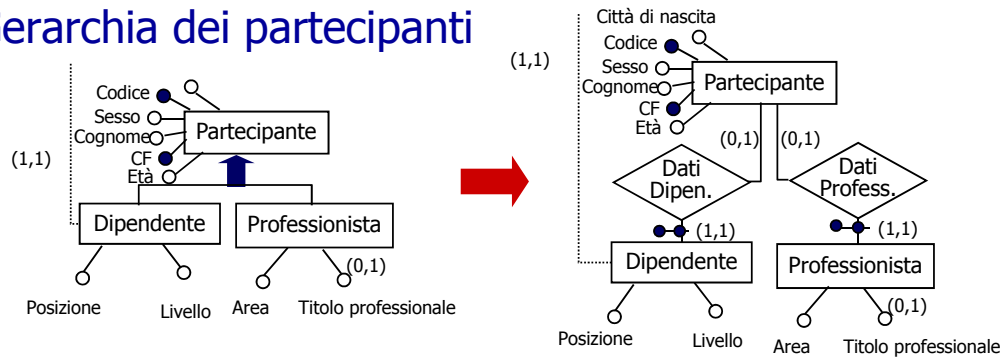
Esempio: gerarchie /a

Gerarchia dei docenti

- Le operazioni che fanno riferimento ai docenti (3,4,6,7) non distinguono i sottotipi
- I due sottotipi non hanno attributi specifici
- Si accorpano le figlie nel padre aggiungendo l'attributo Tipo che ha per dominio l'insieme dei valori C (collaboratore) e I (interno).

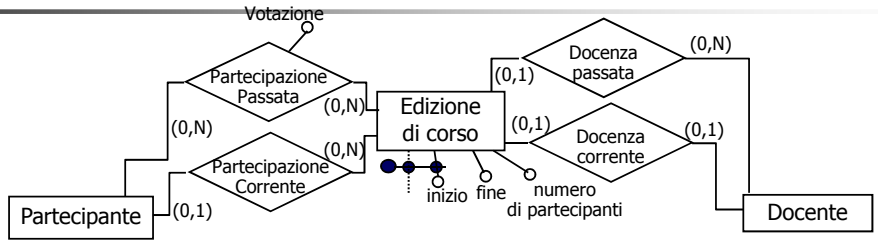
Esempio: gerarchie /b

Gerarchia dei partecipanti



- Le operazioni che fanno riferimento ai partecipanti (1,2,8) non distinguono i sottotipi
- I due sottotipi hanno attributi specifici
- E' preferibile lasciare i due sottotipi come entità e usare due relazioni per evitare di avere entità con troppi attributi e/o con attributi con valori nulli

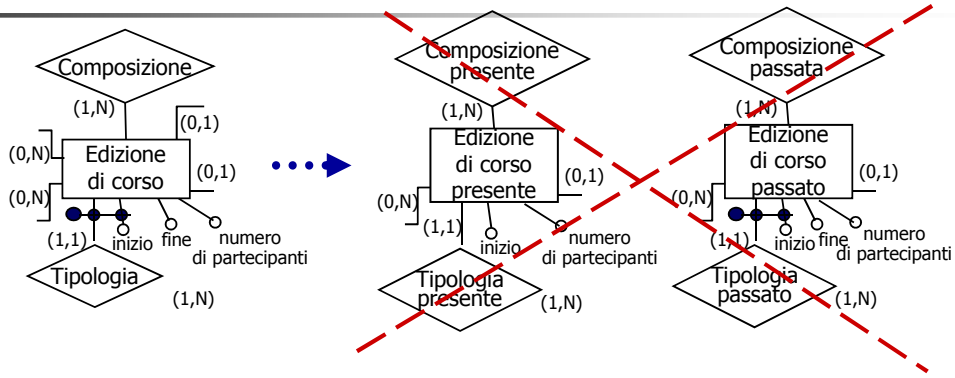
Esempio: partizionamento /a



- L'operazione 5 distingue fra edizioni correnti e passate
- Le relazioni *Partecipazione corrente* e *Partecipazione passata* fanno riferimento a edizioni correnti e passate
- Le relazioni *Docenza corrente* e *Docenza passata* fanno riferimento a edizioni correnti e passate

conviene partizionare l'entità *Edizione di corso* ?

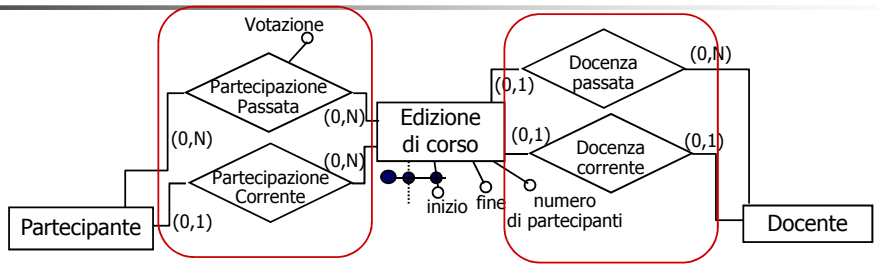
Esempio: partizionamento /b



- Andrebbero duplicate le relazioni *Composizione* e *Tipologia*
- Le operazioni 7 e 8 che non fanno differenza tra edizioni correnti e passate richiederebbero al visita di due entità distinte

Non partizioniamo tale entità

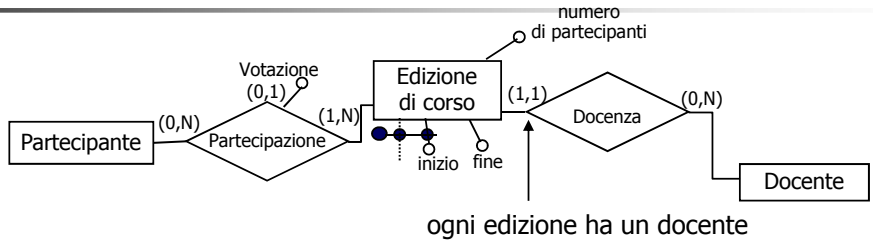
Esempio: accorpamento /a



- Le operazioni in pratica non richiedono la differenza fra i concetti relativi alle relazioni Docenza e Partecipazione corrente e passata
- Occorre trasferire le occorrenze da una relazione all'altra quando una edizione di corso termina
- Il numero medio di istanze di *Partecipazione corrente* è basso (500)

si accorpano i due concetti

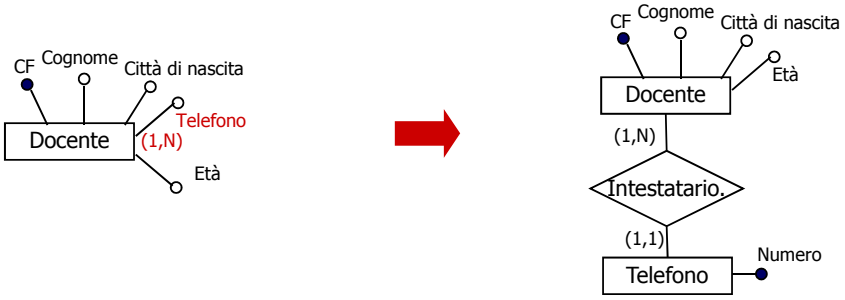
Esempio: accorpamento /b



- L'attributo *Votazione* della relazione *Partecipazione* non si applica alle partecipazioni presenti ed è quindi nullo
  - Lo speco di memoria è solo di 500 x 4 byte = 2000 byte
- Le cardinalità minime dalla parte di Edizione di corso sono 1 perché ogni edizione ha un Docente e almeno un Partecipante

Esempio: attributo multivalore

- Si elimina l'attributo multivalore introducendo l'entità Telefono legata con una relazione uno-a-molti con l'entità Docente



Esempio: identificatori

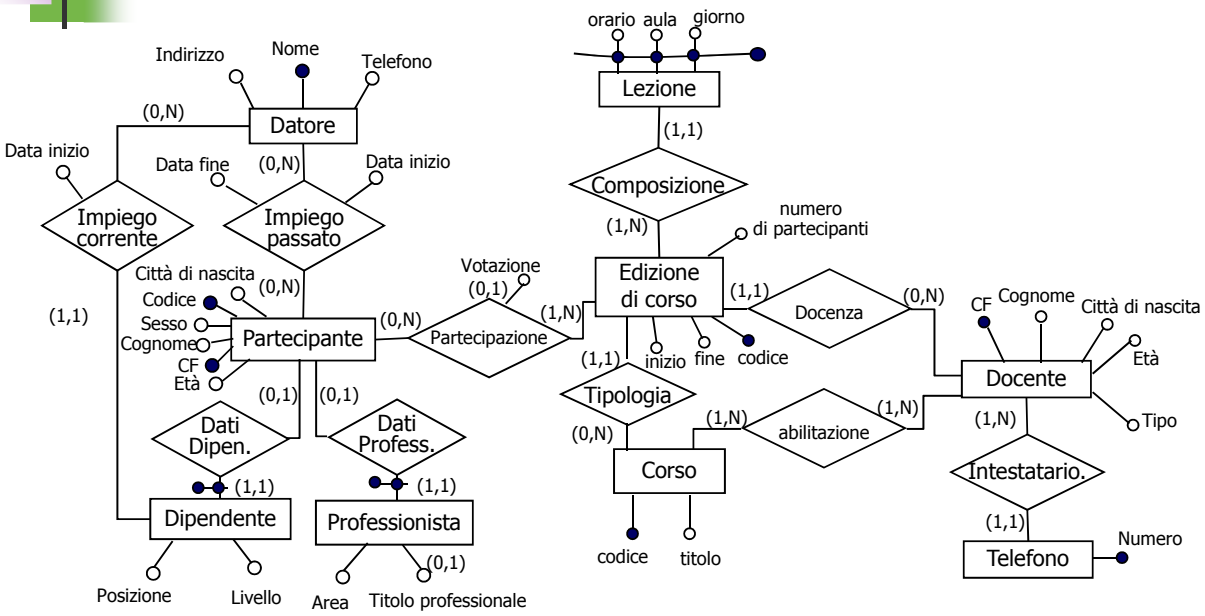
Partecipante

- Codice: 2 byte ←
- CF: 16 byte

Edizione Corso

- Data Inizio e Tipologia Corso (esterno)
- Deve essere usato in Partecipazione e Docenza con molte occorrenze
- E' preferibile inserire un codice ad hoc

Esempio: schema ristrutturato



Esempio: schema relazionale

EdizioneCorso(Codice,DataInizio,DataFine,Corso,Docente,NrPart)

Lezione(Ora,Aula,Giorno,EdizioneCorso)

Docente(CF,Cognome,Età,CittàNascita,Tipo)

Telefono(Numero,Docente), Corso(Codice,Titolo)

Abilitazione(Corso,Docente)

Partecipante(Codice,CF,Cognome,Età,CittàNascita,Sesso)

Partecipazione(Partecipante,EdizioneCorso,Votazione*)

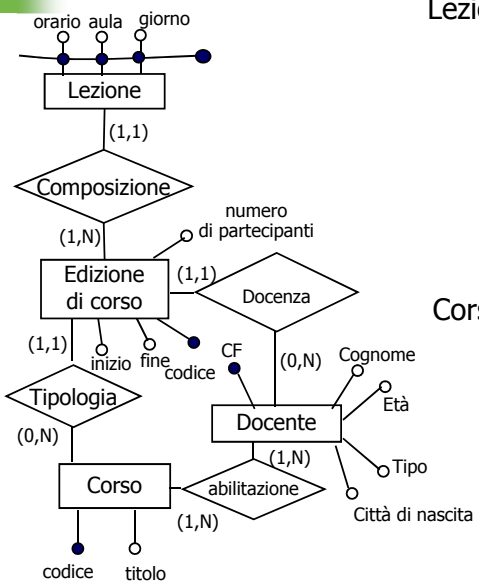
Datore(Nome,Telefono,Indirizzo)

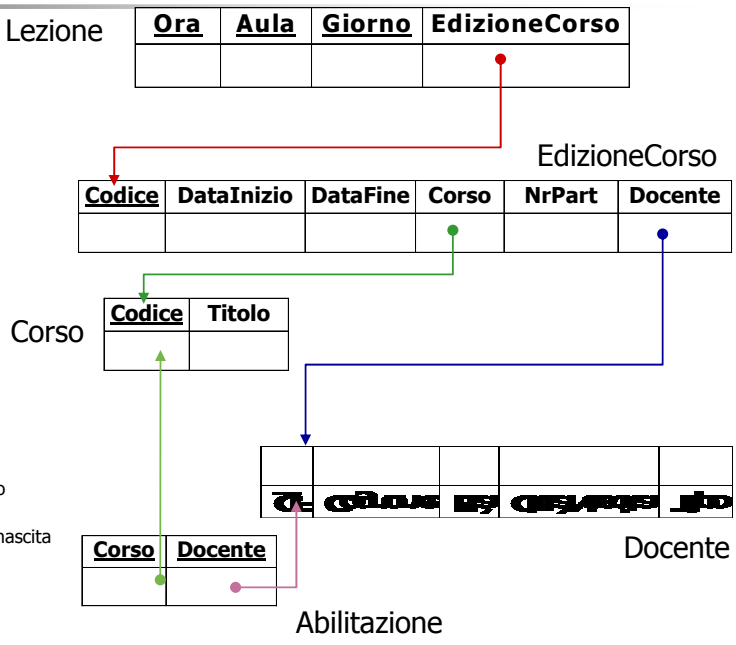
ImpiegoPassato(Partecipante,Datore,DataInizio,DataFine)

Professionista(Partecipante,Area,Titolo*)

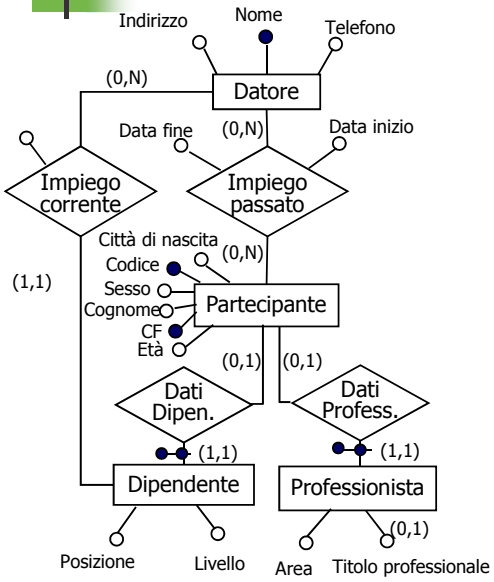
Dipendente(Partecipante,Livello,Posizione,Datore,DataInizio)

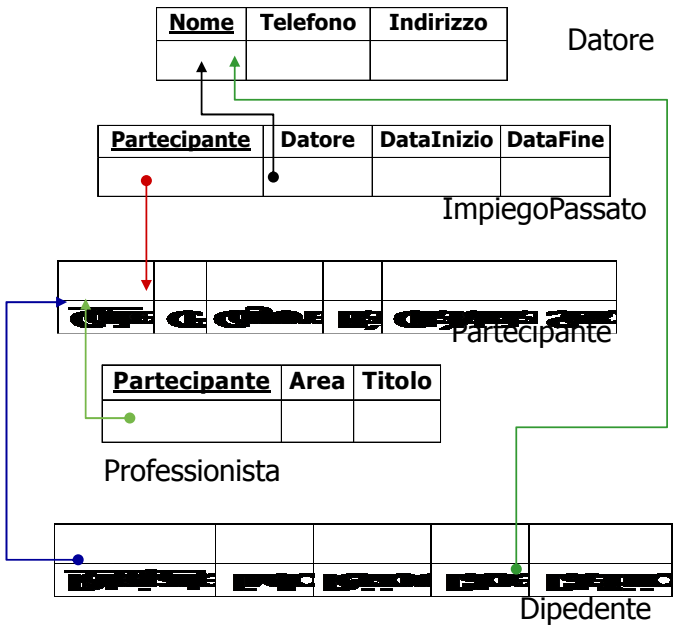
Esempio: schema relazionale /a





Esempio: schema relazionale /b



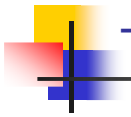


Aspetti avanzati dei sistemi informativi

Grandi quantità di dati

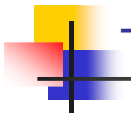
Moderni sistemi informativi e quantità di dati

- I moderni sistemi informativi generano/registrano enormi quantità di dati
  - Archivi di dati delle aziende: prodotti, magazzino, dipendenti, clienti, …
  - L'interazione fra utenti e il web: facebook, twitter, amazon, google, ….
  - I sensori : fotocamere in zone pubbliche, ...
  - Gli esperimenti generano : acceleratori di particelle, sequenziatori (di DNA), ….
- E' in corso una rivoluzione tecnologica per poter sfruttare tale informazione
  - Nuove soluzioni hardware
  - Nuovo software per memorizzare i dati e di elaborarli
  - Software intelligenti (intelligenza artificiale)



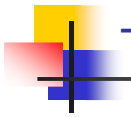
Terminologia

- Data warehouse (trad. magazino dati)
  Sono archivi in cui si concentrano tutti i dati di un'azienda/organizzazione per analizzarli
- OLAP (On line analytical processing)
  Database con funzioni che servono per fare analisi dati basate su semplici statistiche. Gli OLAP possono servire per analizzare i dati presenti in un data warehouse
- Big data
  Le tecnologie per la gestione di enormi quantità di dati. Ci si riferisce a quei casi in cui i dati sono così tanti da dover ricorrere a tecnologie non convenzionali



Terminologia

- Data mining
  Tecnologie per l'estrazione di informazione (non banale, implicita, precedentemente sconosciuta e potenzialmente utile) da una base di dati. Possono essere usate per estrarre informazione dai data warehouse e, rispetto agli OLAP, producono analisi più complesse, non convenzionali. Per i big data si usano le tecnologie del data mining.
- Business intelligence
  E' il data mining applicato al "business"



Terminologia

- Intelligenza artificiale
  Campo dell'informatica che mira a simulare (ed eventualmente migliorare) le capacità più complesse dell'uomo. E' ampiamente usato nel data mining e nei big data.
- Apprendimento automatico
  Campo dell'intelligenza artificiale nel quale gli algoritmi per la risoluzione di un problema sono creati per apprendimento da esempi. Permette di costruire gli strumenti più sofisticati che non possono essere progettati in maniera convenzionale.

Cosa vedremo

Tecnologie che studieremo in questa parte del corso

- Come funzionano data warehouse e on line analytical processing (OLAP)
- Data mining: esempi e tecnologie
- Big data: hardware e software

On Line Analytical Processing

Data Warehouse

- Data Warehouse (magazzino dati)
  - integra in un unico schema globale l'informazione estratta da piu' sorgenti
  - solitamente è interrogabile, ma non modificabile
  - è un'architettura per l'integrazione di database alternativa alla replicazione ai database distribuiti
  - puo' essere
    - ricostruito periodicamente (es. tutte le notti)
    - aggiornato periodicamente (es. tutte le notti)
    - aggiornato immediatamente (es. ogni n transazioni)

OLTP vs OLAP

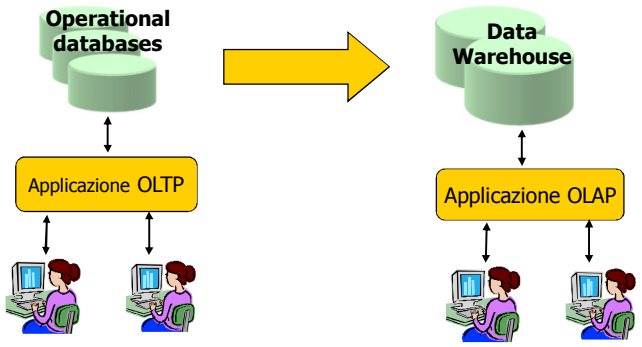
- OLTP (On Line Transaction Processing)
  - Gestione efficiente dei dati in linea (molti operatori)
  - Dati privi di errori e completi
  - Dati relativi allo stato attuale dell'azienda
  - Semplici da realizzare e diffusi capillarmente
  - I sistemi OLTP non sono adatti all'analisi dei dati

OLTP vs OLAP II

- OLAP (On Line Analytical Processing)
  - Pochi utenti dedicati all'analisi dell'andamento dell'azienda
  - I dati storici possono essere utili per la pianificazione e il supporto alle decisioni
    - Capire quali prodotti sono di maggiore successo
    - Stabilire l'efficacia delle promozioni sui prodotti
  - Grandi quantita' di dati
  - Dati spesso scorretti o incompleti

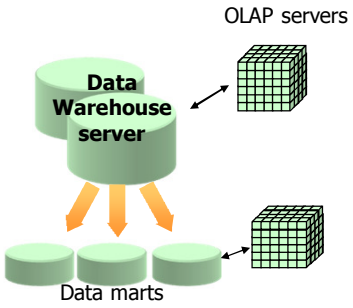
OLTP vs OLAP III

- Gli OLTP sono la fonte principale di informazioni per gli OLAP



Perché OLAP

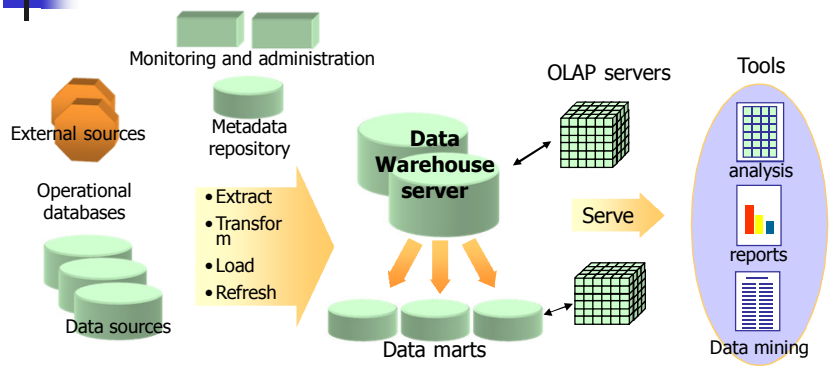
- Definizione di un'interfaccia di analisi dei dati per utenti che svolgono attività di supporto alle decisioni
- Ottimizzazione delle operazioni di analisi invece che di gestione in linea
- Si separa l'ambiente on-line da quello di analisi
- E' l'analisi che avviene in modo interattivo on-line
- Il centro è il magazzino dati (data warehouse - DW)



Esempi di applicazioni OLAP

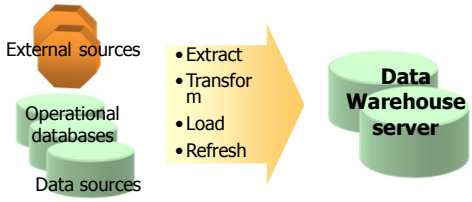
- Vendite
  - analisi e predizione delle vendite
- Marketing
  - analisi delle ricerche di mercato
  - analisi delle promozioni
  - analisi dei consumi
  - segmentazione dei mercati/clienti
- Produzione
  - Pianificazione della produzione
  - Analisi dei difetti

On Line Analytical Processing



- OLAP → Sistemi orientati all'elaborazione e all'analisi dei dati

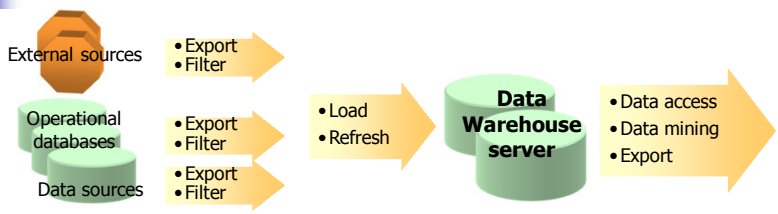
Data warehousing



I sistemi OLTP (i database operativi) sono una fonte dati

- Il data warehouse server trasferisce i dati dai database operativi al suo interno integrandoli (vista unitaria dei dati)
- I dati nella data warehouse sono di tipo storico-temporale
  - L'importazione dei dati è asincrona (disallineamento controllato dei dati) e periodica (riallineamento batch)
  - Occorre garantire la qualità dei dati nella DW

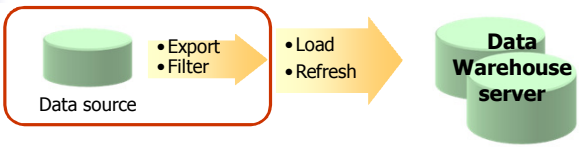
La data warehouse



Fonti dati eterogenee

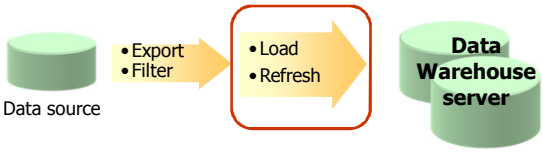
- Raccolte dati non gestite da DBMS (es. i log di un Web server)
- Dati gestiti da DBMS di vecchia generazione (legacy systems)
- Accesso tramite connectors e filtri dati forniti con il DW server
- Creazione/aggiornamento tramite procedure di
  - acquisizione (load)
  - aggiornamento (refresh)

Data filter & export



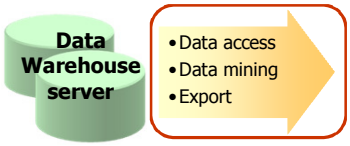
- I data filter controllano la correttezza dei dati prima dell'inserimento nella DW effettuando il data cleaning
  - eliminazione dei dati scorretti con vincoli e controlli su una o più sorgenti dati
- I data export sono i driver che consentono l'estrazione dei dati da una certa sorgente (DBMS, documenti, ...)
  - deve gestire un aggiornamento incrementale (basato sulle modifiche)

Data load & refresh



- Componente di acquisizione dati (load)
  - inserimento iniziale dei dati
  - costruzione delle strutture dati della DW
  - eseguita batch quando la DW non è usata
- Componente di allineamento dati (refresh)
  - aggiornamento incrementale della DW
  - utilizza le modifiche sulle sorgenti dati

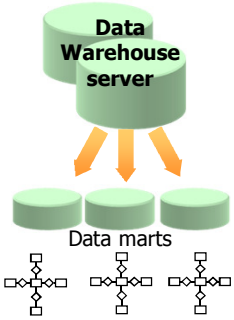
Analisi con la DW



- Componente di accesso ai dati
  - realizza in modo efficiente operazioni complesse di analisi - join fra tabelle, ordinamenti, aggregazioni -
  - operazioni come roll up, drill down e datacube
  - interfaccia di facile uso per gli analisti
- Componente di data mining
  - scoprire in modo automatico regolarità nei dati
- Componente di esportazione dei dati verso altre DW (es. da un DW dipartimentale ad un DW di tutta l'azienda)

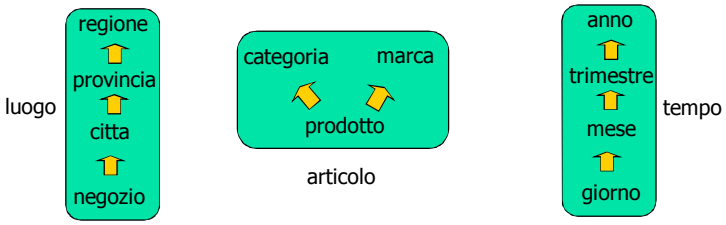
I data mart

- Costruire una DW aziendale completa è complesso
- Si costruiscono schemi per sottoinsiemi semplici dei dati aziendali (data mart) per i quali è chiaro l'obiettivo dell'analisi
  - vendite
  - operazioni di sportello
- I dati di un data mart sono rappresentati secondo uno schema multidimensionale (data cube) e realizzato attraverso uno schema a stella



Visualizzare un data mart: il modello multidimensionale

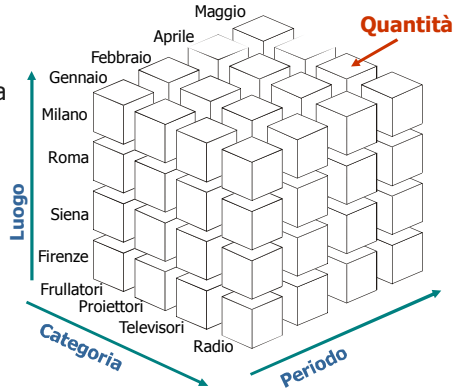
- Il modello multidimensionale è costituito da
  - fatti
    un concetto da analizzare (es. la vendita di un prodotto)
  - misure
    una proprieta' atomica di un fatto (es. il numero delle vendite, l'incasso)
  - dimensioni
    una prospettiva lungo la quale si analizza il fatto (es. il negozio, il mese)
- Le dimensioni sono organizzate in livelli di aggregazione



Data cube

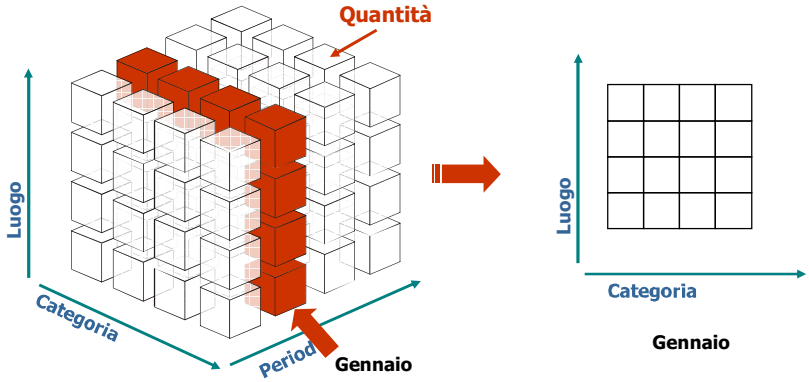
- Data cube
  - con gli assi si indicano le istanze delle dimensioni
  - le celle contengono la misura del fatto

Quantità(Luogo,Categoria,Periodo)



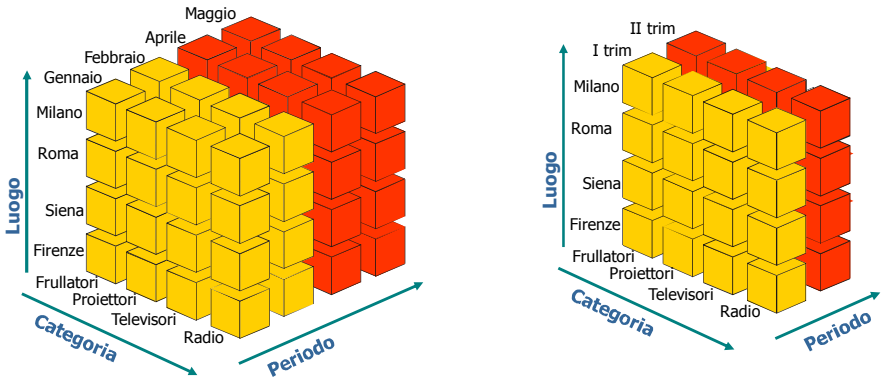
Data cube: operazioni

- Selezione di una vista (slice-and-dice)



Data cube: operazioni II

- Aggregazione dei dati lungo una dimensione salendo nella gerarchia (roll-up)



Data cube: operazioni III

- Al limite l'operazione di roll-up elimina una dimensione
- affinche' sia applicabile, occorre che la misura sia additiva lungo la dimensione prescelta
  - quantita' e incasso sono additive rispetto al periodo
  - scorte non è additiva rispetto al periodo
- L'operazione di drill-down
  - i dati sono disaggregati e resi piu' dettagliati muovendo una dimensione verso il basso della gerarchia
  - è l'opposto di roll-up

Implementazione dei data mart

Il progetto dei data mart

- Il progetto dei data mart richiede la definizione di: fatti, misure e dimensioni di interesse
- Il progetto è un attività costosa che richiede un interazione stretta fra coloro che useranno l'OLAP e i progettisti

L'implementazione dei data mart

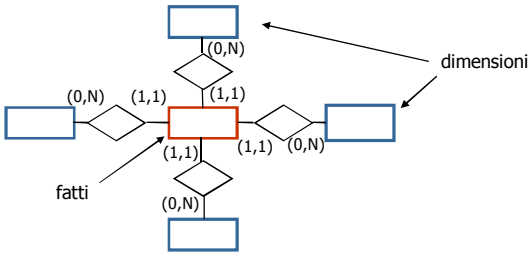
- L'implementazione viene fatta da programamtori/sistemisti per mezzo di strumenti messi a disposizione dagli OLAP

L'analisi dei dei dati

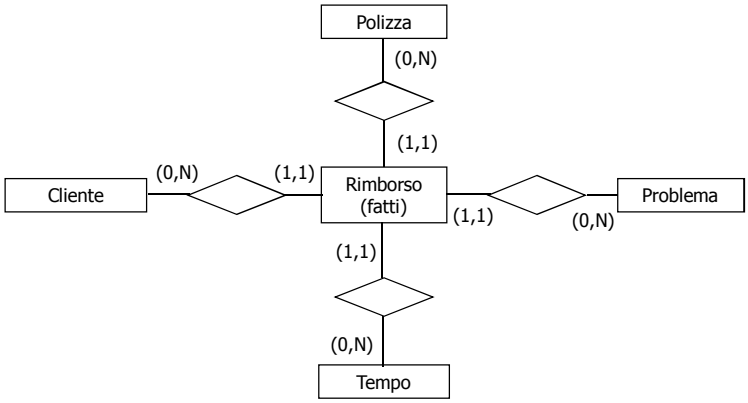
- il gestionale incaricato dell'analisi dei dati usa apposite interfacce grafiche basate sul modello multidimensionale
- Importante: una volta realizzati i data mart, l'interfaccia grafica fornisce i risultati dei data cube al volo, passando da un cubo all'altro immediatamente in seguito alla richiesta dell'utente

Memorizzazione di un data mart: schema a stella

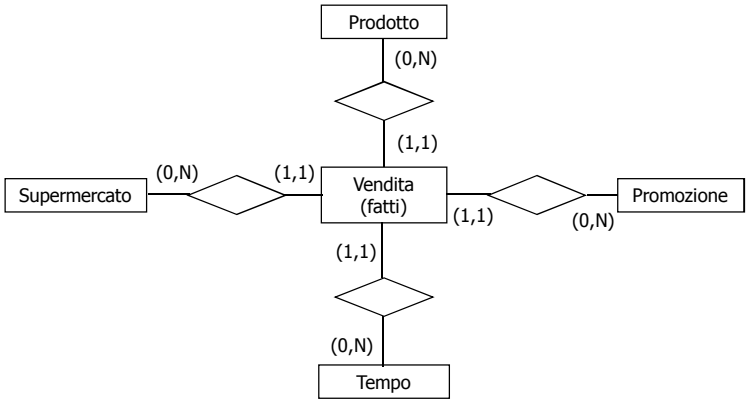
- Corrisponde ad un diagramma ER con struttura semplice
  - Una entità centrale rappresenta i fatti
  - varie entità disposte a raggiera rispetto all'entità centrale rappresentano le dimensioni dell'analisi (≥ 2)
  - relazioni 1:N collegano ogni istanza di fatto ad una sola istanza di ciascuna delle dimensioni



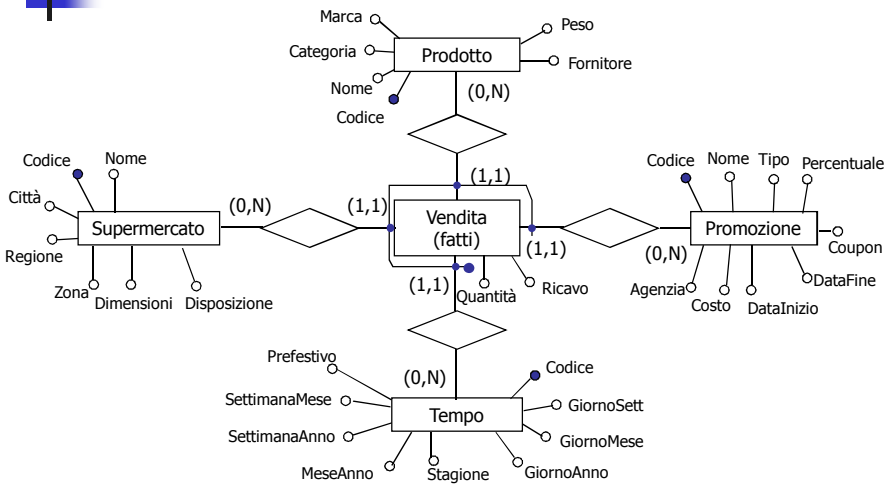
Analisi dei rimborsi (stella)



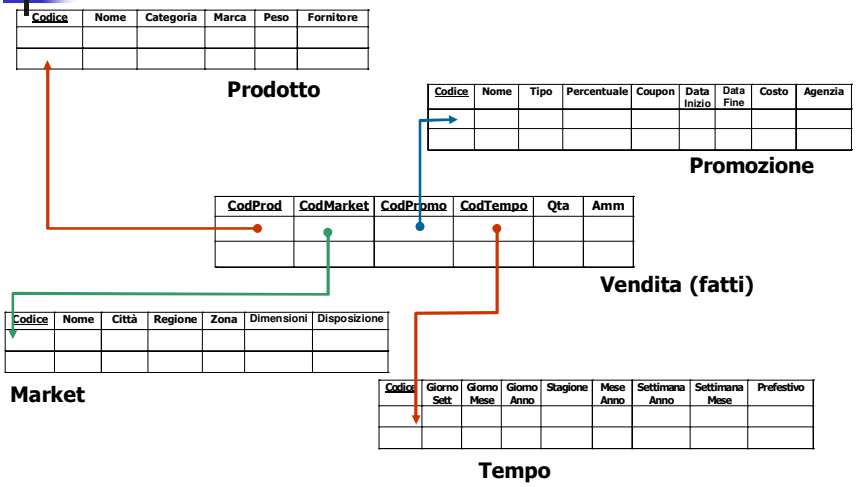
Analisi delle vendite (stella)



Analisi delle vendite (ER)

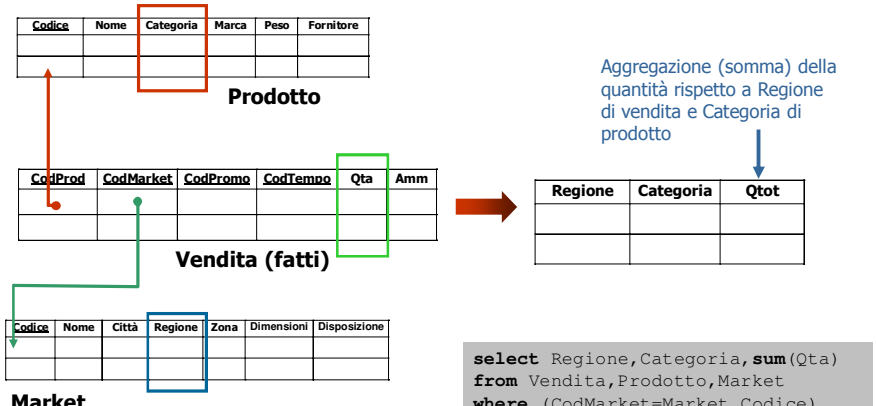


Analisi delle vendite (Logico)



Calcolare i valori contenuti in un data cube

Fissate le dimensioni, il fatto e la misura, il valore contenuti nell data cube possono essere ottenuti come risutato di un'interrogazione SQL con raggruppamento!




```
select Regione,Categoria,sum(Qta)
from Vendita,Prodotto,Market
where (CodMarket=Market.Codice)
  and (CodProd=Prodotto.Codice)
group by Regione,Categoria
```

Data mining

Cos'è il data mining

Esistono diverse definizioni

- Traduzione letterale "estrazione di dati"
- Definizione 1
  Estrazione di informazione (non banale, implicita, precedentemente sconosciuta e potenzialmente utile) da una base di dati
- Definizione 2

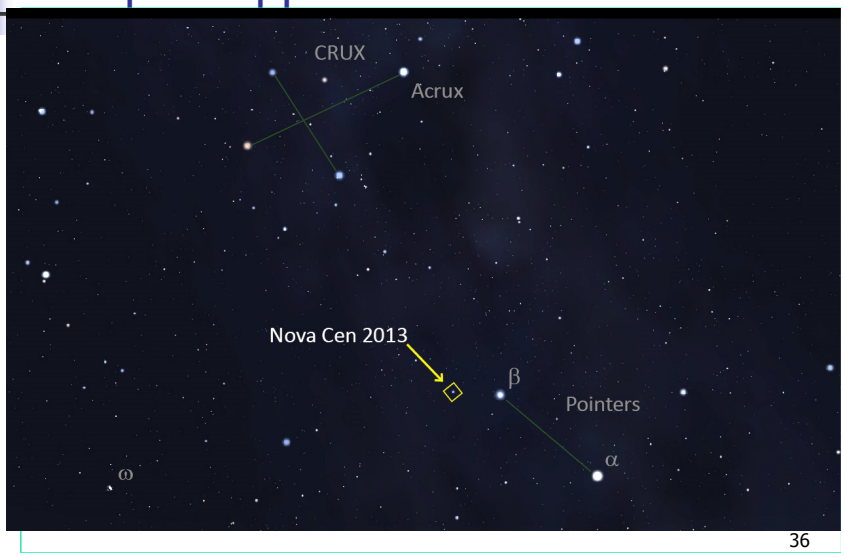
  Esplorazione e analisi di grandi quantità di dati, in modo automatico e semi-automatico) con l'obbiettivo per scoprire pattern (schemi) significativi

Perchè il data mining è utile

Il processo di Data Mining

- è utile perché l'informazione è spesso nascosta a causa di
  - l'enorme quantità di dati: ad. es. il web, le transazioni delle carte di credito, delle compagnie telefoniche, ...
  - la complessità dei dati: ad es. occorre integrare sorgenti di informazioni diverse fra loro, non ci sono noti i fattori che influenzano quello che si cerca, ...
  - la velocità a cui i dati arrivano: ad es. per le carte di credito possono essere decine di transazioni al secondo,...
  - l'eterogeneità dei dati: dati semi-strutturati, strutturati, organizzati in sequenze, in relazioni
  - richieste di analisi non tradizionali: verificare ipotesi, trovare regole, …

Esempi di applicazioni ...

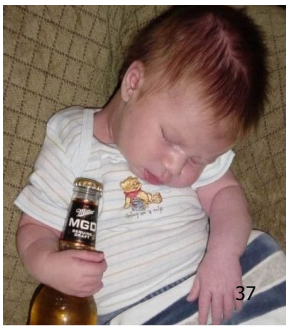


Esempi di applicazioni ...

- **Un ingegnere gestionale che lavora per una catena di supermercati.**
  - Vorrei capire le abitudini dei clienti. Ad esempio, potrei migliorare la disposizione gli oggetti in vendita e aumentare le vendite

- **Fatto!!!**

  Una catena di supermercati negli USA scoprì che i giovani padri, il venerdì sera, comprano insieme la birra insieme a pannolini e li mise vicini nel negozio!



37

Altri esempi ...

Bioinformatica

- Predire la cancerogenità o l'efficacia come cura per una malattia di una certa molecola
  - ad es. usando esempi di molecole cancerogene o efficaci

Applicazioni web

- In un servizio dedicato al cinema (libri, giochi, ..) , suggerire agli utenti nuovi film da vedere (libri da acquistare, giochi da provare,...)
  - ad es. usando le opinioni di altri utenti o il grafo di relazioni fra utenti (vedi ad es. sil sito movielens)
- Su blog e forum individuare gli eventi (ad es. i momenti in cui cambia drasticamente l'argomento di cui si discute) o l'umore degli utenti.
  - a cosa potrebbe servire conoscere eventi e umore?
  - ad es. usando la distribuzione delle parole nel tempo

Applicazioni II

Vendita al dettaglio e marketing

- Scoperta delle associazioni fra le caratteristiche demografiche dei clienti
  - ad es, usando dati ottenuti con le carte fedeltà
- Predizione della risposta alle campagne pubblicitarie
  - ad es. usando esempi di campagne precedenti

Banche

- Uso fraudolento delle carte di credito
  - ad. es. usando la sequenza temporale degli acquisti fatti dall'utente
- Individuare i clienti che stanno per cambiare carta di credito, i clienti fedeli,...
  - ad es. usando i profili degli utenti e la sequenza delle loro operazioni

Fisica

- identificazione di eventi interessanti negli esperimenti fatti con gli acceleratori
  - ad es. usando le informazione provenienti dai sensori

Tipologie di applicazioni

Classificazione delle tipologie di applicazioni

- Le applicazioni del data mining possono essere classificate in differente tipologie
- Ogni tipologia è caratterizzata da caratteristiche comuni che fanno sì che le applicazioni di quel tipo possano essere risolti con algoritmi simili
- Le prossime slide presentano le tipologie più comuni

Tipologie di applicazioni

Analisi delle associazioni

- individuare le regole nascoste del tipo: l'evento A implica l'evento B
  - ad es. chi compra una stampante di solito compra anche il toner

Problemi di classificazione o regressione

- a partire da un insieme di esempi si apprende a classificare un oggetto
  - ad es. si vuol classificare un nuovo utente di un'assiciurazione come utente ad alto rischio o meno: addestra un modello con gli esempi dei vecchi clienti

Problemi di clustering

- Si cerca di organizzare automaticamente gli eventi/oggetti di un database
  - ad es. si vuol identificare le molecole con un proprietà farmacologiche simili

Scoperta di eventi/oggetti anomali (anomaly detection)

- Si cerca di individuare gli eventi, gli oggetti con comportamenti anomali
  - ad es. si vuol individuare le frodi su una carta di credito

Analisi delle associazioni: il problema del carrello della spesa

Il problema del carrello della spesa (market basket)

- Data la registrazione delle "transazioni" di un supermercato:
  - una transazione è un insieme di oggetti acquistati contemporaneamente da un utente

Ricerca delle regole di associazione

- Consiste nell'identificare le regole di implicazione fra gli eventi $H \Rightarrow T$
  - Ad es., *{farina}* $\Rightarrow$ *{lievito}*
    {farina,lievito} $\Rightarrow$ *{latte}*

TID	CID	Data	Prod.	Q.tà
111	201	5/1/05	farina	2
111	201	5/1/05	lievito	1
111	201	5/1/05	latte	3
111	201	5/1/05	carne	6
112	105	7/1/05	farina	1
112	105	7/1/05	lievito	1
112	105	7/1/05	latte	2
113	106	7/1/05	farina	2
113	106	7/1/05	latte	1
114	201	8/1/05	farina	3
114	201	8/1/05	lievito	2
114	201	8/1/05	carne	6
114	201	8/1/05	vino	6

Regole di associazione

Misure associate alle regole e agli insiemi

- Per ogni insieme di oggetti di definisce una misura che valuta quante volte l'insieme compare nel nelle transazioni
  - *supporto(I)*
    misura quante voltel'insieme viene comprato, ad es
    supporto({farina,lievito})=75%

TID	CID	Data	Prod.	Q.tà
111	201	5/1/05	farina	2
111	201	5/1/05	lievito	1
111	201	5/1/05	latte	3
111	201	5/1/05	carne	6
112	105	7/1/05	farina	1
112	105	7/1/05	lievito	1
112	105	7/1/05	latte	2
113	106	7/1/05	farina	2
113	106	7/1/05	latte	1
114	201	8/1/05	farina	3
114	201	8/1/05	lievito	2
114	201	8/1/05	carne	6
114	201	8/1/05	vino	6

Regole di associazione

Misure associate alle regole e agli insiemi

- Per ogni regola $H \Rightarrow T$ i sistemi per l'analisi dell'associazione restituiscono delle misure che servono a valutarne il significato, ad es.
  - *supporto*$(H \Rightarrow T) =$ *supporto*$(H \cup T)$
    misura quante volte la regola è stata vista nell'archivio, ad es
    supporto({farina}⇒{lievito})=75%
  - *confidenza*$(H \Rightarrow T) =$ *supporto*$(H \Rightarrow T)$ / *supporto*(H)
    misura la significatività statistica della regola , permettendo di distinguere le regole utili da quelle banali dove l'ipotesi è sempre vera
    (ad es. *{lievito}⇒{borsa spesa}*)

TID	CID	Data	Prod.	Q.tà
111	201	5/1/05	farina	2
111	201	5/1/05	lievito	1
111	201	5/1/05	latte	3
111	201	5/1/05	carne	6
112	105	7/1/05	farina	1
112	105	7/1/05	lievito	1
112	105	7/1/05	latte	2
113	106	7/1/05	farina	2
113	106	7/1/05	latte	1
114	201	8/1/05	farina	3
114	201	8/1/05	lievito	2
114	201	8/1/05	carne	6
114	201	8/1/05	vino	6

Ricercare le regole d'associazione

Obiettivo:

- Trovare le regole *R* per cui *supporto(R)>min_sup* e *confidenzaR)>min_con*, dove*min_sup* e *min_con* sono due soglie predefinite

Metodo

1. Si cercano gli imsiemi I che hanno un supporto maggore della soglia min_sup
2. Si valutano tutte regole che possone essere derivate dall'insieme I

```
For all the item set S with supporto(S)>min_sup

for each S begin
        Split S into all the pairs A,B such that S=A∪B
        for each A, B begin
                if confidenza(A⇒B)>= min_con
                        then insert A⇒B into the set of the discovered rules
        end
end
```

45

Ricercare gli insiemi più frequenti

- Glli insiemi più frequenti potrebbero essere trovati con l'SQL
- Ma le interrogazioni sarebbero inefficienti perchè richiederebbero N join per insiemi di dimensione N

```
select T1.prod, T2.prod, count(*) as N
from transaction as T1, transactions as T2
where T1.TID=T2.TID and not (T1.prod=T2.prod)
GROUP BY T1.prod, T2.prod
HAVING N >= 3
```

Ricercare gli insiemi più frequenti

Un'altra soluzione non efficiente

- Le transazioni sono scandite dalla prima all'ultima. Il programma tiene traccia, usando una matrice di variabile, delle occorrenze degli insiemi di oggetti

Perché la soluzione non è efficiente?

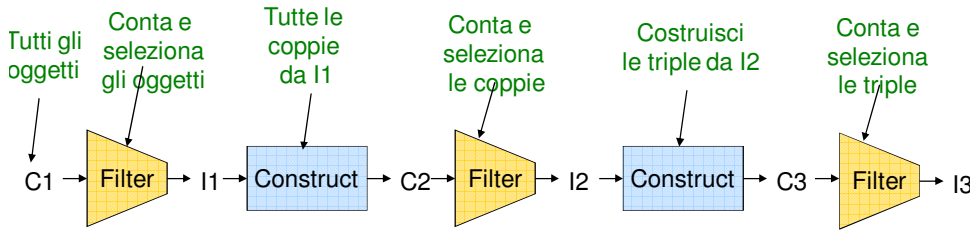
- La matrice di variabili può diventare talmente grande da non poter essere memorizzata in memoria principale (Una catena di negozi potrebbe vendere centinaia di migliaia di oggetti)
- Per questo motivo il programma sopra potrebbe fare continuo swapping

Un algoritmo efficiente: Apriori

- Gli insiemi più frequenti, con frequenza maggiore della soglia min_sup, sono trovati scandendo la tabella delle transazioni n volte
  1. Al primo passo
     - i candidati considerati sono l'insieme C_1 di singoli oggetti. Si scandisce la tabelle per contare le occorrenze degli oggetti in C_1.
     - Da C_1 si selezionano gli elementi I_1 con soglia maggore di min_sup
  2. Secondo passo
     - I condidati consisderati sono le coppie C_2 di oggetti presi da I_1. Si scandisce la tabelle per contare le occorrenze di C_1
     - Da C_2 selezionato le coppie I_2 con supporto maggiore di min_supp
  3. Il passo 2 viene ripetuto per trovare insiemi con numero N crescente di oggetti. I candidati C_N sono costruiti da I_{N-1} L'insieme I_N è il risultato della selezione dei C_N
- Si noti che l'algoritmo funziona perché i candidati considerati ad ogni passo sono molto meno numerosi di tutti i possibili insiemi di oggetti

Un algoritmo efficiente: Apriori

- L'algoritmo funziona perché riduce l'occupazione della memoria: C2, C3, sono molto più piccoli di tutte le possibile coppie, triple …



Esempio

- **Passi ipotetici di A-Priori**
  - C_1 = { {b} {c} {j} {m} {n} {p} }
  - Conta gli oggetti in C_1
  - Seleziona quelli frequenti, ad es I_1 = { b, c, j, m }
  - Genera C_2 = { {b,c} {b,j} {b,m} {c,j} {c,m} {j,m} }
  - Calcola il supporto di C_2
  - Seleziona quelli più frequentj, ad es I_2 = { {b,m} {b,c} {c,m} {c,j} }
  - Genera C_3 = { {b,c,m} {b,c,j} {b,m,j} {c,m,j} }
  - Calcola il supporto di C_3 **
  - Seleziona quelli più frequenti, ad es I_3 = { {b,c,m} }

Classificazione (regressione)

- consiste nell'inferire una proprietà di un pattern sulla base di alcune sue caratteristiche
  - inferire il rischio di incidente per l'utente di una polizza auto, utilizzando dati anagrafici dell'utente e dati sulla macchina assicurata
  - classificare una pagina Web spam o non spam sulla base delle parole contenute nella pagine e su come la pagina è connessa al resto del Web
  - predire l'andamento del bilancio di un'azienda nei prossimi mesi, sulla base dell'andamento passato e di indicatori economici nazionali
  - individuare la presenza di estranei nel filmato di una zona protetta
- la proprietà da inferire può essere
  - un valore numerico qualsiasi (regressione)
  - o appartenere ad un insieme finito (classificazione)
- Quali degli esempi sopra è un esempio di regressione e quali di classificazione?

Un esempio di metodo: classificazione con machine learning

- Un classificatore/regressore spesso viene costruito da esempi
- La costruzione di un classificatore da esempi è un tipico obiettivo del machine learning

Metodo

- Un classificatore è un metodo che implementa una funzione parametrica C_w
- C_w prende in ingresso un vettore di informazioni (features) relative al pattern e restituisce la risposta
- I parametri w vengono stimati su un insieme detto d'addestramento
- L'addestramento può consistere in un problema di attimizzazione

  Occorre minimizzare l'errore commesso dal classificatore sull'inisme di addestramento

Classificazione e machine learning

	viagra	learning	the	dating	nigeria	$spam?$
$\vec{x}_1 = ($	1	0	1	0	0 $)$	$y_1 = 1$
$\vec{x}_2 = ($	0	1	1	0	0 $)$	$y_2 = -1$
$\vec{x}_3 = ($	0	0	0	0	1 $)$	$y_3 = 1$

Esempio: filtro spam – non spam per email

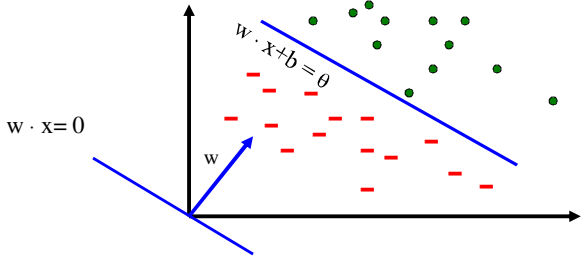
- **Un pattern x ∈ X** (|**X**|= **n** esempi nell'insieme d addestramento)
  - **X è l'insieme di adddestramento**
  - **Un pattern x contiene le parole in un email**
- **Classe y ∈ Y**
  - ***y***: Spam (+1), NoSpam (-1)
- **Obbiettivo : Stimare $C_w(x)$** tale che ***$y = C_w(x)$***

Un esempio di classificatore

- **Classificatore lineare:**

$$C_w(\mathbf{x}) \begin{cases} +1 & \text{if } \mathbf{w^{(1)}\, x^{(1)} + w^{(2)}\, x^{(2)} + \ldots w^{(d)}\, x^{(d)} + b \geq \theta} \\ -1 & \text{otherwise} \end{cases}$$

- **Input:** Vettori $\boldsymbol{x_j}$ e classi $\boldsymbol{y_j}$
  - I vettori $\boldsymbol{x_j}$ contengono valori reli
- **Goal:** Trovare $\boldsymbol{w} = (\boldsymbol{w^{(1)}}, \boldsymbol{w^{(2)}}, \ldots, \boldsymbol{w^{(d)}})$ e un valore $\boldsymbol{b}$
  - tali che siano minimi gli errori sull'insieme di addesrament



La superficce di separazione è lineare

Clustering

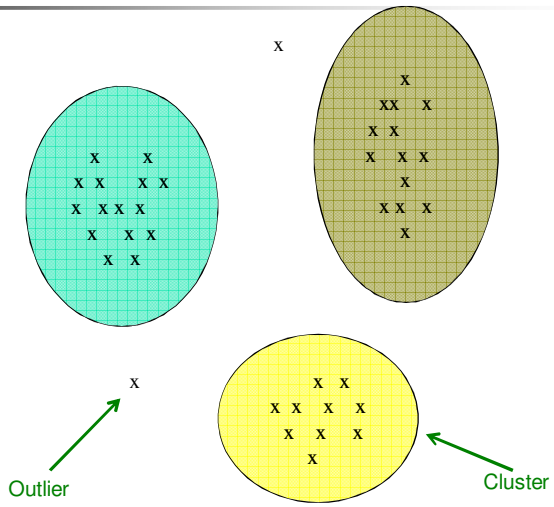
In cosa consiste

- si applica ad un insieme di oggetti (pattern)
- mira a suddividere l'insieme in gruppi (cluster) di oggetti in modo che
  - oggetti nello stesso cluster siano simili
  - oggetti in cluster diversi siano dissimili
- Come per la classificazione, il clustering è spesso affrontato con tecniche provenienti dall'apprendimento automatico:
  - a differenza della classificazione, gli esempi per l'apprendimento includono gli oggetti da analizzare, ma non le classi (i cluster) in cui inserire gli l'oggetto
  - cioè, i cluster non sono predefiniti, ma vengono trovati automaticamente

Esempi di applicazione

- Raggruppamento di molecole con proprietà curative simili
- Raggruppamento di utenti in base al loro comportamento su un sito
- Raggruppamento di utenti in base alle loro caratteristiche sociali ed economiche

Esempio: Clusters & Outliers



Un esempio di metodo: l'algoritmo *k*–means

I cluster sono definiti da punto che funge da centro del cluster e dai punti che il cluster contiene

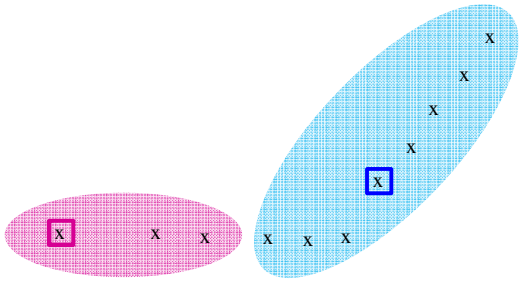
- Si inizia scegliendo un numero di cluster ***k***
- Si inizializza prendendo un punto per cluster
  - **Esempio:** si prendono k punti a caso, oppure se ne prende uno e poi gli altri più lontanti possibile dagli altri…

L'iterazione di *k*–means

1. Ogni punto viene inserito nel cluster più vicino
2. Dopo che tutti i punto sono stati assegnati, si ricalcola il centro del cluster come media dei punti che contiene
3. Si riassegnano i punti ai cluster

- **Si ripetono 2 e 3 fino a convergenza**
  Convergenza: quando punti e centri non si muovono più

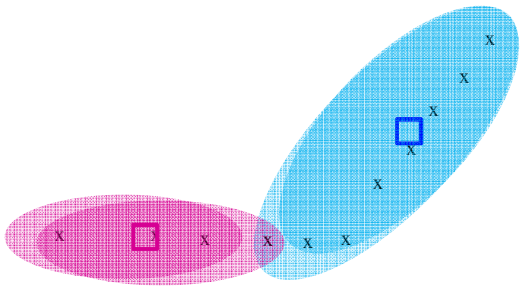
Esempio: l'iterazione di k-means



x … data point
… centroid

Clusters after round 1

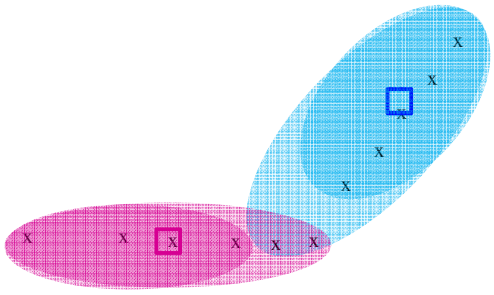
L'algoritmo *k*–means



x … data point
… centroid

Clusters after round 2

L'algoritmo *k*–means



x … data point
… centroid

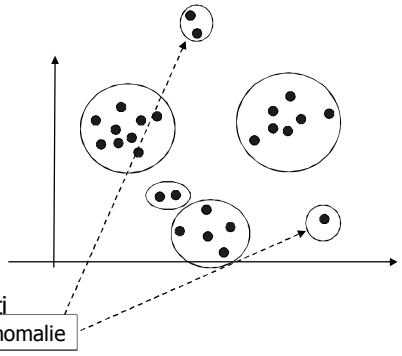
Clusters at the end

Anomaly detection

- mira a individuare gli eventi o gli oggetti che hanno un comportamento diverso dagli altri (detti anche outliers)
  - individuare l'uso fraudolento delle carte di credito
  - individuare i tentati di accesso fraudolento a reti e computer
  - identificazione di anomalie nel funzionamento di un sistema
  - identificazione di anomalie nella salute di una persona
- I metodi per anomaly detection cercano di definire cos'è un comportamento normale da cui deducono cos'è un'anomalia
- Le tecniche sono basate su metodi statistici, visualizzazione grafica, metidi geometrici, clustering

Un esempio di metodo: anomaly detection per mezzo di clustering

- L'idea è quella che applicando un algoritmo di clustering ad un insieme di pattern, i pattern anomali finiscano in cluster piccoli lontani dagli altri cluster
  1. Si applica un algoritmo di clustering
  2. Si cercano i cluster con pochi elementi
  3. Si calcola la distanza dei cluster con pochi elementi dagli altri cluster
     - i pattern anomali sono quelli contenuti nei cluster lontani da tutti gli altri



Il processo di knowledge discovery

Il processo di knowledge discovery è il processo complessivo che dobbiamo realizzare per scoprire informazioni in un insieme di dati usando il data mining. E costituito da 4 fasi

1. Selezione dei dati
   Si selezionano i dati analizzare.
2. Ripulitura dei dati e trasformazione
   Occorre ripulire i dati e prepararli per le operazioni successive. Spesso le tabelle sono denormalizzate e combinate in un'unica tabella
3. Data mining
   Si applicano tecniche di apprendimento automatico, clustering, ...
4. Valutazione, interpretazione, visualizzazione
   Nella maggior parte dei casi i risultati prodotti dal data mining non sono abbastanza affidabili da essere usati direttamente. Essi devono essere valutati e interpretati.

Il processo di knowledge discovery

Realizzare un processo di knowledge discovery per una certa applicazione

- E' costoso e molto complicato
- E' un'attività quasi artigianale: ogni problema ha una soluzione diversa
- Richiede personale con preparazione molto avanzata
- Può portare vantaggi economici importanti

Il ruolo dell'intelligenza artificiale

Quali tecnologie servono?

Il data mining impiega tecnologie molte avanzate provenienti di diversi settori

- Intelligenza artificiale
- Apprendimento automatico
- Architetture di calcolatori
- Sistemi operativi
- Programmazione avanzata
- Metodi matematici
- Sistemi per basi di dati
- ….
- e ovviamente conoscenza del campo di applicazione: bioinformatica, economia,

Intelligenza artificiale e l'apprendimento automatico

L'intelligenza artificiale e l'apprendimento automatico giocano un ruolo fondamentale: le applicazioni richiedono programmi …. Intelligenti

- Individuare le supernove in una foto

  Occorre di distinguere le supernove da stelle variabili, aerei, effetti ottici,….
- Scoprire che i pannolini sono comprati insieme alla birra
  - facile, sì, ma voi avreste cercato questa relazione??
- Cercare molecole simili ad una utile contro una malattia

  In che senso simili?!? Stessa forma 3d, stessa classe, … ?
- Riconoscere i comportamenti scorretti delle automobilisti

  … è già difficile riconoscere una mela in una foto!!

Apprendimento automatico

E quando si sa fare una cosa, ma non si sa come la si fa … fai che il computer apprendere la soluzione

- Usando esempi
- Questa è la soluzione usata dalla natura per gli animali e gli uomini fa

Apprendimento automatico

- Moderno campo dell'informatica
- Crea modelli (programmi) usando insiemi di esempi

69

Perchè serve l'apprendimento automatico: un esempio

Puoi dire come fai a riconoscere una mela?

- Appare come un cerchio rosso

Rosso?

Cerchio?

Perchè serve l'apprendimento automatico: un esempio

Cerchio rosso?

Apprendimento automatico

E quando si sa fare una cosa, ma non si sa come la si fa … fai che il computer apprendere la soluzione

- Usando esempi
  - Ad esempio esempi di mele

In teoria

- Si considera un modello parametrico f_w
- Si adattano i parametri usando esempi

Apprendimento automatico

Tecnologie per l'apprendimento automatico

- Sono moltissime, basati sulla statistica, sistemi a regole, reti neurali, ….
- Si possono usare per problemi di classificazione, clustering, anomaly detection e molto altro

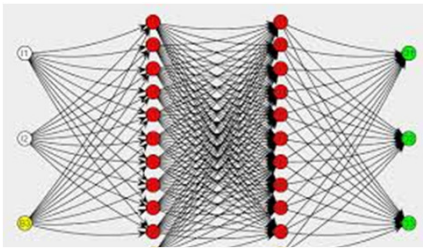
Negli ultimi 10 anni

- c'è stato uno sviluppo enorme in questo settore
- con le artificiali reti neurali profonde (deep networks) si sono raggiunti risultati sorprendenti

Un esempio di un modello di apprendimento automatico

Reti neurali artificiali

- neuroni che propagano il segnale attraverso connessioni
- i neuroni si attivano quando il segnale in arrivo è superiore a una certa soglia
- Il segnale parte da neuroni di ingresso e raggiunge dei segnali di uscita implementando una funzione ingresso-uscita che cambia al variare della forza delle connessioni
- la forza delle connessioni viene aggiustata sulla base di esempi!!!

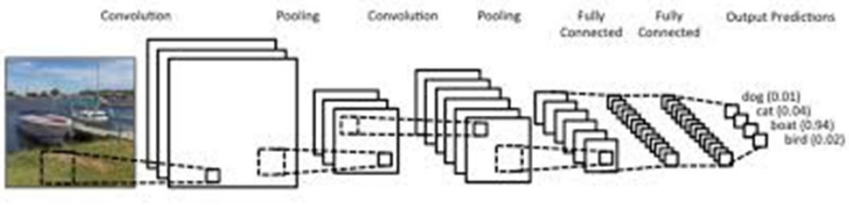


74

Un esempio di un modello di apprendimento automatico

Reti neurali profonde

- una speciale classe di reti neurali
- contengono numerosi strati di neuroni fra gli ingressi e le uscite
- ogni strato può avere caratteristiche diverse



75

Ma funzionano veramente?

Numerosissimi strumenti sono già basati su apprendimento automatico è già funzionano bene

- Il traduttore di Google (deep network)
- Il classificatore delle immagini di Google foto (deep network)
- Il riconoscitore del parlato di Microsoft (deep network)
- I software sulle fotocamere per la localizzazione delle facce/sorriso (apprendimento automatico, altri modelli)
- Il riconoscimento dei cartelli stradati nelle auto (apprendimento automatico)
- L'analisi dati fatta da facebook (deep network, altri modelli)

....

La ricerca va avanti

- investimenti enormi da parte di tutte le grandi aziende (prova a ricercare deep learning market su google per avere un'idea)

Le sfide dei big data

Big data: le sfide

- I big data richiedono enormi capacità di memorizzazione e di elaborazione
  - Good news: i moderni computer sono veloci e hanno dischi grando
- La sfida dello spazio di memoria
  - Itunes memorizza 26M di brani: usando 5M per brano, serve uno storage di 130TB
  - Google memorizza più di 20 miliardi di pagine web: usando 20k per pagina, serve uno storage di 400TB
  - assuming 20KB per page, storing the web requires 400TB
  - Good news: un disco da 5T costa solo 300euros!
- La sfida della capacità di calcolo
  - Leggere le pagine web memorizzate da Google con un solo disco richiederebbe ~4 mesi (i dischi leggono/scrivono a circa30-35 MB/sec): più tempo sarebbe richiesto se si volesse fare qualcosa di più che limitarsi a leggere
  - Good news: leggere le pagine web memorizzate da Google da con 1000 dischi in parallelo (1000 PC) richiede solo 2 minuti!

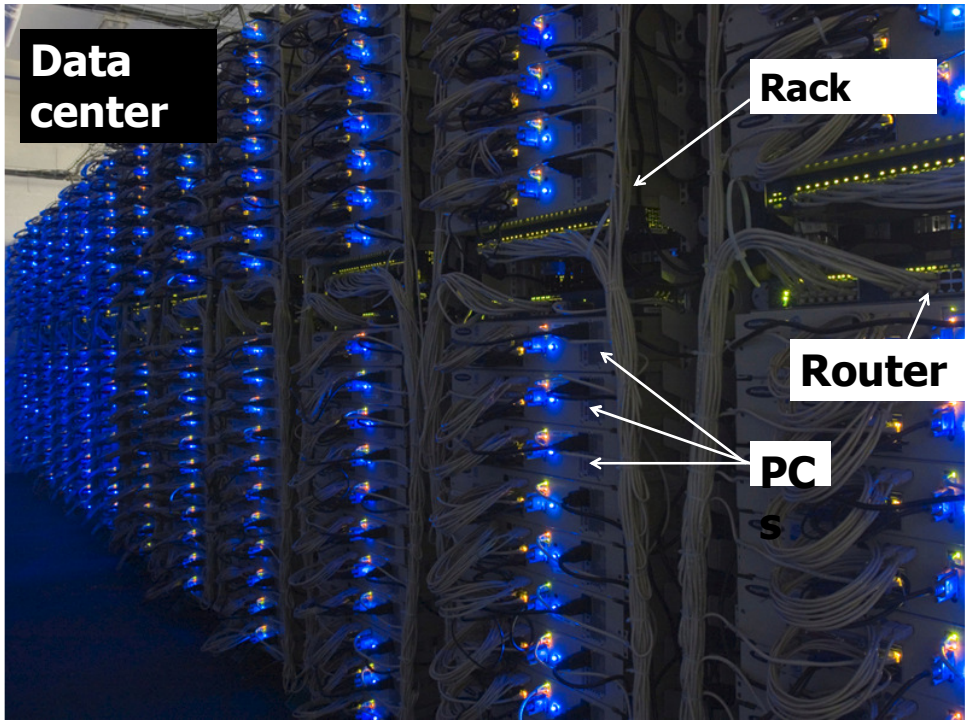
Architetture per i big data

Per i big data si usano singoli computer superveloci?

- no, si usano cluster di decine-centinaia-migliaia di personal computer
- i PC costano poco e messi parallelo sono veloci e permettono di memorizzare quantità di dati enormi

I data center

- contengono migliaia di PC
  - raccolti in armadi (rack)
  - connessi fra loro da router
  - ospitati in capannoni dove la temperatura è controllata
- sono di proprietà di grandi che le usano per le loro attività di calcolo (ad es. Google, Amazon, …) e/o che le affittano ad altri (Amazon, Microsoft, Dada, …)
- con i servizi di cloud è possibili affittare spazio disco, qualche PC, cluster di PC di dimensione fissa o variabile secondo le necessità



Big data: le sfide

Per usare cluster di computer occorre

- nuovi approcci alla gestione dell'affidabilità
- nuovi algoritmi per il calcolo parallelo
- nuovi sistemi operativi e software per lo sviluppo di programmi paralleli

Affidabilità

Vi è capitato che il vostro PC si sia rotto e avete perso tutto??

… pensate se aveste migliaia di PC!!!

- Google ha un milione di server
- Se un se un server si rompe ogni 3 anni (circa 1000 giorni), a Google si rompano 1000 server al giorno!!

Nei cluster di PC

- I dati sono replicati su più server
- Il lavoro da svolgere è diviso fra server
- Alcuni server coordinano il lavoro degli altri:
  se un server si ferma il coordinatore ordina ad un altro di sostituirlo.

Programmazione distribuita

Un esperimento …

- Vuoi costruire un indice ordinato alfabeticamente degli autori dei libri che possiedi con carta e penna. Come fai?
- E ora supponete di voler fare la stessa cosa, ma siete in 1000 e siete nella biblioteca nazionale di Firenze che contiene più di 5 milioni di libri. Come fate?

83

Programmazione distribuita

Da solo …

- per semplificarmi il lavoro, se ho tanti libri, mi preparo un pò di fogli, ognuno per alcune lettere dell'alfabeto
  - Ad es., il foglio degli autori dalla A alla D, uno per quelli dalla E alla H, etc
- Poi prendo in sequenza ogni libro che ho e mi segno gli autori sul foglio corrispondente in sequenza man mano che li trovo
- Quando ho finito, riprendo ciascun foglio e copio su un altro foglio riassuntivo gli autori, questa volta ordinandoli per cognome e nome
- A questo metto insieme tutti i fogli riassuntivi e ho finito.

84

Programmazione distribuita

In 1000 nella biblioteca nazionale di Firenze …

- Per semplificare il lavoro
  - si preparano dei pacchetti di fogli per ciascuno: ad esempio, un pacchetto di 676=26x26 fogli, uno per gli autori con la lettera AA, uno per quelli con la lettera AB, ….
  - Si preparano anche delle scrivanie per raccogliere i fogli con le lettere: ad esempio, una scrivania per la lettera AA, una per la lettera AB, …
- Si divide la biblioteca in 1000 settori e ognuno guarda i libri del suo settore scrivendo sul suo pacchetto di fogli gli autori che trova
- Quando ha finito, prende i suoi 676 fogli e li deposita sulla scrivania corrispondente
- Le prime 676 persone che finiscono vanno ad una scrivania, aspettano che siano arrivati tutti i 1000 fogli e li fondono tutti in un unico foglio riassuntivo dove gli autori sono ordinati
- Poi i fogli riassuntivi sono messi insieme e abbiamo finito

85

Programmazione distribuita

In 1000 nella biblioteca nazionale di Firenze …

- E se c'è qualche «scansafatiche» che dice di lavorare e invece non fa niente…. dobbiamo aspettare che finisca anche lui?
  - No appena c'è qualcuno che non ha più niente da fare va in cerca di uno scansafatiche, un lavoro non completato, e si mette a farlo anche lui: il foglio che useremo sarà quello di che finisce prima fra lui e il supposto scansafatiche

86

Programmazione distribuita

L'esempio precedente

- illustra un possibile funzionamento degli algoritmi di distribuiti
- mette in luce che occorre progettare algoritmi diversi dal solito in ambienti distribuiti!!!

Il compito vero corrispondente all'esempio

- potrebbe essere quello di creare l'indice della parole contenute nel web e di ordinarlo per parola (Google lo costruisce per rispondere alle interrogazioni)
- Le persone rappresenterebbero i processi che in esecuzione sui cluster di PC
- Scrivania e fogli sarebbero la memoria
- Gli scansafatiche sarebbero PC che vanno lenti per motivi difficili da identificare: dischi con errori, cavi malfunzionanti, ….

87

Software per i big data

Esistono vari software che possono aiutare con i big data

- Framework e librerie che aiutano a scrivere programmi distributi come quelli descritti precedentemente: ad es Apache Hadhoop, ispirato al MapReduce di Google
- File system distribuiti (si può memorizzare un file dividendolo fra più PC ed, eventualmente, imponendo che sia replicato per ragioni di affidabilità): ad es, HDFS di Apache
- Sistemi operativi distribuiti (si possono lanciare processi specificando su quale PC devono essere eseguiti)
- ….

88

Alcuni link

- Il cloud di Google con alcuni sistemi di visione basate deep learning
  https://cloud.google.com/vision/
- Weka, uno strumento free per il data mining
  http://www.cs.waikato.ac.nz/ml/weka/
- Il traduttore di Google basato su deep learning
  https://translate.google.it/?hl=it
- Alcune app basate su deep learning
  https://www.tensorflow.org/mobile/