DISPENSA

SGQS modulo rev 01 09/09/2019

MATERIA:
BASI DI DATI

ARGOMENTO:
INTRODUZIONE AI DATABASE

Che argomento faremo insieme?

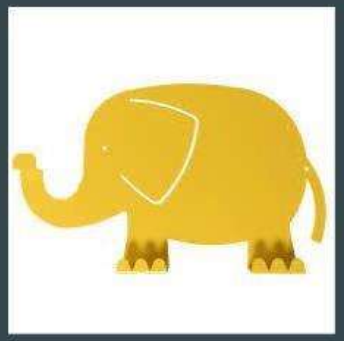
Basi di dati
database relazionali con SQL

Docente:

Simone Leonardi

Informazione

Conoscenza sui fatti
Non schematizzata
Es: “Scoperta nuova specie di elefante giallo”



Dato

- Costituito da simboli
- Schematizzabile
- Es: Holmes 33

Problema:

Senza interpretazione, il dato non è molto utile.

Cosa significa Holmes 33?

Sistema informativo

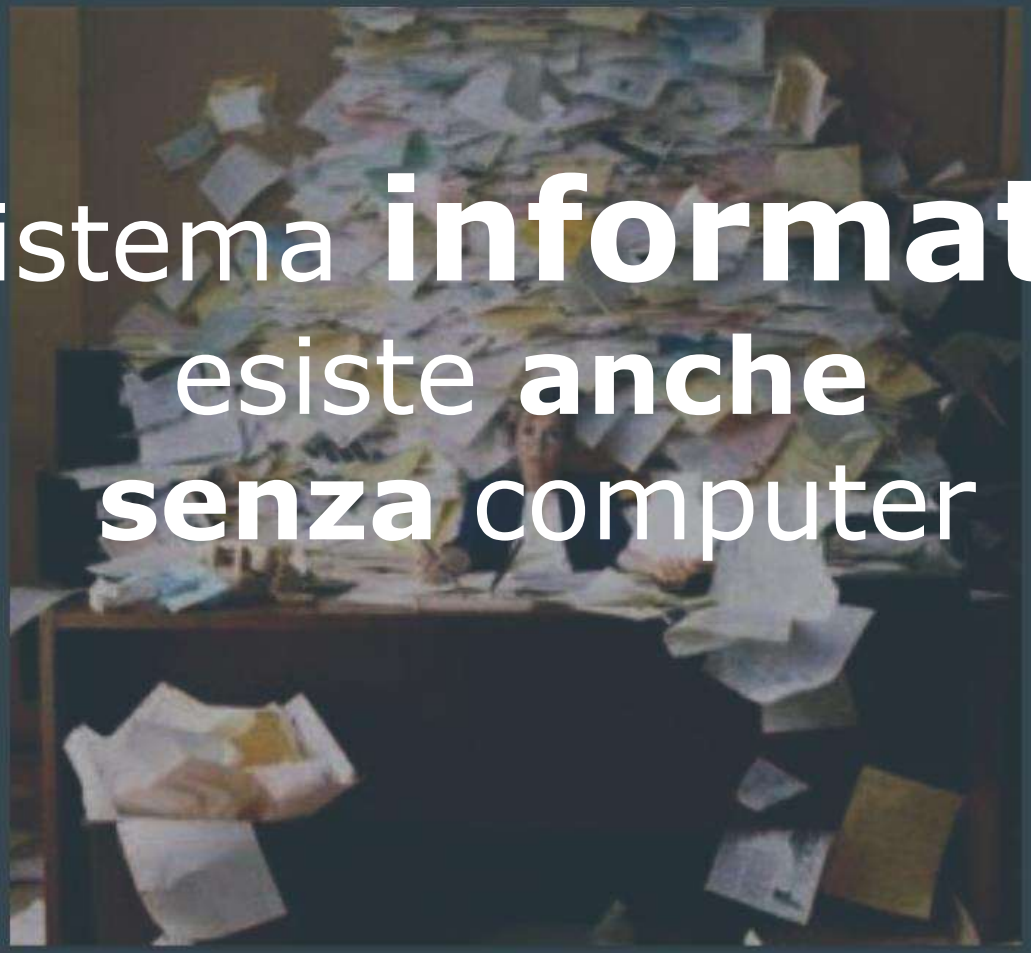
Una **componente** di un'**organizzazione** il cui scopo è quello di **gestire** le **informazioni** utili ai fini dell'organizzazione stessa.

Organizzazione

Gruppo di **persone** formalmente unite per raggiungere uno o più **obiettivi comuni** che individualmente riuscirebbero difficilmente a raggiungere.

Es: enti pubblici, aziende, organizzazioni

Un sistema **informativo**
esiste **anche**
senza computer



Mentre il sistema INFORMATICO è la parte automatizzata

Come gestiamo i dati?

Convenzionale:

- Scrivo e leggo file
- Non distinguo tra dati e applicazione
- Poco efficiente se ho tanti dati
- Relazioni macchinose

Strutturato:

- Software di gestione dati
- Protetti da permessi più dettagliati
- Affidabili in caso di guasti
- Separazione dati applicazione

Impareremo a lavorare con il sistema strutturato, tramite l'utilizzo di un DBMS

DBMS (DataBase Management System): un sistema **software** che è in grado di gestire collezioni **grandi**, **condivise** e **persistenti** di dati, in maniera efficiente e sicura.

DBMS

Funzionalità principali

- Creazione di una base di dati e memorizzazione della stessa
- Accesso in lettura/scrittura ai dati
- Condivisione di dati tra diversi utenti/applicazioni
- Protezione dei dati da accessi non autorizzati
- Affidabilità in caso di guasti (hardware/software)

Utilizzando un DBMS, è possibile implementare un **paradigma di separazione** tra dati ed applicazioni.

Le applicazioni non necessitano di conoscere la **struttura fisica** dei dati (come e dove sono memorizzati su disco) ma solo la loro **struttura logica** (che cosa rappresentano).

Quale DBMS utilizzare?

1. **Oracle**
2. **MySQL**
3. Microsoft SQL Server
4. PostgreSQL
5. DB2
6. Microsoft Access
7. SQLite
8. Teradata
9. SAP Adaptive Server
10. **MariaDB**
11. Hive
12. FileMaker
13. SAP HANA
14. Informix
15. Microsoft Azure SQL
16. Vertica

https://db-engines.com/en/ranking/relational+dbms

Caratteristiche di un DBMS

Efficienza	I dati sono memorizzati utilizzando appositi formati che massimizzano le prestazioni.
Concorrenza	Più utenti possono operare contemporaneamente sugli stessi dati senza interferenze reciproche.
Affidabilità	Le operazioni di modifica non devono mai lasciare il sistema in una situazione inconsistente.
Sicurezza	Gli utenti possono accedere ai dati secondo delle regole definite con un sistema di permessi.

Efficienza

- I DBMS forniscono **strutture** per organizzare i dati all'interno dei file e per **supportare** le operazioni di **ricerca,**
  aggiornamento e cancellazione.
- In genere, si tratta di strutture dati ad albero (es. B-tree) o tabelle di hash.
- Tali strutture sono utilizzate per creare degli **INDICI**.
  Utilizzando opportunamente un indice, la complessità
  computazionale di una ricerca viene ridotta da O(N) a O(log(N)).

Efficienza

Le interrogazioni effettuate vengono tradotte in una sequenza di operatori algebrici utilizzati per **ottimizzare** l'accesso ai dati.
Tale notazione prende il nome di **algebra relazionale**.

$$\sigma_{A \wedge B}(R) = \sigma_A(\sigma_B(R)) = \sigma_B(\sigma_A(R))$$

L'ottimizzazione considera le proprietà matematiche dell'algebra relazionale ed alcune statistiche sulla distribuzione dei dati.

Concorrenza

In molti sistemi informatici è fondamentale gestire operazioni concorrenti di accesso ai dati.
Visa processa in media

1667 transazioni ogni secondo.

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

Concorrenza

OP1
Leggi x
Calcola $x - 100$
Scrivi x

OP2
Leggi x
Calcola $x - 80$
Scrivi x

Esecuzione	Valore di x
OP1 Leggi x	120
OP2 Leggi x	120
OP1 Calcola $x - 100$	120
OP2 Calcola $x - 80$	120
OP1 Scrivi x	20
OP2 Scrivi x	40

Per prevenire tali situazioni, i DBMS implementano algoritmi di controllo della concorrenza affinché le operazioni sui dati, o transazioni, eseguite in concorrenza producano lo stesso risultato di un'esecuzione seriale.

Concorrenza

Il lock manager è la componente del DBMS responsabile della gestione dei lock alle risorse del database.

OP1	OP2
Blocca x	Blocca x
Leggi x	Leggi x
Calcola $x - 100$	Calcola $x - 80$
Scrivi x	Scrivi x
Sblocca x	Sblocca x

Affidabilità

Alcune operazioni sono particolarmente delicate e devono essere gestite secondo la regola del tutto o niente!

Esempio

Trasferimento di denaro (100 $) dal conto x al conto y.

OP1 $x = x - 100$

CRASH!

OP2 $y = y + 100$

I DBMS devono fornire appositi strumenti per annullare le operazioni non completate e fare rollback dello stato del sistema.

Affidabilità

In molti casi i DBMS mettono a disposizione appositi strumenti ed algoritmi per garantire la persistenza dei dati anche in presenza di **malfunzionamenti** hardware/software. Il controllore di affidabilità utilizza un log, nel quale sono indicate tutte le operazioni svolte dal DBMS.

Grazie al log, è possibile fare do/undo delle operazioni.

Affidabilità

Una transazione è l'insieme di una o più operazioni di lettura/scrittura eseguite su un database dal DBMS.

Proprietà ACID di un sistema transazionale	
Atomicità	La transazione deve essere eseguita secondo la regola del tutto o niente.
Consistenza	La transazione deve lasciare il database in uno stato consistente, i vincoli di integrità dei dati non devono essere violati.
Isolamento	L'esecuzione di una transazione deve essere indipendente dalle altre.
Durabilità	L'effetto di una transazione conclusa con successo non deve essere perso.

Sicurezza

La maggior parte dei DBMS implementa politiche di controllo degli accessi ai dati mediante sistemi di permessi.

Utente	Operazione	Dato	Permesso
Utente x	Lettura	Stipendio x	Consentito
Utente x	Scrittura	Stipendio x	Negato
Utente x	Lettura	Stipendio y	Negato

Scalabilità

La scalabilità orizzontale è la possibilità di gestire grandi moli di dati aumentando il numero di nodi usati per lo storage e realizzando un database distribuito. Una funzione di sharding determina la politica di distribuzione dei dati tra i nodi.

Funzionalità di un database distribuito

- Load-balancing
- Gestione delle repliche dati

Come gestire la consistenza delle repliche dati in presenza di partizionamenti della rete e perdita di messaggi?

Modello Relazionale

Nel modello relazionale, i dati sono organizzati in relazioni o tabelle.

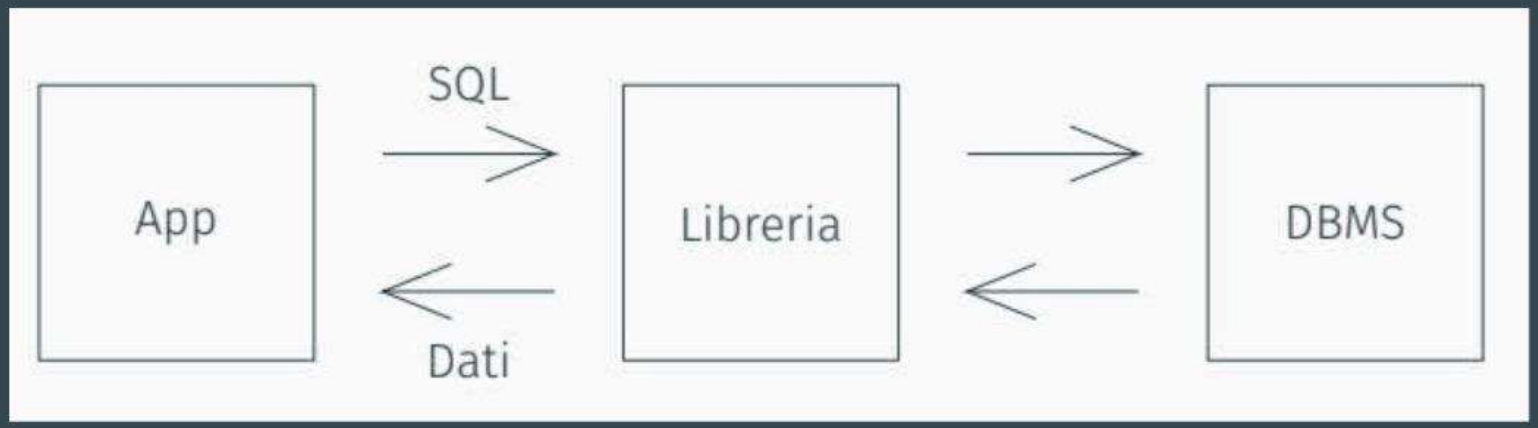
Per esempio, consideriamo una tabella che gestisce le informazioni relative agli impiegati di una azienda.

Nome	Cognome	Dipartimento	Ufficio	Stipendio	Città
Mario	Rossi	Amministrazione	10	45	Milano
Carlo	Bianchi	Produzione	20	36	Torino
Giovanni	Verdi	Amministrazione	20	40	Roma
Franco	Neri	Distribuzione	16	45	Napoli
Carlo	Rossi	Direzione	14	80	Milano
Lorenzo	Gialli	Direzione	7	73	Genova
Paola	Rosati	Amministrazione	75	40	Venezia
Marco	Franco	Produzione	20	46	Roma

Interazione con un DBMS

Le applicazioni che si interfacciano con un DBMS:

- integrano query SQL all'interno del loro codice;
- utilizzano opportune librerie per gestire la connessione al DBMS.



Vantaggi nell'uso di un DBMS

Quando usare un DBMS in un progetto software?

- Necessità di gestire grandi volumi di dati;
- Necessità di costruire sistemi con molte operazioni sui dati;
- Necessità di condividere dati, fornendo l'accesso a diversi utenti;
- Necessità di garantire la persistenza dei dati anche a fronte di possibili guasti
- Necessità di implementare meccanismi di sicurezza per l'accesso ai dati in un ambiente multiutente.

Svantaggi

Quando non usare un DBMS in un progetto software?

- In alcuni sistemi con richieste di efficienza sull'elaborazione, l'overhead introdotto dal DBMS può essere eccessivo;
- Bisogna considerare il costo per l'acquisto del DBMS, la formazione del personale, l'amministrazione del database;
- Le applicazioni di dimensioni ridotte, con un solo utente e con pochi dati da gestire potrebbero non richiedere un DBMS.

Livelli di astrazione

Un DBMS può essere visto come un'architettura software a tre livelli.

- **Livello esterno** $\rightarrow$ descrive come i dati appaiono per un utente;
- **Livello logico** $\rightarrow$ descrive l'organizzazione logica dei dati;
- **Livello fisico** $\rightarrow$ descrive come i dati sono memorizzati sulla memoria secondaria.

Proprietà dei livelli

Proprietà dei livelli in un DBMS:

- **Indipendenza fisica** $\rightarrow$ interagire con il modello logico in modo indipendente dallo schema fisico;
- **Indipendenza logica** $\rightarrow$ interagire con il livello esterno in modo indipendente dallo schema logico dei dati.

Modello logico

1. Insieme di concetti per strutturare i dati relativi ad un certo dominio d'interesse.

Esempi

- Record a struttura fissa
- Record a struttura variabile

Modello logico

2. Insieme di regole per modellare eventuali vincoli e restrizioni sui dati.

Esempi

- Il voto d'esame è un numero intero compreso tra 18 e 30
- Il codice fiscale è una stringa di 16 caratteri
- La data di nascita è espressa nel formato *GG-MM-AAAA*

Modello logico

I DBMS possono differire sulla base del modello logico disponibile.

Modelli logici

- **Modello relazionale**
- Modello gerarchico
- Modello reticolare
- Modello ad oggetti
- Approcci NoSQL

Modello relazionale

È stato proposto nel 1970 da **Edgar Frank Codd**, ricercatore presso il San Jose Research Laboratory dell'IBM.

Garantisce l'indipendenza fisica e logica tra i livelli.

È intuitivo poiché basato su nozioni di algebra di base.

I DBMS basati sul modello relazionale sono detti **RDBMS**.

Definizione informale

I dati sono organizzati in record di dimensione fissa e divisi in tabelle dette relazioni.

Nome	Cognome	Dipartimento	Ufficio	Stipendio	Città
Mario	Rossi	Amministrazione	10	45	Milano
Carlo	Bianchi	Produzione	20	36	Torino
Giovanni	Verdi	Amministrazione	20	40	Roma

- Colonne $\rightarrow$ **Attributi**
- Intestazione $\rightarrow$ **Schema**
- Righe $\rightarrow$ **Istanze**

Definizione informale

- Nome della relazione: `DIPENDENTI`
- Attributi: `Nome, Cognome, Dipartimento, Ufficio, Stipendio, Città`
- Schema: `DIPENDENTI (Nome, Cognome, Dipartimento, Ufficio, Stipendio, Città)`
- Istanza: `<Mario, Rossi, Amministrazione, 10, 45, Milano>`

Vincoli

Ordine dei dati

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

Dati della relazione

- Non possono esistere attributi uguali
- Non possono esistere righe uguali
- I dati di una colonna devono essere omogenei

Relazione senza istanze

È possibile avere una relazione senza istanze.

Nome	Cognome	Dipartimento	Ufficio	Stipendio	Città

Il viceversa è invece impossibile.

???	???	???	???	???	???
Mario	Rossi	Amministrazione	10	45	Milano
Carlo	Bianchi	Produzione	20	36	Torino
Giovanni	Verdi	Amministrazione	20	40	Roma

Dominio

Ogni attributo dispone di un **dominio** che ne definisce l'insieme dei valori validi.

Esempi

- Dom(Nome) = `VARCHAR`
- Dom(Cognome) = `VARCHAR`
- Dom(Stipendio) = `INTEGER`

Prima forma normale

Una relazione si dice in **prima forma normale** (1NF) se tutti gli attributi sono definiti su domini atomici e non su domini complessi.

Nome	Cognome	Dipartimento	Altre informazioni
Mario	Rossi	Amministrazione	10, 45, Milano
Carlo	Bianchi	Produzione	20, 36, Torino
Giovanni	Verdi	Amministrazione	20, 40, Roma

Riferimenti

Nel modello relazionale, i riferimenti tra dati in relazioni differenti sono espressi mediante valori.

Cliente

IdConto	Nome	Cognome	Data Nascita
0001	Mario	Rossi	12/09/1984
0002	Carlo	Bianchi	29/04/1978

Versamento

IdConto	Data	Importo
0001	07/08/2016	100
0002	15/12/2017	250

Informazioni incomplete

Che accade se il valore di un attributo per una certa istanza, o tupla, non è noto/è inesistente/è senza informazione?

Le informazioni mancanti sono etichettate con il valore **NULL**.

Nome	Cognome	Dipartimento	Ufficio	Stipendio	Città
Mario	Rossi	Amministrazione	NULL	45	Milano
Carlo	Bianchi	Produzione	20	36	Torino
Giovanni	Verdi	Amministrazione	20	40	Roma

Per definizione, **NULL** $\neq$ **NULL**.

Informazioni incomplete

È fondamentale limitare il numero di valori **NULL** presenti in una relazione!

Nome	Cognome	Dipartimento	Ufficio	Stipendio	Città
Mario	NULL	Amministrazione	NULL	NULL	Milano
NULL	Bianchi	Produzione	NULL	36	NULL
NULL	NULL	NULL	20	NULL	Roma

Vincoli di integrità

Non tutte le istanze di una relazione possono considerarsi lecite!

Nome	Cognome	Dipartimento	Ufficio	Stipendio	Città
Mario	Rossi	Amministrazione	10	**-15**	Milano
Carlo	Bianchi	Produzione	20	36	Torino
Giovanni	Verdi	Amministrazione	20	40	Roma

Vincoli di integrità

Cliente

IdConto	Nome	Cognome	Data Nascita
0001	Mario	Rossi	12/09/1984
0002	Carlo	Bianchi	29/04/1978

Versamento

IdConto	Data	Importo
0001	07/08/2016	100
0003	15/12/2017	250

Vincoli di integrità

Un **vincolo** è una funzione booleana che associa ad una istanza di una base di dati un **valore di verità** (true/false).

Un'istanza si dice lecita se soddisfa tutti i vincoli.

- Vincoli intrarelazionali
  - Vincoli di tupla
  - Vincoli di chiave
- Vincoli interrelazionali

Vincoli di tupla

I vincoli di tupla esprimono condizioni su ciascuna tupla, considerata singolarmente. Possono essere espressi mediante espressioni algebriche o espressioni booleane.

Esempio

1. $\text{Voto} \geq 18 \wedge \text{Voto} \leq 30$
2. $\neg(\text{Lode} = \text{Sì} \wedge \text{Voto} \neq 30)$

Corso	Studente	Voto	Lode
C0001	S0001	42	No
C0001	S0002	18	Sì

Vincoli di chiave

Una **chiave** è un insieme di attributi che consente di identificare in maniera univoca le tuple di una relazione.

Matricola	Nome	Cognome	Ufficio	Stipendio	Città
0001	Mario	Rossi	10	45	Milano
0002	Carlo	Bianchi	20	36	Torino
0003	Giovanni	Verdi	20	40	Roma

- Non esistono due impiegati con la stessa matricola.
- Data la matricola di un impiegato, è possibile risalire a tutti i suoi dati (nome, cognome, ufficio, stipendio, città).

Superchiave

Un sottoinsieme $\mathcal{K}$ di attributi di una relazione è una **superchiave** se non contiene due tuple distinte con gli stessi valori per $\mathcal{K}$.

Matricola	Nome	Cognome	Ufficio	Stipendio	Città
0001	Mario	Rossi	10	45	Milano
0002	Carlo	Bianchi	20	36	Torino
0003	Giovanni	Verdi	20	40	Roma

- {Matricola} è una superchiave.
- {Ufficio} non è una superchiave.
- {Matricola, Nome} è una superchiave.

Chiave

Una **chiave** di una relazione R è una superchiave minimale di R.

Matricola	Nome	Cognome	Ufficio	Stipendio	Città
0001	Mario	Rossi	10	45	Milano
0002	Carlo	Bianchi	20	36	Torino
0003	Giovanni	Verdi	20	40	Roma

- {Matricola} è una chiave.
- {Ufficio} non è una chiave.
- {Matricola, Nome} non è una chiave.

Chiave primaria

Una **chiave primaria** è una chiave di una relazione su cui non sono ammessi valori **NULL**. Gli attributi di una chiave primaria sono in genere indicati con una sottolineatura.

Matricola	Nome	Cognome	Ufficio	Stipendio	Città
0001	Mario	Rossi	10	45	Milano
0002	Carlo	Bianchi	20	36	Torino
0003	Giovanni	Verdi	20	40	Roma

`DIPENDENTI (Matricola, Nome, Cognome, Ufficio, Stipendio, Città)`

Chiave primaria

Ogni relazione deve disporre di una chiave primaria!

Nel caso in cui tutte le chiavi presentino dei valori `NULL` è sufficiente aggiungere un identificativo numerico progressivo.

La chiave primaria viene utilizzata per:

- accedere a ciascuna tupla della base di dati in maniera univoca;
- correlare dati tra relazioni differenti.

Una chiave primaria può essere composta da più attributi.

Vincoli interrelazionali

Nel modello relazionale, una base di dati può essere composta da molte relazioni collegate tra loro. Collegamenti tra relazioni differenti sono espresse mediante valori comuni in attributi replicati.

Studente

Matricola	Nome	Cognome
S0001	Mario	Rossi
S0002	Carlo	Bianchi
S0003	Giovanni	Verdi

Valutazione

Verbale	**Studente**	Materia	Voto
V0001	S0001	M0001	28
V0002	S0001	M0002	30
V0003	S0002	M0001	25

`Studente.Matricola` $\rightarrow$ `Valutazione.Studente`

Vincoli interrelazionali

Un vincolo di integrità referenziale (**foreign key**) fra gli attributi $\mathcal{K}$ di una relazione R_1 e un'altra relazione R_2 impone ai valori su $\mathcal{K}$ in R_1 di comparire come valori della chiave primaria di R_2.

Ogni riga della tabella referenziante si collega al massimo ad una riga della tabella referenziata, sulla base dei valori comuni nell'attributo/negli attributi replicati.

Può accadere che un'operazione di aggiornamento su una relazione causi violazioni di vincoli di integrità su altre relazioni. Soluzioni:

- non consentire l'operazione;
- eliminazione a cascata;
- inserimento di valori `NULL`.

Vantaggi e svantaggi

Vantaggi

- È un modello intuitivo, basato su proprietà algebrico/logiche
- Garantisce l'indipendenza dallo schema fisico
- Riflessività, ovvero le meta-informazioni di una relazione sono gestite a loro volta attraverso relazioni

Svantaggi

- Poca flessibilità, infatti tutte le istanze di una relazione devono possedere la stessa struttura
- Ridondanza dei dati causata dai vincoli