Basi di Dati

DaveRhapsody

4 Marzo 2020

Indice

Introduzione

Un database, o base di dati, o db (che scriverò d'ora in poi) è un insieme di dati, tipo deposito, per qualsiasi genere di uso, sia aziendale, che personale. I suddetti dati sono inseriti, letti, e soprattutto organizzati secondo certe regole.

Alcuni esempi:

- Agende telefoniche
- Studenti di una classe o scuola
- Qualsiasi insieme generico in cui ogni elemento differisce da un altro secondo delle linee guide (campi) che si decidono alla base.

Da Linguaggi di Programmazione sono state viste le Struct, o Record, che sono delle strutture per i dati statiche, concrete ed eterogenee, aventi più campi non necessariamente dello stesso tipo.

Bene, un DB è un array di record, in cui bisogna garantire integrita, consistenza e NON ridondanza dei dati.

0.1 Interazione tra campi

Tra loro i campi di questi record possono interagire, nel senso che partendo dal valore di un determinato campo, si può ricavare il valore di un altro campo. Esempio? Il codice fiscale, che è ricavato da una formula che non ricordo mai nella vita forever MA prendendo come dati il nostro nome, cognome, etc.

Questo sarà un campo calcolato

0.2 Dove si colloca un DB?

- Intefaccia utente
- Applicativi
- Software di ambiente e di sistema
- DB
- Sistema operativo

- Hardware
- Sistema di comunicazione di rete

Impropriamente si può definire nel mezzo

0.3 Problemi da NON avere

- Ridondanza dei dati: non devono esserci dati ripetuti, ogni record è U N I V O C O. Per renderlo tale credo nel corso che vedremo come si fa, spoiler: chiavi, la nostra futura bacinella di bestemmie.
- Rischi di incoerenza: i dati devono essere consistenti, ossia dato un valore, se lo si attribuisce ad un simbolo (dato, variabile, campo), quel valore dovrà essere SEMPRE quello, per ogni volta che si richiama quel simbolo

0.4 Condivisione dei dati

Data un'organizzazione avente più dipendenti, è naturale che la suddetta possa condividere un determinato insieme di dati, infatti ad ogni settore corrisponde un sistema informativo (Per chi ha fatto economia, il SIA).

Cosa accade quando si condivide una risorsa? Esatto, bisogna fare in modo che non avvengano accessi concorrenti, quindi sono implementate funzioni e procedure di prevenzione di questo genere di problemi. Un DB non protetto da modifiche NON consentite, oltre a fare schifo tipo fortissimo forever maonna guarda da bruciarlo, è NON integro.

Un DB deve essere integro

Capitolo 1

DBMS

Il DBMS (Database Management System) è un software in grado di gestire i DB.. E grazie al piffero direi, ma perchè si usa? Perchè un DB è una risorsa condivisa, e siccome potrebbe contenere dati importanti, serve un software che consenta di tenerla pulita e integra.

1.1 Cosa gestiscono

Le moli di dati su cui operano i DBMS sono tendenzialmente di grandi dimensioni, persistenti (con periodo di vita indipendenti dalle singole esecuzioni dei programmi che ci lavorano) e condivise, nel senso che diverse applicazioni ci possono lavorare sopra.

1.2 Cosa devono garantire

Dalle slides si riportano queste 3 qualità:

- Affidabilità
- Sicurezza
- Effiecienza

E tra l'altro bastava anche solo specificare consistenza ed integrità, ma il ci siamo intesi

1.3 Concetti fondamentali

1.3.1 Schema

Lo schema è la descrizione dei campi di una tabella (banalmente, è come una classe in Java)

1.3.2 Istanza

L'istanza è un record allocato a cui assegno un valore per ogni campo (in Java erano gli oggetti)

1.3.3 Modello

Il modello è l'insieme dei vari costrutti (o regole) utilizzati per organizzare i dati e descriverne i cambiamenti nel tempo.

Cosa compone un modello Le strutture di rappresentazione dei dati: Nel nostro caso si analizzerà la relazione. In base alla struttura cambia il modello, ad esempio il modello **relazionale** userà la relazione, mentre il modello a oggetti ad esempio userà altre strutture.

1.3.4 Tipo di modello

Ce ne sono due:

Logico: Vengono utilizzati dai programmi e non dipendono dalle strutture fisiche. Esempio: Relazionale, reticolare, etc.

Concettuale: Sono ancora più in alto del modello logico, quindi sono anche indipendenti dal DBMS, e sono delle descrizioni del mondo reale, usati in fase di progettazione (Quando studieremo il modello E-R partiranno le imprecazioni)

1.4 Architettura dei DBMS

Tra un BD e L'udente (o i programmi) ci sono due schemi, schema fisico e schema logico. Lo schema fisico è vicino al DB mentre il logico all'utente. (Quando verrà detto Utente, si intende sia Umano che Programma).

1.4.1 Schemi

Come menzionato prima ci sono schema fisico e logico, il fisico è banalmente la reppresentazione di files, blocchi di memoria, cache etc. mentre il logico è quello che abbiam visto prima, quindi schema relazionale etc.

1.4.2 Indipendenza dei dati

Il livello logico è indipendente dal fisico, nel senso che se tu hai un record avente dentro un ID e altri due campi, ti importa poco se verrà salvata su un SSD o su un floppy disk di fine 1800, sempre un record dovrai avere.

Osservazione: Prendiamo l'ipotesi di un record avente id e numero di telefono. (Non l'ho specificato ma l'ID è un campo numerico intero che identifica univocamente il record, fine) Se volessimo in modo sadico dividere il numero in prefisso + numero? Esatto, cambia lo schema logico, ed il record passa da 2 a 3 campi.

E lo schema fisico? Dalle slide non si capisce, quindi azzardo una risposta: siccome lo schema logico è indipendente dal fisico, non dovrebbe cambiare.. Ma non ne son sicuro.

1.4.3 Le viste

Se l'utente accedesse di cattiveria direttamente allo schema logico, ad ogni cambiamento allora dovrebbe cambiare tutto il suo programma. Come si risolve? Con le viste (O schemi esterni) che sono dei sottoinsiemi dello schema logico, che quindi contengono quantità di dati limitate (allo scopo di quell'applicativo).

In che senso? Se l'utente è la portineria, gli si crea una vista avente dati legati alla portineria. E sì, schema logico ed esterno sono indipendenti dal fisico, questo significa che l'applicativo non cambierà se cambio il supporto fisico su cui ho messo il DB. E addirittura il livello esterno è indipendente dal logico.

1.4.4 Linguaggi del DBMS

Ce ne sono due, uno è un linguaggio descrittivo dei dati, e l'altro è di manipolazione (sql). Uno descrive le strutture, l'altro ci scrive dentro e legge.

Capitolo 2

Modello Entità-Relazione

Il modello E-R è uno schema concettuale che consente di rappresentare la realtà tramite entità e relazioni tra esse.

2.1 Concetto di entità

E' un'astrazione di un certo insieme di dati (come le classi in Java). Graficamente un'entità è rappresentata da un rettangolo con all'interno il nome dell'entità.

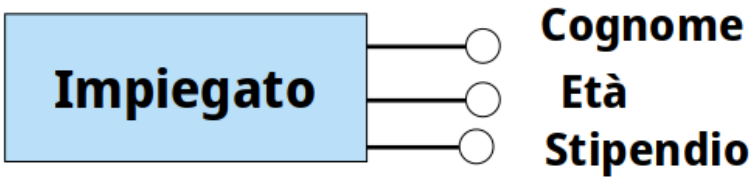
Il nome dev'essere descrittivo , ad ogni entità inoltre occorre dare una definizione. Del tipo Automobile, è quell'oggetto che se è colorato di rosso ed ha un V8 sotto il cofano ti permette di avere orgasmi multipli (dai che mi volete bene, lo so <3).

L'istanza di un'entità è il caso specifico: Automobile = entità, Bmw, Ferrari, Porsche è l'istanza. Il nome che identifica un'entità deve essere singolare ed espressivo, no abbreviazioni, no codici.

2.2 Attributo dell'entità

E' una proprità, o meglio, un campo del record che serve ai fini dell'applicazione , questo valore dipende solo dall'istanza dell'entità, non dipende da altri elementi nello schema. Inoltre ogni attributo ha un intervallo di valori finito. (Il concetto di infinito in informatica non esiste se non per le mie funzioni ricorsive rottissime di Prolog e Lisp)

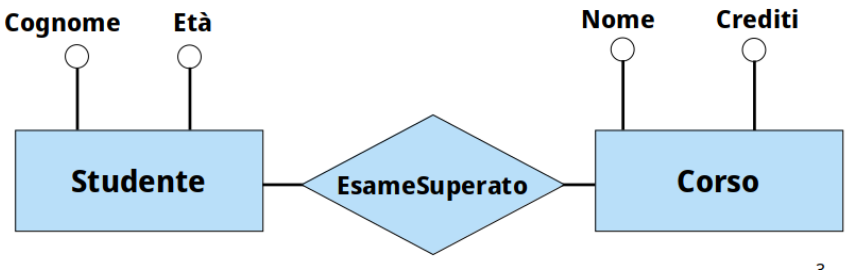
Reppresentazione:



Se il campo è chiave, il pallino diventa pieno. (Sto odiando questa slide, è tutto così non simmetrico che sclero)

2.3 Le Relazioni

Una relazione (o associazione) è un legame logico tra più entità, ed il numero di esse determina il grado (numero di entità obv), si rappresenta con un rombo e dentro ci si scrive cosa lega le entità. Come in questo esempio:



Siccome c'è differenza tra queste relazioni e le relazioni del modello relazionale, per comodità le chiamerò associazioni. Quindi associazioni -> modello e-r, relazioni -> modello relazionale.

Come scrivere le associazioni: Bisogna usare dei sostantivi al singolare. Come da esempio: Se ho due entità studente ed esame ed in mezzo ci metto "Supera" è sbagliato. Si deve mettere "Esame superato". (Alle superiori ero abituato che il verbo andasse all'infinito, che ha più senso ed è più leggibile, ma icchè vi devo dì, noi s'attacca l'asino indoe vuole i' padrone).

Tra due entità posso insierire più associazioni: Ad esempio prendendo uno studente ed un corso, ci puoi applicare due associazioni, una può essere "frequenZa" e l'altra boh.. "Frequenza passata". E da notare che nelle slides ha messo "Frequenta" e "Frequentato in passato".. Niente verbi, ovvio.

Anche un'associazione può avere degli attributi: E' una proprietà **locale** che serve ai fini dell'utente, non è una proprietà della singola entità ma di tutte coloro che sono in legame.

In termini umani: E' una funzione che associa ad ogni istanza di quella associazione un valore che appartiene ad un dominio dell'attributo.

Esempio: Studente - superamento - Esame, un attributo di superamento potrebbe essere Voto. (Tra l'altro lui ha messo esameSuperato.. boh, diobono ok che superato è un aggettivo, ma è participio passato di superare, queste slides sono un casino).

2.4 Vincoli di cardinalità

La cardinalità di una realzione specifica le quantità di interazione di istanze di più entità in un'associazione. si specifica un valore minimo ed uno massimo. Quindi è un dominio, un insieme finito di valori che è possibile assumere.

Definizione: Un vincolo di cardinalità è un limite di istanze di una associazione a cui può partecipre un'istanza di una o più entità, caratterizza meglio il significato di una associazione.

Esempio banale babbarello: Studente - Frequenza - Corso, Da 0 a N studenti frequentano N corsi. 1 corso può essere frequentato da N studenti. 1 studente frequenta N corsi.

2.5 Tipi di assoiazione

Rimanendo nel contesto delle relazioni binarie, ci sono 3 tipi di relazione:

1. Relazione uno a uno
2. Uno a Molti
3. Molti a Molti

Le quali si spiegano da sole, non c'è molto da specificare.

2.6 Generalizzazioni e Relazioni Is-A

Partendo dall'Is-A

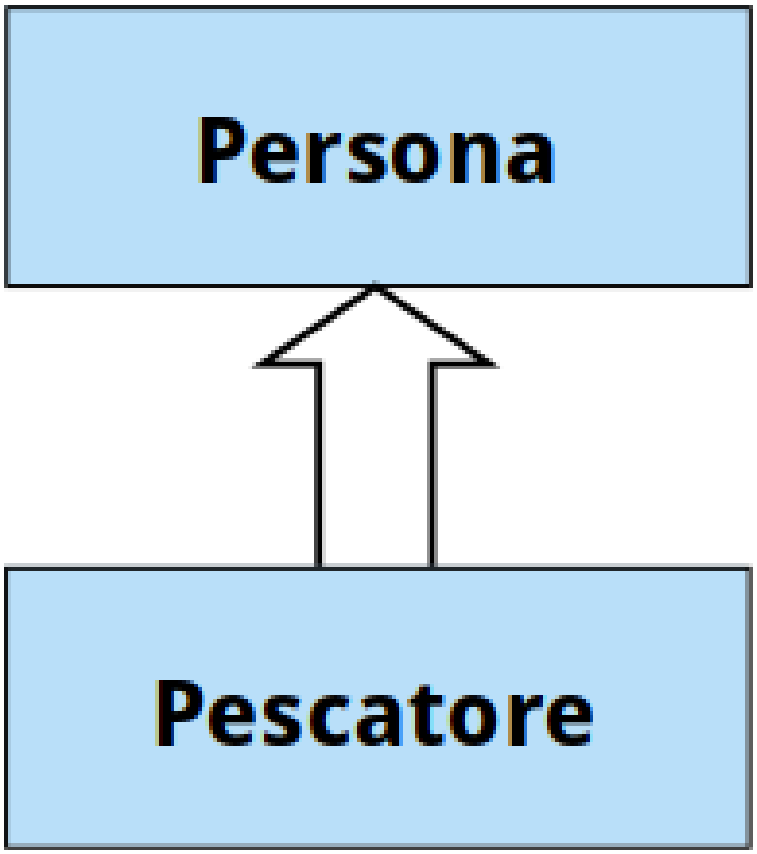
2.6.1 Relazione IS-A

E' una relazione che introduce il fatto che un'istanza possa appartenere a più entità, esempio Pilota, Pilota di Rally. E' anche definita relazione di sottoinsieme, e si utilizza per entità in cui una sia sottoinsieme dell'altra, o meglio. Una deve essere "padre" dell'altra, che si chiamerà "figlia".

Esempio: Fuoristrada Is-A Automobile Is-A Veicolo

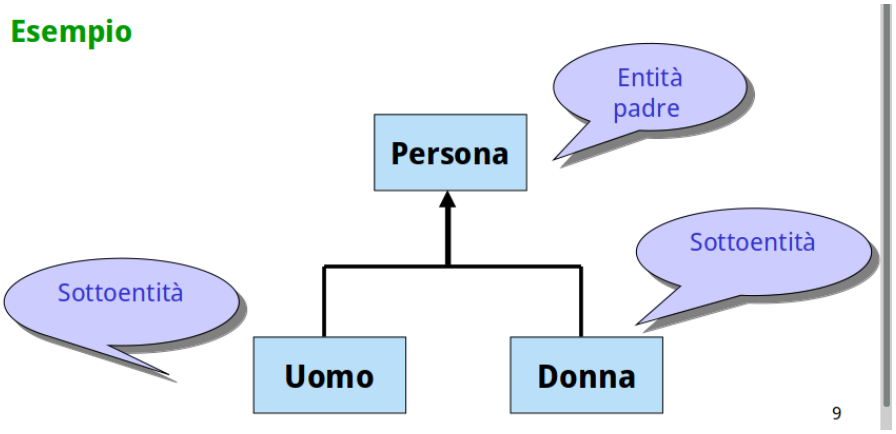
Principio di ereditarietà: Ogni attributo e proprietà dell'entità padre è anche proprietà della sua sottoentità, va da sè che la figlia può avere anche le sue proprietà. (Oh è come le classi di Java, funziona tutto così)

Graficamente:



2.6.2 Generalizzazione tra entità

Hai un'entità padre ed una serie di figlie, quello che fai è con un arco unirle, fine.



Una generalizzazione può essere di due tipi:

- Completa: L'unione delle istanze delle sottoentità è uguale all'insieme delle istanza del padre

- Non completa: in cui questo non accade

Spiegato meglioglioglio: Hai uomo e donna, e poi essere umano, se prendi tutti gli uomini + tutte le donne e fai un insieme contenente tutti essi, ottieni l'essere umano. (Sì, LGBT, abbiate pietà, non è il momento di polemizzare)
Ipotizziamo gli animali: hai cane gatto e criceto, e poi animali.. Beh se sommi tutti i gatti criceti e cani, non ottieni l'insieme degli animali. Mancano i gamberetti per esempio.

Detto scientificamente accurato: Se le sotto entità sono partizioni dell'entità padre, è completa, altrimenti no.

L'entità padre può avere più generalizzazioni

2.7 Attributi composti

Un attributo può esser definito su un dominio di più campi. In altri termini, un attributo può essere a sua volta un record. Tipo indirizzo (composto da via numero e cap), è uno di questi. Potrebbe essere anche un attributo del tipo "dati anagrafici". Quando si ha questa situazione l'attributo è composto.

2.8 Relazioni n-arie

La relazione BI-naria coinvolge due entità, la ternaria tre, la quaternaria.. Via, si è capito, se è n-aria coinvolge n entità. A sua volta anche la relazione può avere i suoi attributi.

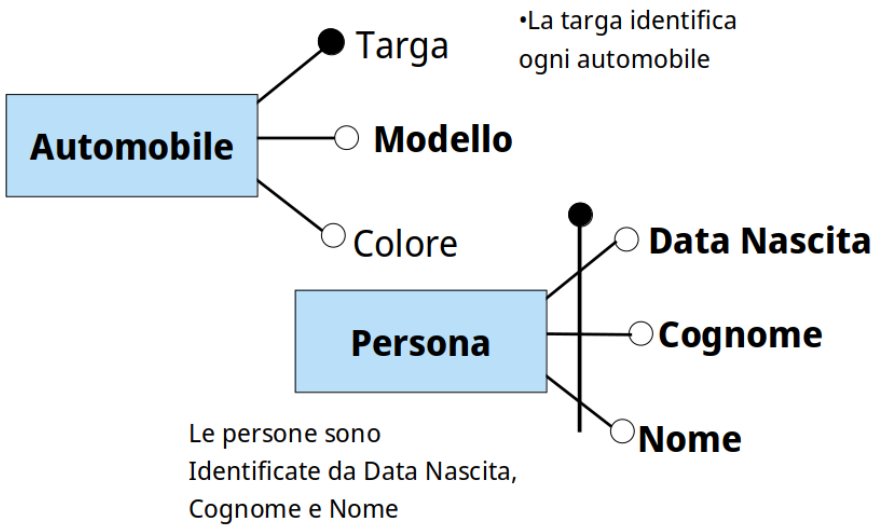
Esistono anche relazioni ricorsive o definite su se stesse: Il classico esempio è il papa. papa - successione - papa, per tenervelo in testa pensate a "Morto un papa, se ne fa n'altro".

2.9 Identificatori/Chiavi

E' un insieme di attributi o relazioni che ti permettono di identificare le istanze di un'entità. (Per chi ha già fatto sta roba sono le chiavi primarie). Ogni entità ha un certo insieme di identificatori (Da 1 a ∞, basta che ce ne sia uno $\forall$ entità)

2.9.1 Tipi di identificatore/chiave

- **Interno**: Formati solo da attributi dell'entità stessa. può essere solo uno, possono essere anche più di uno. Come in questo esempio:



- **Esterno**: E' formato da attributi dell'entità e da relazioni che la coinvolgono, oppure solo da queste relazioni.

Esempio grafico: