Informatica per Statistica
Riassunto della lezione del 27/11/2013

Igor Melatti

Introduzione alle basi di dati

- Questo riassunto è da intendersi come un commento alle slide `BD2002-01.PDF`
- Slides 3-4: si sottolinea che l'approccio alle basi di dati può essere sia tecnologico che metodologico, e che tale distinzione può essere in realtà fatta per l'intera informatica
  - tecnologico: riferito ai prodotti effettivamente utilizzabili, sia hardware che software
    - hardware: esistono computer dedicati e ottimizzati per le basi di dati, anche se nei casi più semplici (e comuni) va bene anche un portatile, o comunque un computer desktop moderno
    - software: esistono programmi fatti apposta per gestire basi di dati: sono i *DBMS*, ovvero *DataBase Management System*
  - metodologico: riferito alle metodologie da seguire per progettare correttamente una base di dati
  - ovvero, che passi seguire per poter poi usare in modo appropriato e al massimo delle sue possibilità la parte tecnologica
- Slide 5: quello di cui si parla in un corso di basi di dati
  - i linguaggi per l'utilizzo dei dati sono l'SQL con le sue varie estensioni (dipendenti dal DBMS che si sta usando)
  - i sistemi per la gestione dei dati sono i DBMS, insieme all'hardware sul quale vengono eseguiti (e che eventualmente può essere dedicato esclusivamente a tale scopo)
  - le metodologie di progettazione è ciò di cui si parla in queste lezioni: le più importanti sono quelle per la ***progettazione concettuale*** e la ***progettazione logica***
  - i modelli per l'organizzazione dei dati sono quelli che si usano nelle metodologie di progettazione

    - per la progettazione concettuale ci sarà un modello concettuale dei dati
    - per la progettazione logica ci sarà un modello logico dei dati

- Slides 9-14: prendiamo per esempio l'"azienda" Università (ovvero, una qualsiasi università italiana)
  - sistema organizzativo: l'intero sistema delle segreterie più gli organi che stabiliscono le regole (senato accademico, consigli di facoltà, consigli di dipartimento, CAD, etc...)
  - sistema informativo: come le varie segreterie mantengono le informazioni; fino a qualche decennio fa tutto su carta, ora buona parte (ma non tutto) è informatizzata
    - esempi di informazioni da mantenere: anagrafica (nome, cognome, indirizzo, codice fiscale...) dei docenti, degli studenti e delle segreterie, dati sugli stipendi, sugli esami da fare e superati, etc
  - sistema informatico: il DBMS per l'iscrizione automatica agli esami da parte degli studenti (Infostud per la Sapienza), il DBMS per la gestione degli stipendi dei docenti, il sito web dell'Università, ...

- Slide 18: nelle basi di dati, si usano i dati per rappresentare le informazioni
- Slide 19: differenza tra informazione e dato:
  - l'informazione accresce la conoscenza su un qualche argomento
  - il dato, per diventare informazione, deve essere interpretato e/o contestualizzato

- Slides 22 e 24: molto importante, le due definizioni di base di dati
- Slide 27: per "memoria centrale" si intende la RAM, per "limite fisico" si intende quello dei dispositivi di memoria di massa, come dischi magnetici (come l'hard disk) e cassette (sì, si usano anche quelle)
- Slide 28: essendo le basi di dati persistenti, non si può usare la RAM per memorizzarle
  - tuttavia la RAM viene comunque usata dai DBMS per i calcoli intermedi

- Slide 29: nel caso dell'Università, tanto il sistema per gestire le prenotazioni agli esami quanto quello che gestisce l'inserzione di nuovi studenti devono necessariamente convidere alcuni dati
  - quindi le segreterie e gli studenti possono accedere (potenzialmente in contemporanea) agli stessi dati

- Slides 30-32: esempio di ridondanza, per due volte si dice qual è il docente di un certo corso
- Slides 33-34: differenza tra una base di dati ed un semplice archivio; normalmente, si indica con archivio qualcosa di più semplice di una base di dati, ad esempio non convidisa
- Slide 36: ad esempio, segreterie e studenti accedono entrambi ad alcuni dati comuni, ma l'inserimento di un nuovo studente è concesso solo alle segreterie
- Slides 38-41: quando un utente di una base di dati richiede dei dati (*interrogazione*), o cerca di modificarli, esegue quella che viene detta *transazione*
  - essenzialmente, la richiesta dell'utente viene spezzata in un certo numero di piccole operazioni (questa fase è automatica, la fa il DBMS e l'utente non se ne accorge)
  - un DBMS garantisce che tali operazioni o vengono effettuate tutte insieme, o non ne viene effettuata nessuna (*atomicità* delle transazioni)
  - un DBMS garantisce che se più utenti generano contemporaneamente transazioni che hanno effetto sugli stessi dati (si parla quindi di transazioni *concorrenti*), allora tali transazioni non si disturbano tra loro
  - ovvero, l'effetto dell'esecuzione concorrente è lo stesso che se fossero effettuate sequenzialmente (prima l'una e poi l'altra)
  - ad esempio, se due diverse segreterie aggiungono uno studente ciascuna, allora il numero di matricola che si genera sarà correttamente diverso per ognuno dei due nuovi studenti aggiunti
    * questo potrebbe sembrare ovvio, ma non lo è
    * si consideri nuovamente l'esempio dell'aggiunta degli studenti
    * si immagini il numero di matricola come un semplice contatore del numero degli studenti (cosa non lontana dalla realtà...)
    * si supponga che l'aggiunta di un nuovo studente generi la seguente transazione: `leggi`(matricola), `incrementa`(matricola), `scrivi`(matricola)
    * si supponga, come spesso avviene, che l'incremento avvenga in RAM, mentre `leggi` e `scrivi` hanno effetto sui dati persistenti
    * quindi, se due segreterie (ad esempio quella di Statistica, chiamata S per brevità, e quella di Informatica, chiamata I per brevità) effettuano l'aggiunta di un nuovo studente, si avranno due transazioni contemporanee che possono essere scritte così: $\texttt{leggi}_S$(matricola), $\texttt{incrementa}_S$(matricola), $\texttt{scrivi}_S$(matricola) per Statistica, e $\texttt{leggi}_I$(matricola), $\texttt{incrementa}_I$(matricola), $\texttt{scrivi}_I$(matricola) per Informatica

    - queste 6 operazioni possono intervallarsi tra di loro in modo imprevedibile (fermo restando, ovviamente, che all'interno di una stessa transazione l'ordine viene rispettato, quindi ad esempio $\texttt{incrementa}_S$(matricola) viene sempre prima di $\texttt{scrivi}_S$(matricola), ma potrebbe venire dopo $\texttt{scrivi}_I$(matricola))
    - nota bene: in RAM, le due matricole, dopo essere lette dalla base di dati con l'operazione `leggi`, sono memorizzate in due locazioni *diverse*, una per la transazione di S e una per I
    - se questo intervallamento di operazioni risultasse $\texttt{leggi}_S$(matricola), $\texttt{leggi}_I$(matricola), $\texttt{incrementa}_S$(matricola), $\texttt{scrivi}_S$(matricola) $\texttt{incrementa}_I$(matricola), $\texttt{scrivi}_I$(matricola), allora entrambe le segreterie leggerebbero lo stesso numero di matricola, e assegnerebbero quindi a due studenti diversi lo stesso numero di matricola
    - questo invaliderebbe la funzione principale della matricola: quella di individuare un singolo studente

- Slides 45-46: saltare
- Slide 47: modello dei dati, descrive quali strumenti si usano per descrivere come organizzare i dati (che, di partenza, sono disorganizzati)
  - nelle lezioni successive verranno introdotti il modello relazionale (per la progettazione logica, quindi a livello logico) e il modello entità–relazione (per la progettazione concettuale, quindi a livello concettuale)
  - quindi esistono diversi modelli di dati, a seconda del livello di dettaglio che si vuole avere nella descrizione dell'organizzazione dei dati
  - anche una volta deciso il livello di dettaglio (per esempio quello logico), possono esistere vari modelli utilizzabili
  - i DBMS tipicamente ne supportano (ovvero, permettono di usarne) solo uno
  - un DBMS *relazionale* è chiamato così perché supporta il modello relazionale dei dati (a livello logico)
- Slides 48-51: ci si ritornerà (lezione 26)
- Slides 53-54: i concettuali sono quelli di più alto livello, una volta fatti quelli si può andare più nel dettaglio con un modello logico
- Saltare le slides 55–75, con l'eccezione della 72
  - i progettisti e realizzatori di DBMS sono coloro che scrivono il codice (quasi sempre in C) del programma per i DBMS; non è argomento di questo corso

- progettisti di DBMS: coloro che, partendo da una descrizione informale (e possibilmente ambigua) delle informazioni di partenza (ovvero, quelle che si vogliono organizzare per formare una base di dati) progettano la base di dati (attraverso le progettazioni concettuale e logica)
- amministratori di DBMS: una volta finita la progettazione logica, occorre:
  * realizzare in pratica la base di dati, ovvero implementarla usando un DBMS appropriato
  * manutenerla, ovvero sorvegliarne il funzionamento, creando gli account per gli utenti (ovvero definendo quali utenti sono abilitati all'uso e cosa possono fare), modificando eventualmente alcune parti qualora ne sopraggiunga la necessità (è spesso necessario aggiungere funzionalità e/o nuove informazioni da mantenere) e risolvendo malfunzionamenti
- progettisti e programmatori di applicazioni: scrivono programmi (spesso in C) che si *interfacciano* con la base di dati, ovvero richiedono e/o modificano dati; non è argomento di questo corso
- utenti finali: eseguono delle operazioni predefinite (erano descritte nella descrizione informale di partenza della base di dati, e sono state realizzate nella fase di implementazioni della base di dati con un DBMS), che come descritto sopra sono opportunamente trasformate in transazioni
  * non occorre accedere direttamente al DBMS: si possono usare delle applicazioni scritte dai progettisti e programmatori di applicazioni (quelli del punto precedente)
- utenti casuali: i DBMS permettono comunque di definire interrogazioni non previste tramite l'SQL; per effettuare tali interrogazioni occorre solitamente avere accesso diretto al DBMS
- progettisti e realizzatori di DBMS sono praticamente sempre diversi dalle altre figure presentate sopra
- per basi di dati piccole, tutti gli altri attori possono anche coincidere
- non è detto che debbano essere per forza scritte delle applicazioni che accedono alla base di dati, soprattutto per basi di dati piccole: si può usare direttamente il DBMS stesso
- nei casi più complessi (ad esempio il caso dell'Università visto prima) servono tutti gli attori di cui sopra, e saranno tipicamente persone diverse