Tecnologia di un Database Server (centralizzato)

Introduzione generale

Angelo Montanari

Dipartimento di Matematica e Informatica
Università di Udine

Introduzione

Dalle funzionalità ai **meccanismi**...

Ragioni:

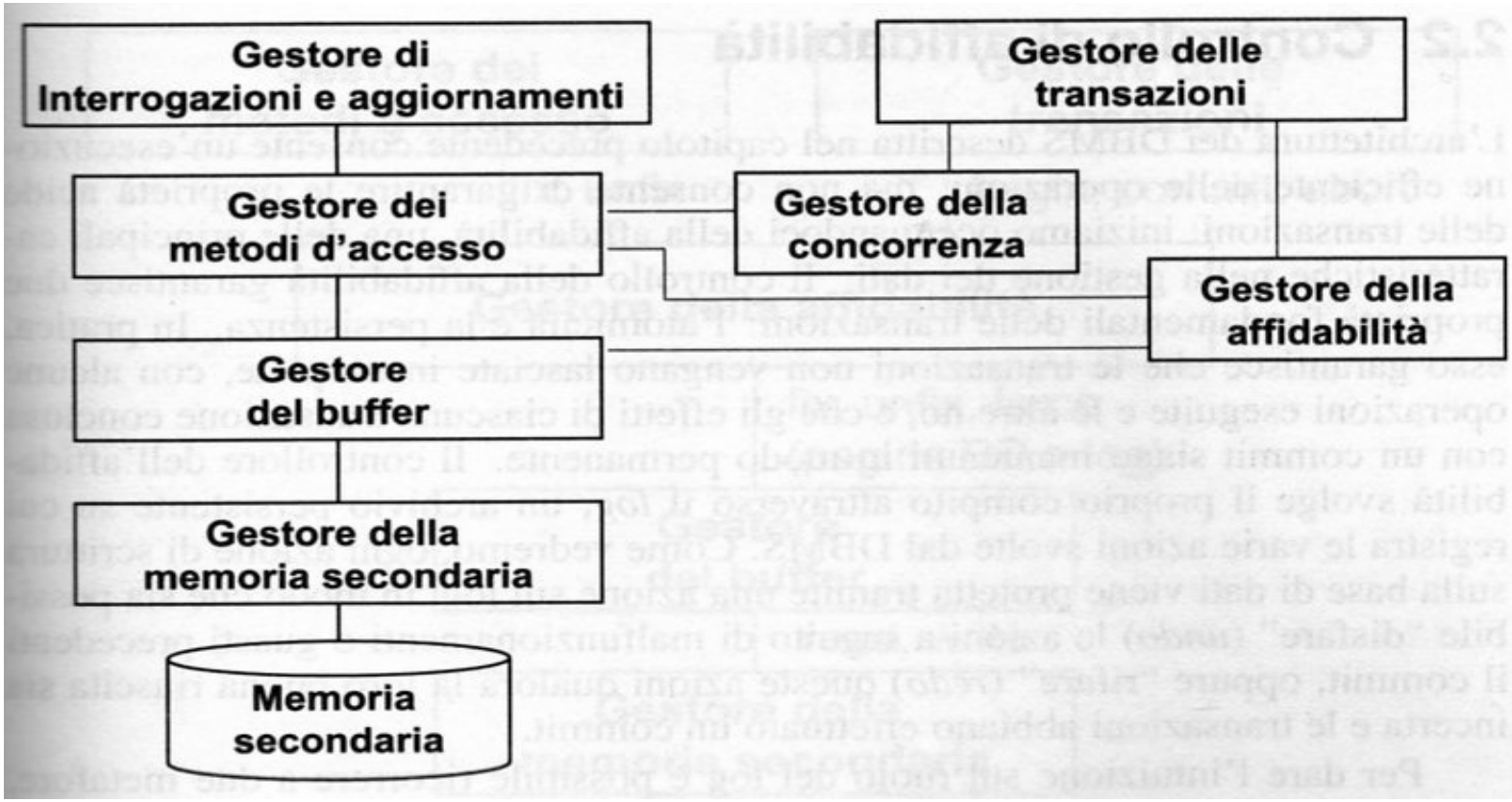
- configurazione del database server;
- modalità di esecuzione delle applicazioni che influenzano il comportamento del sistema;
- (linea di tendenza) molti meccanismi vengono estratti dal DBMS e messi a disposizione come servizi nel software di rete (utili per configurare un'applicazione).

Componenti di un database server

Componenti di un server per la gestione dei dati:

1. **ottimizzatore** (o **gestore delle interrogazioni**): stabilisce le strategie di accesso ai dati per l'esecuzione delle interrogazioni;
2. **gestore dei metodi di accesso ai dati**: trasforma le richieste contenute nei comandi di alto livello (SQL) in operazioni di lettura e scrittura di dati in memoria secondaria;
3. **buffer manager**: gestisce il trasferimento effettivo delle pagine della base di dati da/a memoria secondaria a/da memoria principale, interagendo col gestore della memoria secondaria;
4. **controllore della concorrenza**: gestisce gli accessi concorrenti alla base di dati;
5. **controllore dell'affidabilità**: garantisce il buon funzionamento del sistema in presenza di malfunzionamenti e guasti.

Architettura di un database server



Le transazioni

Una prima definizione (informale): una **transazione** identifica un'unità elementare di lavoro svolta da un programma applicativo, cui si vogliono associare particolari caratteristiche di correttezza, robustezza ed isolamento.

Un sistema che mette a disposizione un meccanismo per la definizione e l'esecuzione di transazioni viene detto sistema transazionale.

Due comandi per l'**incapsulamento**:

- begin (o start) transaction (bot) e end transaction (eot)

Il comando end transaction è di norma implicito. Se viene omesso il comando begin/start transaction, il sistema assume che ogni singola istruzione costituisca una transazione (modalità **autocommit**). E' ciò che accade usualmente negli ambienti interattivi.

Le istruzioni di commit e abort

Due istruzioni particolari:

- **commit** (commit work)
- **abort** (rollback work)

Una sola delle due istruzioni può/deve comparire in una transazione ben formata.

Un esempio (descrizione astratta di una transazione):

```
begin transaction
X := X - 10;
Y := Y + 10;
commit work;
end transaction
```

Un esempio di transazione in SQL

Immaginiamo di voler specificare una transazione che trasferisce 10000 euro dal conto corrente numero 43719 al conto corrente numero 65286. In SQL, una tale transazione può essere formulata nel seguente modo:

```
start transaction;
update ContoCorrente
    set Saldo = Saldo + 10000
    where NumConto = 65286;
update ContoCorrente
    set Saldo = Saldo - 10000
    where NumConto = 43719;
commit work;
```

Il ruolo delle istruzioni commit / rollback work

L'istruzione **commit work** forza una conclusione positiva della transazione, con la registrazione permanente di tutti gli aggiornamenti nella base di dati.

In transazioni più articolate, che utilizzano strutture di controllo, ci possono essere dei casi in cui, dopo aver effettuato parte delle operazioni, il verificarsi di alcune condizioni indesiderate impone l'annullamento degli aggiornamenti effettuati, col conseguente ripristino della situazione precedente l'(inizio dell')esecuzione della transazione. Tale obiettivo può essere raggiunto tramite l'esecuzione dell'istruzione **rollback work**.

Immaginiamo, ad esempio, che la precedente transazione vada annullata qualora, per effetto del prelievo, il saldo del conto numero 43719 diventi negativo.

Una variante dell'esempio precedente

```
start transaction;
update ContoCorrente
    set Saldo = Saldo + 10000
    where NumConto = 65286;
update ContoCorrente
    set Saldo = Saldo - 10000
    where NumConto = 43719;
select Saldo into A
    from ContoCorrente
    where NumConto = 43719;
if A >= 0
    then commit work;
    rollback work;
```

Proprietà acide delle transazioni: atomicità - 1

Atomicità. Una transazione è un'unità indivisibile di esecuzione: o produce tutti i suoi effetti o non ne produce alcuno (tutto o niente).

Se durante l'esecuzione delle istruzioni che compongono una transazione si verifica un'errore che rende impossibile completare le letture/modifiche della base di dati previste dalla transazione, il sistema deve essere in grado di ripristinare la situazione immediatamente precedente l'inizio dell'esecuzione della transazione, disfando il lavoro svolto dalle istruzioni eseguite sino a quel momento (operazione di **undo**).

Al contrario, dopo l'esecuzione del commit, il sistema deve garantire che la transazione lasci la base di dati nello stato finale prodotto. In alcuni casi, ciò può imporre di dover rifare il lavoro svolto (operazione di **redo**).

La corretta esecuzione dell'operazione di commit fissa l'istante atomico/ indivisibile in cui la transazione **va a buon fine** (prima di essa ogni guasto provoca un rollback, dopo può richiedere un redo).

Proprietà acide delle transazioni: atomicità - 2

Diverse ragioni possono portare all'**aborto di una transazione** (i meccanismi che realizzano l'abort sono sempre gli stessi):

- esecuzione del comando rollback work: un suicidio autonomamente deciso dalla transazione (vedi esempio);
- in diversi casi, il sistema, accertato che la transazione non è in grado di portare a compimento in modo corretto il proprio lavoro, può uccidere la transazione;
- varie transazioni possono essere uccise a seguito di un guasto di sistema.

Di norma, l'esecuzione delle transazioni va a buon fine (terminano con un commit); in casi sporadici (ad esempio, malfunzionamenti o situazioni impreviste), terminano con un abort.

Proprietà acide delle transazioni: consistenza

Consistenza. L'esecuzione della transazione non deve violare i vincoli di integrità definiti sulla base di dati. Tale condizione può forzare l'annullamento della transazione o un intervento per correggerne gli effetti indesiderati (si vedano le politiche di reazione alle violazioni dei vincoli di integrità referenziale).

La verifica dei vincoli di integrità può essere di **tipo immediato** (effettuata durante l'esecuzione della transazione, rimuovendo gli effetti dell'esecuzione della transazione che provocano la violazione dei vincoli, senza forzare l'abort della transazione) o di **tipo differito** (effettuata a conclusione della transazione dopo che è stata richiesta l'esecuzione del commit; in questo caso, se un vincolo è stato violato, l'istruzione commit work non va a buon fine e gli effetti della transazione vengono annullati)

Proprietà acide delle transazioni: isolamento

Isolamento. L'esecuzione di una transazione deve essere (logicamente) indipendente dalla contemporanea esecuzione di altre transazioni, ossia si richiede che il risultato dell'esecuzione (fisicamente) concorrente di un insieme di transazioni sia lo stesso che le stesse transazioni produrrebbero qualora ciascuna di esse fosse eseguita (fisicamente) da sola.

La proprietà di isolamento vuole anche garantire che l'**esito** (positivo o negativo) di ciascuna transazione sia **indipendente** dall'esito delle altre. In particolare, si vuole impedire che l'esecuzione di un rollback di una transazione causi l'esecuzione del rollback di altre, generando una reazione a catena (effetto domino). Ciò potrebbe accadere, ad esempio, qualora una transazione leggesse dati modificati da una transazione non ancora completata, ossia una transazione che non abbia ancora eseguito il commit.

Proprietà acide delle transazioni: persistenza

Persistenza. L'effetto di una transazione che ha eseguito il commit correttamente non deve essere perso per alcun motivo.

Come sono garantite le proprietà acide delle transazioni?

- Le proprietà di **atomicità** e di **persistenza** sono garantite dal **controllo dell'affidabilità**.
- La proprietà di **isolamento** è garantita dal **controllo della concorrenza**.
- La proprietà di **consistenza** è di fatto gestita dai **compilatori del DDL** che introducono opportuni controlli di consitenza nei dati e opportune procedure per la loro verifica, che vengono poi eseguite dalle transazioni.