LA GESTIONE DELLE TRANSAZIONI

La presenza simultanea di programmi utente che richiedono l'accesso al contenuto da pre base di dati rappresenta lo scenario tipico dei sistemi OLTP- On Line Transaction Processing.

Ad esempio, basti pensare ad un sistema di e-commerce in cui migliaia di utenti attraverso apposite interfacce web possono effettuare acquisti on-line, generando una moltitudine di operazioni contemporanee che un DBMS deve gestire garantendo, nel contempo, la consistenza e integrità dei dati, nonché l'affidabilità del sistema di basi di dati a valle del malfunzionamento di componenti sia hardware sia software.

In un tale contesto, la mancanza di un protocollo per il controllo della concorrenza potrebbe portare la base di dati in uno stato non consistente (es. utenti che acquistano più prodotti di quelli effettivamente disponibili), dall'altra parte, l'assenza di procedure per il recupero (o recovery) della base di dati in corrispondenza di guasti potrebbe comportare la perdita di informazioni di interesse vitale per l'organizzazione (es. i dati degli acquisti della giornata).

Per tale motivo ogni DBMS possiede un modulo per la Gestione della Concorrenza ed uno per la Gestione dell'Affidabilità che si occupano di garantire l'accesso concorrente alla base di dati, preservandone il contenuto anche in corrispondenza di crash del sistema.

Nella pratica, un DBMS esegue delle particolari procedure dette transazioni.

Definizione Transazione: Una transazione su una base di dati è un'operazione, o una sequenza di operazioni, generata da un utente o programma applicativo che legge o aggiorna il contenuto di una base di dati.

Una transazione rappresenta quindi un'unità logica di elaborazione che corrisponde a una serie di operazioni fisiche elementari (letture/scritture) sulla base di dati. Essa può coincidere con un intero programma, una parte di un programma o con un singolo comando (es. uno statement SQL di INSERT o UPDATE) che vengono eseguiti in un ambiente DBMS, e nel contempo, costituisce una unità atomica (non divisibile) di modifiche fatte allo stato di una base di dati.

L'esecuzione di una transazione può avere due possibili esiti: una transazione o termina in uno stato finale previsto dal programma in esecuzione (attraverso quello che in gergo viene detto commit) o porta il sistema ad uno stato precedente all'esecuzione della transazione (attraverso un abort).

Se una transazione viene completata con successo si dice che quest'ultima è stata "committata" (in gergo committed) portando la base di dati in un nuovo stato consistente. Di contro, se la transazione non viene eseguita con successo essa viene "abortita" (in gergo aborted) e la basi di dati riportata nello stato precedente alla sua esecuzione. Nel caso di abort, eventuali azioni ed effetti parziali della transazione devono essere essere "disfatti" (attraverso quello che viene detto in gergo rollback o undo), in quanto lascerebbero la base di dati in uno stato di inconsistenza.

Si noti che talora le transazioni possono anche contenere solo interrogazioni: in tal caso, l'atomicità é sempre assicurata in quanto una interrogazione non modifica lo stato di una base di dati.

Di seguito è riportato un esempio di transazione.:

```
UPDATE PRODOTTI SET Quantità=Quantità- WHERE Codice=x;

INSERT INTO CARRELLO(Cliente, Prodotto, Qta, Data)

VALUES (z,x,y, '13-Gen-2015 20:30:54');
```

Nell'ambito di un sistema di e-commerce, essa costituisce una possibile sequenza di operazioni relative ad una procedura di acquisto di un singolo prodotto.

In particolare, la quantità in magazzino di un dato prodotto e viene decrementata (attraverso un'operazione di UPDATE sulla tabella PRODOTTI) di una certa quantità y, che corrisponde poi a quella ordinata dallo specifico cliente z all'interno del suo carrello elettronico (attraverso un'operazione di INSERT nella tabella CARRELLO).

Le transazioni devono possedere alcune proprietà fondamentali, in particolare devono soddisfare le cosiddette **proprietà ACID**, come proposte da Haerder e Reuter nel 1983, e definite di seguito.

(Proprietà ACID: **atomicità**). È la cosiddetta proprietà del "tutto o niente": una transazione deve essere atomica ovvero deve essere eseguita nella sua interezza oppure non eseguita per niente.

(Proprietà ACID: **consistenza**). Una transazione deve godere della proprietà di consistenza, ovvero deve causare una trasformazione di uno stato consistente della base di dati in un altro stato consistente. Un DBMS, in particolare, deve poi assicurare che tutti i vincoli definiti sulla base di dati siano soddisfatti durante l'esecuzione di transazioni.

(Proprietà ACID: **isolamento**). Le transazioni devono essere isolate ovvero eseguite in modo indipendente l'una dalle altre. Questo sta a significare che gli effetti parziali di transazioni incomplete non devono essere visibili alle altre transazioni. Questa proprietà garantisce che le modifiche apportate da una transazione non siano visibili ad altre transazioni fino a quando la transazione stessa non viene commutata. Questo isolamento aiuta a prevenire i conflitti tra transazioni concorrenti.

(Proprietà ACID: **durability** o persistenza). Una transazione che è terminata con un "commit" deve essere persistente ovvero i suoi effetti devono essere registrati in modo permanente nella base di dati e non devono essere mai persi per alcun motivo.

Con riferimento all'esempio precedente, la proprietà di atomicità assicura che le singole operazioni di una transazione di acquisto o siano entrambe eseguite o nessuna delle due: non potrà esistere, ad esempio, una situazione in cui venga decrementata la quantità di un prodotto in magazzino senza che sia creato il corrispettivo record nel carrello per l'utente interessato con la stessa quantità acquistata.

Per ciò che concerne la consistenza, il DBMS assicurerà che l'esecuzione delle transazioni non violi eventuali vincoli di integrità della base di dati: ad esempio una transazione che cerca di acquistare una quantità di prodotto maggiore di quella disponibile in magazzino potrebbe essere abortita.

L'isolamento invece garantisce che ogni transazione d'acquisto sia eseguita in maniera indipendente da tutte le altre evitando possibili anomalie dovute al fatto che il DBMS gestisce una moltitudine di richieste contemporanee: una transazione non dovrebbe vedere gli aggiornamenti parziali della quantità dei prodotti in magazzino generate da transazioni che non hanno ancora effettuato il commit.

Infine, con la persistenza si garantisce che gli effetti di tutte le transazioni che hanno effettuato il commit siano memorizzati in maniera persistente e duratura: anche dopo il verificarsi di malfunzionamenti alle componenti hardware o software del sistema, il DBMS sarà sempre in grado di ricostruire il contenuto della base di dati relativo all'ultimo stato consistente non perdendo traccia in alcun modo delle azioni delle transazioni di acquisto che prima del guasto avevano effettuato il commit.

Da un punto di vista più astratto, una transazione coinciderà con un blocco di istruzioni caratterizzato da un inizio (BEGIN TRANSACTION, esplicitato in SQL) e da una fine (END TRANSACTION, di norma non esplicitato in SQL) e al cui interno deve essere eseguito una e una sola volta almeno uno dei seguenti comandi:

• commit, per terminare correttamente e rendere persistenti le azioni della transazione;

• rollback, per abortire la transazione.

Si può notare che, oltre agli stati Active (in cui la transazione viene eseguita), Committed (in cui la transazione si trova dopo avere eseguito correttamente il commit) e Aborted (in cui la transazione si trova dopo essere stata abortita), sono stati introdotti due stati intermedi:

• Partially Committed (Parzialmente Committata): si verifica quando viene eseguita l'ultima istruzione di una transazione. A questo punto, la transazione potrebbe essere abortita se, ad esempio, viene violato qualche vincolo di integrità oppure se il sistema ha subito un crash e non si è avuta la possibilità di salvare in memoria secondaria i dati aggiornati dalla transazione stessa. In entrambi i suddetti casi, la transazione si porta prima nello stato Failed in attesa di essere poi definitivamente abortita. Se invece, all'atto del commit, la transazione viene completata con successo e tutti i suoi effetti sono registrati su memoria secondaria, essa può passare allo stato Committed.

• Failed (Fallita): si verifica quando la transazione non può essere per qualche motivo committata in maniera corretta o se la transazione viene abortita mentre si trova in esecuzione.