




**Linguaggio SQL: costrutti avanzati**

Gestione delle transazioni

**Gestione delle transazioni**

- ▷ Introduzione
- ▷ Transazioni in SQL
- ▷ Proprietà delle transazioni



2



**Gestione delle transazioni**

Introduzione




**Esempio applicativo**



- ▷ Operazioni bancarie
  - operazione di prelievo dal proprio conto corrente mediante bancomat



4



**Esempio applicativo**




- ▷ Operazioni bancarie
  - operazione di prelievo dal proprio conto corrente mediante bancomat
  - operazione di versamento di denaro contante sul proprio conto corrente



5



**Prelievo**



- ▷ Operazioni svolte
  - specificare l'importo richiesto
  - verificare la disponibilità
  - memorizzare il movimento
  - aggiornare il saldo
  - abilitare l'erogazione della somma richiesta



6

**Prelievo**



▷ Operazioni svolte
 


- specificare l'importo richiesto
- verificare la disponibilità
- memorizzare il movimento
- aggiornare il saldo
- abilitare l'erogazione della somma richiesta

▷ Tutte le operazioni devono essere eseguite correttamente, altrimenti il prelievo non va a buon fine


7

**Prelievo**


▷ *Cosa succede se un cointestatario diverso del conto fa un altro prelievo?*


8


**Prelievo**

▷ *Cosa succede se un cointestatario diverso del conto fa un altro prelievo?*

▷ *Cosa succede in caso di malfunzionamento?*


9


**Versamento**



▷ Operazioni svolte
 


- verificare l'importo versato
- memorizzare il movimento
- aggiornare il saldo

▷ Tutte le operazioni devono essere eseguite correttamente, altrimenti il versamento non va a buon fine


10

**Versamento**


▷ *Cosa succede se un'altra persona fa un versamento sullo stesso conto?*


11

**Versamento**

▷ *Cosa succede se un'altra persona fa un versamento sullo stesso conto?*

▷ *Cosa succede in caso di malfunzionamento?*


12

### Esempio: operazioni bancarie

- ▷ La base di dati bancaria è un ambiente multiutente
  - diversi operatori possono operare contemporaneamente sulla stessa porzione di dati

### Esempio: operazioni bancarie

- ▷ La base di dati bancaria è un ambiente multiutente
  - diversi operatori possono operare contemporaneamente sulla stessa porzione di dati
- ▷ La gestione corretta delle informazioni richiede
  - meccanismi per la gestione dell'*accesso concorrente* alla base di dati

### Esempio: operazioni bancarie

- ▷ La base di dati bancaria è un ambiente multiutente
  - diversi operatori possono operare contemporaneamente sulla stessa porzione di dati
- ▷ La gestione corretta delle informazioni richiede
  - meccanismi per la gestione dell'*accesso concorrente* alla base di dati
  - meccanismi per il *ripristino* (recovery) dello stato corretto della base di dati in caso di guasti

### Gestione delle transazioni

- ▷ Necessaria quando più utenti possono accedere contemporaneamente ai dati
- ▷ Offre meccanismi efficienti per
  - gestire l'*accesso concorrente* ai dati
  - effettuare il *recovery* a seguito di un malfunzionamento

### Transazione

- ▷ Una transazione è una sequenza di operazioni che
  - rappresenta un'unità elementare di lavoro
  - può concludersi con un successo o un insuccesso
    - in caso di successo, il risultato delle operazioni eseguite deve essere memorizzato in modo permanente nella base di dati
    - in caso di insuccesso, la base di dati deve ritornare allo stato precedente l'inizio della transazione

### Sistema transazionale

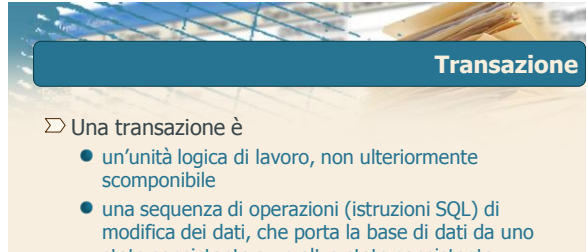
- ▷ Un sistema che mette a disposizione un meccanismo per la definizione e l'esecuzione di transazioni viene detto *sistema transazionale*
- ▷ I DBMS contengono blocchi architetturali che offrono servizi di gestione delle transazioni



## Gestione delle transazioni

### Transazioni in SQL

DBG




## Transazione

⊃ Una transazione è

- un'unità logica di lavoro, non ulteriormente scomponibile
- una sequenza di operazioni (istruzioni SQL) di modifica dei dati, che porta la base di dati da uno stato consistente a un altro stato consistente
  - non è necessario conservare la consistenza negli stati intermedi

DBG

20




## Inizio di una transazione

⊃ Per definire l'inizio di una transazione, il linguaggio SQL prevede l'istruzione

- START TRANSACTION

DBG

21



## Inizio di una transazione

⊃ Per definire l'inizio di una transazione, il linguaggio SQL prevede l'istruzione


- START TRANSACTION

⊃ Di solito l'istruzione di inizio della transazione è omessa

- l'inizio è implicito
  - prima istruzione SQL del programma che accede alla base di dati
  - prima istruzione SQL successiva all'istruzione di termine della transazione precedente

DBG

22



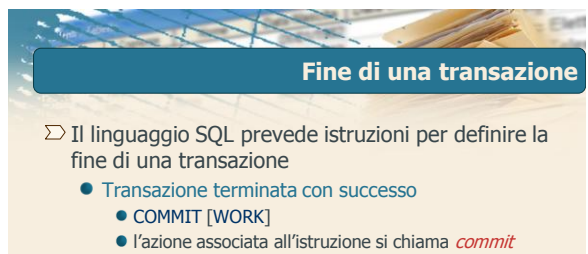
## Fine di una transazione

⊃ Il linguaggio SQL prevede istruzioni per definire la fine di una transazione

- Transazione terminata con successo
  - COMMIT [WORK]
  - l'azione associata all'istruzione si chiama *commit*

DBG

23



## Fine di una transazione

⊃ Il linguaggio SQL prevede istruzioni per definire la fine di una transazione

- Transazione terminata con successo
  - COMMIT [WORK]
  - l'azione associata all'istruzione si chiama *commit*
- Transazione terminata con insuccesso
  - ROLLBACK [WORK]
  - l'azione associata all'istruzione si chiama *abort*

DBG

24

## Commit

- ▷ Azione eseguita quando una transazione termina con successo

## Commit

- ▷ Azione eseguita quando una transazione termina con successo
- ▷ La base di dati è in un nuovo stato (finale) corretto
- ▷ Le modifiche dei dati effettuate dalla transazione divengono
  - permanenti
  - visibili agli altri utenti

## Rollback

- ▷ Azione eseguita quando una transazione termina a causa di un errore
  - per esempio, di un errore applicativo

## Rollback

- ▷ Azione eseguita quando una transazione termina a causa di un errore
  - per esempio, di un errore applicativo
- ▷ Tutte le operazioni di modifica dei dati eseguite durante la transazione sono "annullate"
- ▷ La base di dati ritorna nello stato precedente l'inizio della transazione
  - i dati sono nuovamente visibili agli altri utenti

## Esempio

- ▷ Trasferire la somma 100
  - dal conto corrente bancario  
IT92X0108201004300000322229
  - al conto corrente bancario  
IT32L0201601002410000278976

## Esempio


- ▷ Trasferire la somma 100
    - dal conto corrente bancario  
IT92X0108201004300000322229
    - al conto corrente bancario  
IT32L0201601002410000278976
- START TRANSACTION;

**Esempio**

▷ Trasferire la somma 100

- dal conto corrente bancario  
IT92X0108201004300000322229
- al conto corrente bancario  
IT32L0201601002410000278976

```
START TRANSACTION;
UPDATE Conto-Corrente
SET Saldo = Saldo + 100
WHERE IBAN='IT92X0108201004300000322229';
```

 31

**Esempio**

▷ Trasferire la somma 100

- dal conto corrente bancario  
IT92X0108201004300000322229
- al conto corrente bancario  
IT32L0201601002410000278976

```
START TRANSACTION;
UPDATE Conto-Corrente
SET Saldo = Saldo + 100
WHERE IBAN='IT92X0108201004300000322229';
UPDATE Conto-Corrente
SET Saldo = Saldo - 100
WHERE IBAN= 'IT32L0201601002410000278976';
```

 32

**Esempio**

▷ Trasferire la somma 100

- dal conto corrente bancario  
IT92X0108201004300000322229
- al conto corrente bancario  
IT32L0201601002410000278976

```
START TRANSACTION;
UPDATE Conto-Corrente
SET Saldo = Saldo + 100
WHERE IBAN='IT92X0108201004300000322229';
UPDATE Conto-Corrente
SET Saldo = Saldo - 100
WHERE IBAN= 'IT32L0201601002410000278976';
COMMIT;
```

 33

**Gestione delle transazioni**


Proprietà delle transazioni



**Proprietà delle transazioni**

▷ Le proprietà principali delle transazioni sono


- Atomicity – atomicità

 35

**Proprietà delle transazioni**

▷ Le proprietà principali delle transazioni sono

- Atomicity – atomicità
- Consistency – consistenza

 36

### Proprietà delle transazioni

- ▷ Le proprietà principali delle transazioni sono
- Atomicity – atomicità
  - Consistency – consistenza
  - Isolation – isolamento

### Proprietà delle transazioni

- ▷ Le proprietà principali delle transazioni sono
- Atomicity – atomicità
  - Consistency – consistenza
  - Isolation – isolamento
  - Durability – persistenza (o durabilità)

### Proprietà delle transazioni

- ▷ Le proprietà principali delle transazioni sono
- Atomicity – atomicità
  - Consistency – consistenza
  - Isolation – isolamento
  - Durability – persistenza (o durabilità)
- ▷ Sono riassunte dall'acronimo (inglese) **ACID**

### Atomicità

- ▷ Una transazione è un'unità indivisibile (atomo) di lavoro
- devono essere eseguite tutte le operazioni contenute nella transazione
  - oppure nessuna delle operazioni contenute nella transazione deve essere eseguita
    - la transazione non ha nessun effetto sulla base di dati

### Atomicità

- ▷ Una transazione è un'unità indivisibile (atomo) di lavoro
- devono essere eseguite tutte le operazioni contenute nella transazione
  - oppure nessuna delle operazioni contenute nella transazione deve essere eseguita
    - la transazione non ha nessun effetto sulla base di dati
- ▷ La base di dati non può rimanere in uno stato intermedio assunto durante l'esecuzione di una transazione

### Consistenza

- ▷ L'esecuzione di una transazione deve portare la base di dati
- da uno stato iniziale consistente (corretto)
  - a uno stato finale consistente

### Consistenza

- ▷ L'esecuzione di una transazione deve portare la base di dati
  - da uno stato iniziale consistente (corretto)
  - a uno stato finale consistente
- ▷ La correttezza è verificata dai vincoli di integrità definiti sulla base di dati



43

### Consistenza

- ▷ L'esecuzione di una transazione deve portare la base di dati
  - da uno stato iniziale consistente (corretto)
  - a uno stato finale consistente
- ▷ La correttezza è verificata dai vincoli di integrità definiti sulla base di dati
- ▷ Quando si verifica la violazione di un vincolo di integrità il sistema interviene
  - per annullare la transazione
  - oppure, per modificare lo stato della base di dati eliminando la violazione del vincolo



44

### Isolamento

- ▷ L'esecuzione di una transazione è indipendente dalla contemporanea esecuzione di altre transazioni



45

### Isolamento

- ▷ L'esecuzione di una transazione è indipendente dalla contemporanea esecuzione di altre transazioni
- ▷ Gli effetti di una transazione non sono visibili dalle altre transazioni fino a quando la transazione non è terminata
  - si evita la visibilità di stati intermedi non stabili
    - uno stato intermedio può essere annullato da un rollback successivo
    - in caso di rollback, è necessario effettuare rollback delle altre transazioni che hanno osservato lo stato intermedio (effetto domino)



46

### Persistenza

- ▷ L'effetto di una transazione che ha effettuato il commit è memorizzato in modo permanente
  - le modifiche dei dati eseguite da una transazione terminata con successo sono permanenti dopo il commit



47

### Persistenza

- ▷ L'effetto di una transazione che ha effettuato il commit è memorizzato in modo permanente
  - le modifiche dei dati eseguite da una transazione terminata con successo sono permanenti dopo il commit
- ▷ Garantisce l'affidabilità delle operazioni di modifica dei dati
  - i DBMS offrono meccanismi di ripristino dello stato corretto della base di dati dopo che si è verificato un guasto



48