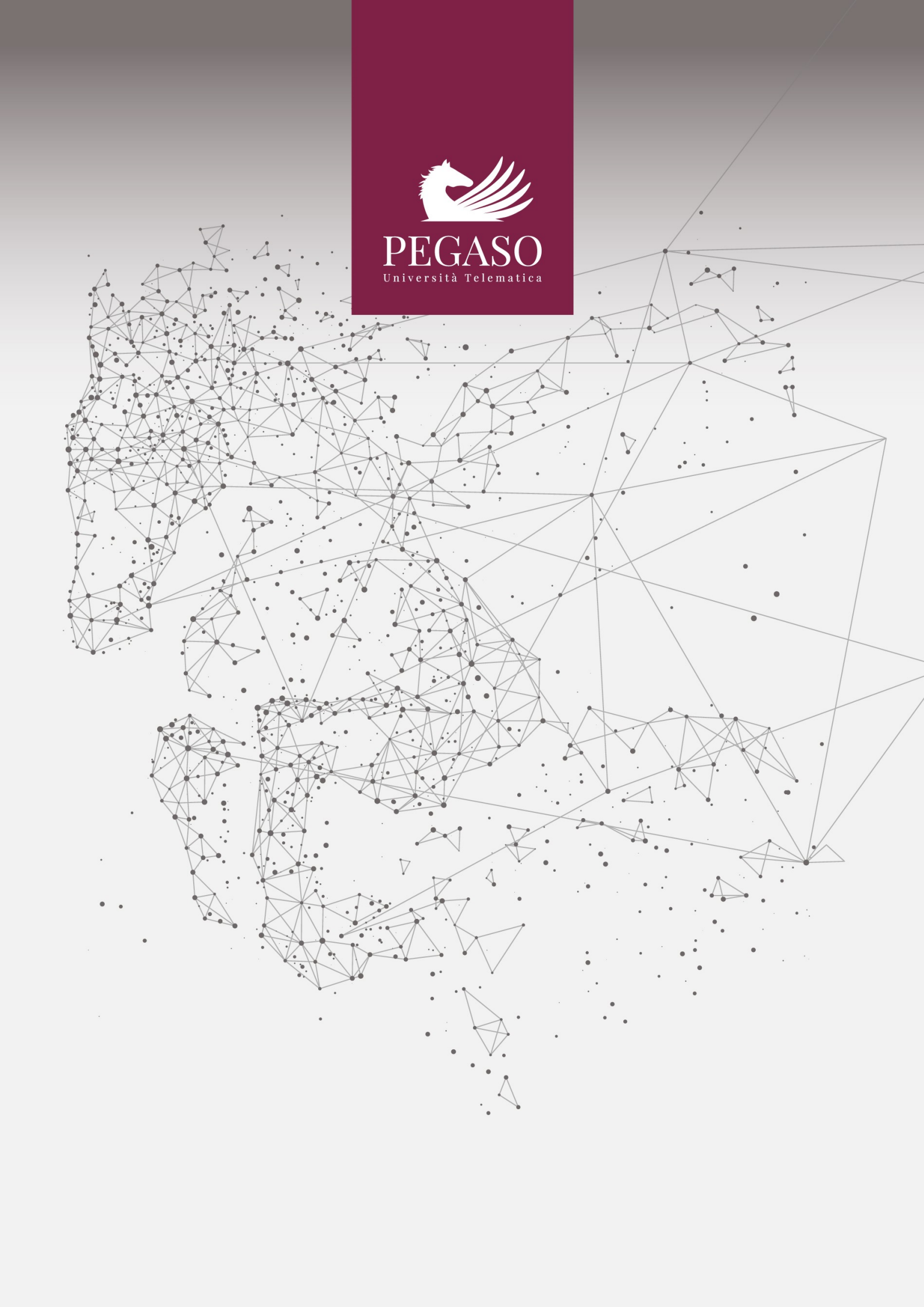
PEGASO
Università Telematica

Indice

1. Risorse e privilegi

I comandi di *Data Control Language* vengono utilizzati per imporre la sicurezza del database in un ambiente di DBMS con più utenti. Due tipi di comandi DCL sono GRANT e REVOKE. Solo l'amministratore del database o il proprietario dell'oggetto del database possono fornire /rimuovere privilegi su un oggetto del database. In questa sezione affronteremo questa problematica. In SQL è possibile specificare chi (utente) e come (lettura, scrittura, …) può utilizzare la base di dati (o parte di essa). Oggetto dei privilegi (diritti di accesso) sono di solito le tabelle, ma anche altri tipi di risorse, quali singoli attributi, viste o domini. Un particloare utente predefinito come **_system** (amministratore della base di dati) ha tutti i privilege. Il creatore di una risorsa ha tutti i privilegi su di essa, compresa la possibilità di cancellare o midificare le tabelle: DROP o ALTER.

1.1 Privilegi

Un **privilegio** è caratterizzato dai seguenti fattori:

- la risorsa cui si riferisce
- l'utente che concede il privilegio
- l'utente che riceve il privilegio
- l'azione che viene permessa
- la trasmissibilità del privilegio

Mentre i tipi di privilegi che un utente può acquisire sono i seguenti:

- *insert*: permette di inserire nuovi oggetti (ennuple)
- *update*: permette di modificare il contenuto
- *delete*: permette di eliminare oggetti
- *select*: permette di leggere la risorsa
- *references*: permette la definizione di vincoli di integrità referenziale verso la risorsa (può limitare la possibilità di modificare la risorsa)
- *usage*: permette l'utilizzo di una risorsa, ad esempio in una definizione (per esempio, di un dominio)

Il commando SQL che concede privilege agli utenti è il commando **GRANT.** La sintassi è la seguente:

GRANT < *Privileges* | all privileges > on *Resource*

to *Users* [with grant option]

Dove la clausola GRANT OPTION specifica se il privilegio può essere trasmesso ad altri utenti.

Il commando SQL che revoca privilege agli utenti è il commando **REVOKE.** La sintassi è la seguente:

REVOKE *Privileges* on *Resource* from *Users*

[restrict | cascade]

Le clausole RESTRICT e CASCADE servono per la propagazione delle revoche.

1.2 Esempi

Vediamo adesso qualche esempio dell'utilizzo dei comandi GRANT e REVOKE.

Esempio 1: GRANT SELECT ON Department TO Stefano

Questo commando concede all'utente Stefano il privilegio di select sulla tabella Dipartimento. La clausola WITH GRANT OPTION, se aggiunta al precedente commando, specifica se deve essere concesso a Stefano anche il provilegio di propagare il privilegio ad altri utenti.

Esempio 2: GRANT ALL PRIVILEGES ON Impiegato to Paolo, Riccardo

Questo commando concede a Paolo e Riccardo tutti i privilegi che possono essere concessi da chi esegue il commando. Riassumendo quindi:

GRANT privilege_name ON object_name TO {user_name |PUBLIC |role_name}

[WITH GRANT OPTION];

Dove:

- *privilege_name* è il diritto di accesso o privilegio concesso all'utente. Alcuni dei diritti di accesso sono ALL, EXECUTE e SELECT
- *Object_name* è il nome di un oggetto di database come TABLE, VIEW, STORED PROC e SEQUENCE
- *user_name* è il nome dell'utente a cui viene concesso un diritto di accesso
- PUBLIC viene utilizzato per concedere i diritti di accesso a tutti gli utenti
- I RUOLI sono un insieme di privilegi raggruppati
- WITH GRANT OPTION: consente a un utente di concedere diritti di accesso ad altri utenti
- Ad esempio: GRANT SELECT ON employee TO user1; Questo comando concede un'autorizzazione SELECT sulla tabella dei dipendenti all'utente1. È necessario utilizzare attentamente l'opzione WITH GRANT perché, ad esempio, se si seleziona GRANT SELECT sulla tabella dei dipendenti su user1 utilizzando l'opzione WITH GRANT user1 può privilegiare la tabella dei dipendenti su ad un altro utente, ad esempio utente2, ecc. Successivamente, se si revoca il privilegio SELECT sul dipendente da user1, ancora user2 avrà il privilegio SELECT sulla tabella dei dipendenti.

Ci chiediamo a questo punto come autorizzare un utente a vedere solo alcune ennuple di una relazione. Questa operazione si può fare attraverso una vista:

- Definiamo la vista con una condizione di selezione
- Attribuiamo le autorizzazioni sulla vista, anziché sulla relazione di base

Il comando REVOKE rimuove i diritti di accesso utente oi privilegi agli oggetti del database. Più in dettaglio, la sintassi per il comando REVOKE è:

REVOKE privilege_name ON object_name

FROM {user_name |PUBLIC |role_name}

Ad esempio: REVOKE SELECT ON employee FROM user1;

Questo comando REVOCA un privilegio SELECT sulla tabella dei dipendenti da user1. Quando si sceglie REVOKE SELECT su una tabella da un utente, l'utente non sarà più in grado di SELEZIONARE i dati da quella tabella. Tuttavia, se l'utente ha ricevuto i privilegi SELECT su quella tabella da più di un utente, può SELEZIONARE da quella tabella fino a quando tutti coloro che hanno concesso l'autorizzazione lo revocano.

Non si possono revocare privilegi non concessi.

2. Le transazioni: commit e rollback

In questa sezione studiamo due importanti proprietà delle transazioni: COMMIT e ROLLBACK. Da un punto di vista generale, una transazione è una sequenza di scambio di informazioni e lavoro correlato (come l'aggiornamento del database) che viene trattato come una unità allo scopo di soddisfare una richiesta e per garantire l'integrità del database. Affinché una transazione venga completata e le modifiche al database diventino permanenti, è necessario completare una transazione nella sua interezza. Una transazione tipica è un ordine di merce del catalogo chiamato da un cliente e inserito in un computer da un rappresentante del cliente. La transazione dell'ordine comporta la verifica di un database di inventario, la conferma della disponibilità dell'articolo, l'inoltro dell'ordine e la conferma della corretta esecuzione dell'ordine e dell'orario previsto per la spedizione. Se viene visualizzata come una singola transazione, tutti i passaggi devono essere completati prima che la transazione abbia esito positivo e il database viene effettivamente modificato per riflettere il nuovo ordine. Se qualcosa accade prima che la transazione sia completata con successo, tutte le modifiche al database devono essere tenute traccia di modo che possano essere annullate.

Un programma che gestisce o supervisiona la sequenza di eventi che fanno parte di una transazione viene talvolta chiamato TRANSACTION MONITOR. Le transazioni sono supportate da SQL per l'utente del database standard e per l'interfaccia di programmazione.

Quando una transazione viene completata correttamente, si dice che le modifiche al database sono state COMMITTED; quando una transazione non viene completata, le modifiche vengono ripristinate (fase di ROLLBACK).

Ad esempio, nel prodotto IBM CICS, una transazione è un'unità di elaborazione dei dati dell'applicazione risultante da un particolare tipo di richiesta di transazione. In CICS, un'istanza di una particolare richiesta di transazione da parte di un operatore di computer o utente viene chiamata un'attività.

Meno frequentemente e in altri contesti informatici, una transazione può avere un significato diverso. Ad esempio, nell'elaborazione batch del sistema operativo mainframe IBM, una transazione è un processo o una fase del lavoro.

A questo punto siamo in grado di approfondire come nel linguaggio SQL evoluto vengono gestite le transazioni.

In SQL, una transazione è *una sequenza di operazioni all'interno di una coppia di istruzioni che ne specifichino l'inizio e la conclusione.*

Esempio di transazione**:**

start transaction

update ContoCorrente

set Saldo = Saldo – 10

where NumeroConto = 12345;

update ContoCorrente

set Saldo = Saldo + 10

where NumeroConto = 55555;

commit work;

Questa transazione aggiorna la tabella ContoCorrente, nel numero di conto corrente 12345 diminuendo di 10 il saldo sul conto stesso, mentre la seconda transazione deposita 10 sul numero di conto corrente 55555.

Una transazione inizia al primo comando SQL dopo la "connessione" alla base di dati oppure alla conclusione di una precedente transazione (lo standard indica anche un comando start transaction, non obbligatorio, e quindi non previsto in molti sistemi).

Una transazione termina con l'istruzione: commit [work].

Le operazioni specificate a partire dall'inizio della transazione vengono eseguite sulla base di dati.

Se per motivi vari si dovesse rinunciare a portare a termine l'operazione, il comando ROLLBACK ripristina le informazioni precedenti. Molti sistemi prevedono una modalità autocommit, in cui ogni operazione forma una transazione.

3. Proprietà acide

ACID (atomicità, consistenza, isolamento e durata) è un acronimo e un trucco mnemonico per l'apprendimento e il ricordo dei quattro attributi principali garantiti a qualsiasi transazione da un gestore transazioni (che è anche chiamato monitor delle transazioni). Il concetto di ACID è descritto in ISO / IEC 10026-1: 1992 Sezione 4. Ciascuno dei quattro attributi può essere misurato rispetto a un benchmark. In generale, tuttavia, un gestore transazioni o un monitor è progettato per realizzare transazioni ACID.

È interessante notare che, in un sistema distribuito, un modo per ottenere transazioni ACID consiste nell'utilizzare un commit a due fasi (2PC), che garantisce che tutti i siti coinvolti debbano impegnarsi al completamento della transazione o che nessuno esegua, e la transazione viene annullata (vedere il rollback). Gli attributi sono: Atomicità, Consistenza, Isolamento e Durabilità (persistenza). Vediamoli in dettaglio.

3.1 ACID

Le proprietà introdotte precedentemente per una transazione ACID sono le seguenti:

- **Atomicità**:
  - Insieme di operazioni da considerare indivisibile ("atomico"), corretto anche in presenza di concorrenza e con effetti definitivi;
  - Unità elementare di lavoro svolta da un'applicazione;
  - Proprietà
    - Correttezza
    - Robustezza
    - Isolamento
- **Consistenza**:
  - La sequenza di operazioni sulla base di dati viene eseguita per intero o per niente. Esempio:
    - trasferimento di fondi da un conto A ad un conto B: o si fanno il prelevamento da A e il versamento su B o nessuno dei due
    - "Durante" l'esecuzione ci possono essere violazioni, ma se restano alla fine allora la transazione deve essere annullata per intero ("abortita")

- **Isolatezza**
    - L'effetto di transazioni concorrenti deve essere coerente (ad esempio "equivalente" all'esecuzione separata)
        - se due assegni emessi sullo stesso conto corrente vengono incassati contemporaneamente si deve evitare di trascurarne uno;
- **Durevolezza (Persistenza)**
    - La conclusione positiva di una transazione corrisponde ad un impegno (in inglese commit) a mantenere traccia del risultato in modo definitivo, anche in presenza di guasti e di esecuzione concorrente.

Bibliografia

- Atzeni P., Ceri S., Fraternali P., Paraboschi S., Torlone R. (2018). Basi di Dati. McGraw-Hill Education.
- Batini C., Lenzerini M. (1988). Basi di Dati. In Cioffi G. and Falzone V. (Eds). Calderini. Seconda Edizione.