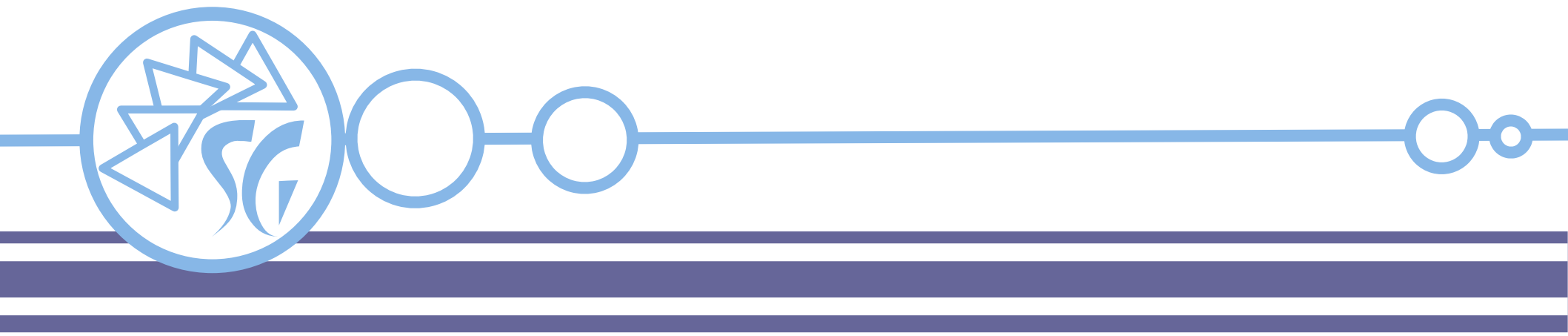

Il Linguaggio SQL
Data Control Language

Ing. Simone Giustetti

www.giustetti.net

DCL

È il sottoinsieme di SQL dedicato all'amministrazione delle utenze ed alla sicurezza

Le forme di autenticazione ammesse sono molteplici, ma raggruppabili in due famiglie: Esterna o Interna

La famiglia interna usa strutture e risorse proprie del RDBMS

La famiglia esterna usa plug-in appositi per demandare l’onere ad altri servizi. Può supportare uno o più sistemi di autenticazione centralizzata

DCL

La sicurezza interna si basa su 3 entità:

- **Utenze**
- **Permessi**
- **Ruoli**

I dati ed i parametri di configurazione sono salvati in apposite tabelle che possono essere interrogate sia con le istruzioni standard che con comandi dedicati

L’uso dei comandi dedicati è consigliato

DCL

Utenze: Utenti che possono autenticarsi e collegarsi al RDBMS e ad alcuni suoi database

Permessi: Suddivisi in permessi di sistema e permessi sui dati. I permessi sui dati sono granulari e possono riguardare le colonne di una tabella

Ruoli: Sono insiemi di permessi raggruppati in base alle funzioni svolte.

- Garantiscono un livello di astrazione
- Semplificano la gestione della sicurezza

Utenze in MariaDB / MySQL

Sono account / login, ossia una coppia dominio + utenza

```
CREATE USER <user> IDENTIFIED BY '<password>';

CREATE USER sg_test_01@test_db_01 IDENTIFIED BY
'sg_test_01';
```

Una volta creata un'utenza può collegarsi al RDBMS mediante il comando:

```
mariadb -u <user> -p <DB>
mysql   -u <user> -p <DB>
```

Utenze in Oracle

Sono una coppia nome utente / password

```
CREATE USER <user> IDENTIFIED BY '<password>';

CREATE USER sg_test_01 IDENTIFIED BY 'sg_test_01';
```

Una volta creata un’utenza bisogna assegnarle il ruolo CONNECT perché possa collegarsi al RDBMS

Modificare Utenze in MariaDB / MySQL

È possibile modificare solo la password di un'utenza

```
SET PASSWORD FOR <user> =
    PASSWORD( '<password>' );
```

MariaDB >= 10.2.0 riconosce la sintassi

```
ALTER USER <user> IDENTIFIED BY '<password>';

ALTER USER CURRENT_USER()
    IDENTIFIED BY '<password>';
```

Modificare Utenze in Oracle

È possibile modificare solo la password di un’utenza

```
ALTER USER <user> IDENTIFIED BY '<password>';
```

Rimuovere un’utenza

Per rimuovere un’utenza si usa il comando **DROP**

```
DROP USER <user>;
DROP USER sg_test_01@test_db_01;  -- MariaDB / MySQL
DROP USER sg_test_01;  -- Oracle
```

L’esistenza di permessi / ruoli assegnati all'utenza causa:

- Fallimento dell'operazione
- Permangono righe spurie nel DB dedicato alla sicurezza

Si consiglia di rimuovere privilegi e/o ruoli prima di rimuovere l'utenza

Permessi di MariaDB / MySQL

Esistono tre categorie principali di permessi:

- Di sistema: USAGE;
- Sulle tabelle: ALTER, CREATE, CREATE TEMPORARY TABLES, DROP, INDEX, TRIGGER;
- Sui dati e/o colonne: DELETE, INSERT, SELECT, UPDATE.

Permessi di Oracle

Esistono quattro categorie principali di permessi:

- Di sistema: CONNECT, RESOURCE, DBA;
- Sulle tabelle: ALL, ALTER, INDEX, REFERENCES;
- Sui dati e/o colonne: DELETE, INSERT, SELECT, UPDATE;
- Sulle procedure: EXECUTE.

Assegnare i Permessi

I permessi si assegnano con il comando **GRANT**:

```
-- MariaDB / MySQL
GRANT <privilege> ... ON <schema>.<table>
   TO <user>@<FQDN>;
-- Oracle
GRANT <privilege> ... ON <schema>.<table> TO <user>;


GRANT ALL PRIVILEGES ON *.* TO 'studiosg'@'localhost';

GRANT ALL PRIVILEGES ON employees.*
    TO sg_test_03@'localhost';

GRANT SELECT ON employees.employees
   TO sg_test_02@'localhost';
```

Revocare i Permessi

I permessi si revocano con il comando **REVOKE**:

```
-- MariaDB / MySQL
REVOKE <privilege> ... ON <schema>.<table>
    FROM <user>@<FQDN>;
-- Oracle
REVOKE <privilege> ... ON <schema>.<table>
    FROM <user>;

REVOKE ALL PRIVILEGES ON *.*
    FROM 'studiosg'@'localhost';

REVOKE ALL PRIVILEGES ON employees.*
    FROM sg_test_03@'localhost';

REVOKE SELECT ON employees.employees FROM sg_test_02;
```

Ruoli

I ruoli sono creati e rimossi come qualsiasi altro oggetto di database

```
CREATE ROLE <role>;

DROP ROLE <role>;
```

I permessi sono gestiti come per le utenze

```
GRANT <privilege> ON <schema>.<table> TO <role>;

REVOKE <privilege> ON <schema>.<table> FROM <role>;
```

Assegnazione dei Ruoli in MariaDB / MySQL

I ruoli devono essere assegnati alle utenze

```
SET DEFAULT ROLE <role> FOR <user>@<FQDN>;

-- Remove default role
SET DEFAULT ROLE NONE FOR <user>@<FQDN>;

-- Not enabled by default
GRANT <role> TO <user>@<FQDN>;
```

Assegnazione dei Ruoli in Oracle

I ruoli devono essere assegnati alle utenze

```
ALTER USER <user> DEFAULT ROLE <role>;

-- Remove default role
ALTER USER <user> DEFAULT ROLE NONE;

-- Not enabled by default
GRANT <role> TO <user>;
```

Ruolo Predefinito

Il ruolo di default è assunto immediatamente dopo l'autenticazione. Non è necessaria alcuna azione per abilitarlo

Gli atri ruoli devono essere assunti esplicitamente

```
-- User enables a role (MariaDB / MySQL)
SET ROLE <role>;

-- User enables a role (Oracle)
SET ROLE <role>;
```

I ruoli possono essere assegnati ad altri ruoli

Utenze in PostgreSQL

L’autenticazione in PostgreSQL è gestita mediante i **ruoli**. **Non esistono utenze**.

Un ruolo equivale ad una utenza oppure ad un gruppo di utenze, a seconda di come sia configurato.

Ogni ruolo può possedere oggetti creati nel database.

Ogni ruolo può assegnare permessi sui propri oggetti ad altri ruoli.

É possibile rendere un ruolo membro di un altro in modo che ne erediti i privilegi.

Ruoli di PostgreSQL

L'elenco dei ruoli definiti è visualizzabile interrogando la vista di sistema **pg_roles**.

```
SELECT rolname FROM pg_roles;
```

È necessario applicare un filtro per elencare solo quelli aventi il permesso di collegarsi al database:

```
SELECT rolname FROM pg_roles WHERE rolcanlogin;
```

Ogni database contiene un ruolo predefinito per l'utenza di sistema che ha eseguito il comando **initdb** (Solitamente **postgres**).

Collegarsi a PostgreSQL

Ci si collega al database specificando un ruolo.

Il ruolo usato per collegarsi imposta i permessi predefiniti di chi si collega.

Collegandosi con **psql**, il ruolo viene specificato con l'opzione *-U*.

Se non venisse specificato nulla, il client usa l'utenza del sistema operativo.

Il ruolo di sistema **LOGIN**, permette di collegarsi ad un database.

Permessi di Sistema di PostgreSQL

I permessi predefiniti vengono assegnati all'atto della creazione di un ruolo:

```
CREATE ROLE <ruolo> <ruolo di sistema>, ...;

CREATE ROLE test LOGIN;
```

Ruoli di una Istanza PostgreSQL

RUOLI “AMMINISTRATIVI” DI UNA ISTANZA POSTGRESQL	
Ruolo	**Descrizione**
LOGIN	Consente di collegarsi ad un database.
CREATEDB	Consente di creare nuovi database.
CREATEROLE	Consente di create nuovi ruoli.
REPLICATION	Configura una replica dei dati tra server. Richiede anche il ruolo LOGIN.
SUPERUSER	Ruolo per amministrare il sistema.

Ruoli di PostgreSQL Protetti da Password

È possibile proteggere un ruolo assegnando una password. Consente di entrare in un database solo a seguito dell'avvenuta autenticazione.

Le password di PostgreSQL sono separate da quelle del sistema operativo e salvate nel catalogo dell'istanza.

```
CREATE ROLE <ruolo> PASSWORD '<password>';
```

Ereditarietà dei Permessi

Un ruolo eredita automaticamente tutti i permessi dai ruoli di cui fa parte.

Per creare un ruolo che non erediti i privilegi si usa l'istruzione **NOINHERIT**.

```
CREATE ROLE <ruolo> NOINHERIT;
```

L'ereditarietà può essere manipolata per ogni comando **GRANT** specificando l'opzione:

WITH INHERIT [**FALSE** | **TRUE**]

Aggirare i Permessi dei Ruoli

Un ruolo può essere eletto ad amministrativo ed aggirare i permessi a basso livello dei ruoli mediante l'istruzione **BYPASSRLS**.

```
CREATE ROLE <ruolo> BYPASSRLS;
```

Limitare le Connessioni in PostgreSQL

È possibile porre un limite al numero di connessioni concorrenti effettuabili da un ruolo mediante l'istruzione **CONNECTION LIMIT**.

```
CREATE ROLE <ruolo> CONNECTION LIMIT <integer>;
```

Assegnare / Rimuovere Ruoli in PostgreSQL

I ruoli in PostgreSQL si amministrano attraverso le istruzioni **GRANT** e **REVOKE**.

```
GRANT ROLE <ruolo_1> TO <ruolo_2>, ...;

REVOKE ROLE <ruolo_1> FROM <ruolo_2>, ...;
```

PostgreSQL impedisce che vengano effettuate assegnazioni circolari.

Assumere un Ruolo in PostgreSQL

Ad autenticazione avvenuta un utente assume i permessi dei ruoli di cui fa parte, che gli siano stati assegnati con l'ereditarietà.

Un ruolo può assumere i permessi di un altro di cui sia membro utilizzando l'istruzione **SET**.

```
SET ROLE <ruolo>;
```

A seguito del comando il ruolo usufruirà dei soli permessi assegnati a <ruolo> e perderà, invece, i permessi di tutti gli altri ruoli di cui sia membro e che gli siano stati assegnati con l'ereditarietà.

Eliminare un Ruolo in PostgreSQL

Tutti gli oggetti di un ruolo devono essere preventivamente rimossi o assegnati ad altro ruolo.

```
ALTER <ENTITY> <entità> OWNER TO <ruolo>;

ALTER TABLE anagrafica_assenza OWNER TO sg;

REASSIGN OWNED BY <ruolo_1> TO <ruolo_2>;

DROP OWNED BY <ruolo_1>;
```

I comandi devono essere eseguiti per ogni database.

```
DROP ROLE <ruolo>;
```

Permessi in PostgreSQL

Esistono diverse categorie principali di permessi:

- Di sistema: LOGIN, ...;
- Sui database: ALL, CONNECT, CREATE, TEMP, TEMPORARY;
- Sulle tabelle: ALL, INSERT, REFERENCES, SELECT, UPDATE;
- Sui dati e/o colonne: ALL, DELETE, INSERT, REFERENCES, SELECT, TRIGGER, TRUNCATE, UPDATE;
- Su funzioni e procedure: ALL, EXECUTE;
- Sulle sequenze: SELECT, UPDATE, USAGE.

Ruoli di MS SQL Server

In Ms SQL Server gli account di sistema devono essere rimappati in utenti o ruoli del server

Non necessitano di una seconda autenticazione

È possibile creare utenti interni e SQL Server si occupa direttamente della loro autenticazione. È una pratica sconsigliata perché non usa la crittografia

È possibile rimappare anche i gruppi di Windows. Ogni membro di un gruppo viene autenticato automaticamente da SQL Server

Ripristinare il Ruolo Iniziale in PostgreSQL

A seguito di una o più istruzioni **SET**, la situazione iniziale può essere ripristinata utilizzando l'istruzione **RESET ROLE**.

```
RESET ROLE;
```

A seguito del comando, il ruolo usufruirà dei propri permessi e di quelli di tutti i ruoli che gli siano stati assegnati con l'ereditarietà.

Utenze Predefinite di MS SQL Server

- **dbo**: Database Owner / Proprietario del Database
- **guest**: Consente l'accesso al server anche per account non mappati su nessuna utenza del Database
- **sa**: Amministratore del servizio

Ruoli Predefiniti di MS SQL Server

- **db_owner**
- **public**: Ruolo assegnato di ufficio ad ogni utente di un database. I privilegi di public sono concessi a tutte le utenze. Non può essere rimosso
- **sysadmin**: È il ruolo di sa. Ha privilegi illimitati. Tutti gli amministratori di dominio ricevono il ruolo sysadmin

Gestione dei Ruoli in Ms SQL Server

I ruoli devono essere assegnati ad un'utenza che ha il compito di amministrarli

Solo l'amministratore può aggiungere o rimuovere utenze al / dal ruolo

I permessi possono essere ereditati da altre utenze o altri gruppi

REVOKE: Revoca i privilegi di un utenza / ruolo, ma non impedisce che i permessi possano essere ereditati
DENY: Impedisce che i permessi possano essere assunti per qualsiasi ragione

Creare Utenze in Ms SQL Server

Creazione / mappatura di utenze

```
CREATE USER <user> IDENTIFIED BY '<password>';

CREATE USER <user> FROM <login>;

CREATE USER <user> FROM <FQDN>\<login>;
```

Creare Ruoli in Ms SQL Server

Creazione di un ruolo

```
CREATE ROLE <role> AUTHORIZATION <user>;
```

Assegnazione dei Ruoli in Ms SQL Server

Le utenze devono essere aggiunte ad un ruolo e non viceversa

```
ALTER ROLE  <role>  {
   ADD MEMBER <user>
   |  DROP MEMBER <user>
   |  WITH NAME = <new_name>
};
```

```
CREATE ROLE commerciale;
```

```
ALTER  ROLE commerciale ADD  MEMBER sergio_de_sergi;
```

```
ALTER  ROLE commerciale DROP MEMBER sergio_de_sergi;
```

Assegnazione dei Permessi in Ms SQL Server

I permessi sono diversi dallo standard

ALL PERMISSIONS assegna solo una parte dei permessi.
DENY sulla tabella non ha la precedenza su GRANT per una colonna.

```
GRANT SELECT ON <schema>.<table_name> TO <user>;
```

Informazioni & Licenze