LU12c - Rechte & Rollen: vergeben, anzeigen, entziehen

Ziel: Mit **GRANT** Rechte auf passenden Ebenen vergeben, mit **SHOW GRANTS** prüfen, mit **REVOKE** entziehen. Wir bilden WordPress-Rollen **sinngemäss** auf **MySQL-Rollen** ab:

WP-Rolle (App)	Sinnbild DB-Rolle (MySQL)	Typische DB-Rechte (Beispiel)
Administrator	`wp_admin`	**ALL PRIVILEGES** auf `wttw.*`
Redakteur (Editor)	`wp_editor`	`SELECT, INSERT, UPDATE, DELETE` auf **Inhaltstabellen** (`posts`, `comments`), `SELECT` auf `users`
Autor	`wp_author`	`SELECT, INSERT, UPDATE` auf `posts` **eigener Inhalte** (fachlich; DB-seitig vereinfachen wir), `SELECT, INSERT` auf `comments`, `SELECT` auf `users`
Leser/Integration	`wp_read`	`SELECT` auf `wttw.*`

Hinweis: „Eigene Inhalte“ lässt sich **fachlich** (App-Logik) sauber lösen; auf reiner DB-Ebene bräuchte es z. B. **Row-Level-Security** oder Trigger. Für die Übung vereinfachen wir auf Tabellen-Ebene.

0) Ebenen-Überblick

Ebene	Schreibweise	Wirkung (Beispiel)
Datenbank	`wttw.*`	alle Tabellen in `wttw`
Tabelle	`wttw.posts`	nur diese Tabelle
Spalte	`users(email)`	nur bestimmte Spalte(n)

1) Rollen anlegen & berechtigen (einmalig als root)

```
-- Rollen
CREATE ROLE IF NOT EXISTS wp_admin, wp_editor, wp_author,
wp_read;
```

```
-- Rechte an Rollen vergeben
GRANT ALL PRIVILEGES                ON wttw.*        TO
wp_admin;

GRANT SELECT                        ON wttw.users    TO
wp_editor, wp_author;
GRANT SELECT, INSERT, UPDATE, DELETE ON wttw.posts   TO
wp_editor;
GRANT SELECT, INSERT, UPDATE        ON wttw.posts    TO
wp_author;

GRANT SELECT, INSERT, DELETE        ON wttw.comments TO
wp_editor;
GRANT SELECT, INSERT                ON wttw.comments TO
wp_author;

GRANT SELECT                        ON wttw.*        TO
wp_read;
```

2) Rollen Benutzern zuweisen & Standardrollen setzen

```
-- Benutzer aus Seite 1:
-- 'caro_admin'@'localhost', 'martin_admin'@'localhost',
'tran_editor'@'localhost', 'wp_api'@'localhost'

GRANT wp_admin  TO 'caro_admin'@'localhost',
'martin_admin'@'localhost';
GRANT wp_editor TO 'tran_editor'@'localhost';
GRANT wp_read   TO 'wp_api'@'localhost';

-- Standardrollen aktiv bei Login
SET DEFAULT ROLE ALL TO
  'caro_admin'@'localhost',
  'martin_admin'@'localhost',
  'tran_editor'@'localhost',
  'wp_api'@'localhost';
```

3) Rechte prüfen: SHOW GRANTS

```
SHOW GRANTS FOR 'tran_editor'@'localhost';
-- Rolle "ausklappen":
```

```
SHOW GRANTS FOR 'tran_editor'@'localhost' USING wp_editor;

SHOW GRANTS FOR 'wp_api'@'localhost';
SELECT CURRENT_ROLE();  -- aktive Rollen der aktuellen
Session
```

4) Funktions-Tests

Als `tran_editor` (Redakteur) verbinden

```
USE wttw;

-- darf Inhalte pflegen:
INSERT INTO posts(author_id, title, STATUS) VALUES
(3,'Utrecht – 10 Sehenswürdigkeiten','draft'); -- □
UPDATE posts SET STATUS='published', published_at=NOW()
WHERE post_id=LAST_INSERT_ID();              -- □
INSERT INTO comments(post_id, author, body) VALUES
(1,'Leser','Toller Beitrag!');                  -- □

-- kein DDL:
DROP TABLE posts; -- □ (erwarteter Fehler)
```

Als `wp_api` (read-only Integration) verbinden

```
USE wttw;
SELECT COUNT(*) FROM posts;          -- □
INSERT INTO posts(author_id,title) VALUES (1,'hack'); -- □
```

5) REVOKE – Rechte entziehen (an Benutzer oder Rolle)

Temporär Kommentare sperren (alle Redakteure betroffen, da über Rolle):

```
-- als root: an der Rolle entziehen
REVOKE INSERT, DELETE ON wttw.comments FROM wp_editor;

-- Prüfung:
SHOW GRANTS FOR 'tran_editor'@'localhost' USING wp_editor;

-- Rückgängig machen:
GRANT INSERT, DELETE ON wttw.comments TO wp_editor;
```

Gute Praxis:

- **Least-Privilege** umsetzen (wp_read/wp_author/wp_editor/wp_admin).
- Host **nicht** mit `%` freigeben, wenn nicht nötig.
- Standardrollen setzen (`SET DEFAULT ROLE`).
- Rechte regelmässig prüfen (`SHOW GRANTS`).

</WRAP>