Access control: DCL (Data Control Language)

Roles

PMOS A role is a named bundle of privileges and/or other roles, that can be granted to a user or role.

P OS The built-in PUBLIC role is granted to all users. (M: no PUBLIC role)

PMOS CREATE ROLE role1; SQL:1999; Std: <role definition>

PMOS DROP ROLE [IF EXISTS] role1; SQL:1999; Std: <drop role statement>
(PMS)

Granting/revoking permissions and roles

user1, role1, or PUBLIC (M: no PUBLIC role)
(P: a user is actually a role)

PMO GRANT role1 [,...] TO <grantee> [,...] [WITH ADMIN OPTION] Std: <grant role stmt.>

the grantee(s) can grant the role(s)/privilege(s) to others

PMOS GRANT <privileges> TO <grantee> [,...] [WITH GRANT OPTION] Std: <grant privilege stmt.>

PMO REVOKE [ADMIN OPTION FOR] role1 [,...] FROM <grantee> [,...] {CASCADE|RESTRICT} Std: <revoke role stmt.>

remove only ability to grant to others
(MO: not implemented)

If the grantee(s) granted the role(s)/privilege(s) **to others** thanks to "WITH ADMIN/GRANT OPTION" (possibly recursively) then:
a) CASCADE revokes also them, b) RESTRICT raises error (P: it is default)
(MO: no CASCADE/RESTRICT keywords - DBMS does nothing with these role(s)/privilege(s), except Oracle which works for privileges as CASCADE keyword)

PMOS REVOKE [GRANT OPTION FOR] <privileges> FROM <grantee> [,...] {CASCADE|RESTRICT} Std: <revoke privilege stmt.>

(S: no RESTRICT; see doc!)

S ALTER ROLE role1 {ADD|DROP} MEMBER {user1|role1}

The most important standard object <privileges> (no system privileges, e.g. for creating tables):

PMOS • {<action> [,...] | ALL PRIVILEGES} ON table1/view1 - allows ≥ 1 or all action(s) on the table/view
The <action> can be:

PMOS - SELECT/INSERT/UPDATE/DELETE - allows to perform a DML operation on table1/view1 SQL-86

PMOS - REFERENCES - allows to create a FK constraint referencing table1; SQL-92
only needed for CREATE/ALTER TABLE;
(O: "REVOKE REFERENCES ..." needs "CASCADE CONSTRAINTS" option - see doc)

PM - TRIGGER - allows to create a trigger on table1 SQL:1999

PMOS The SELECT/INSERT/UPDATE/REFERENCES (S: not INSERT) privilege can be also granted per column(s) by writing "(col1 [col2, ...])" after the <action>, but granting on a view is recommended if feasible, instead.

PMOS • EXECUTE ON routine1 - allows to run the procedure/function (P: granted by default to PUBLIC) SQL:1999

P • USAGE ON sequence1 - allows to use the sequence generator SQL:2003
(P: also required for the "table1_col1_seq" tied to SERIAL column!)
(O: "SELECT ON sequence1") (S: often not needed, see doc)

A keyword in <privileges> after "ON"?
(PM: TABLE/SEQUENCE is optional, PROCEDURE/FUNCTION is required)
(OS: not allowed)

SQLite doesn't implement GRANT/REVOKE nor CREATE/DROP role1, because they would be meaningless for an embedded DB.

The SQL Standard doesn't define:
- creating and deleting users,
- built-in users and roles,
- system privileges (it defines only object privileges),

They are described in the "*DBMS organization - DB, schema, user, connecting, SET ROLE, referencing objects, synonym, tablespace.txt*" and "*Access control - <name of DBMS> specifics.txt*" files.