Database Management Systems

Security, Views, and Other Topics

Security

- Based on the existence of authorization ID's (user names).
- Privileges can be granted to or revoked from authorization IDs on database elements (objects).
- Common database elements include relations/tables, views, sequences, stored procedures, etc.
- Two aspects about the privileges
    - how they are created initially: Whoever created the database element has all possible privileges on this element.
    - how they are passed from user to user: Granting

Privileges

- select
- insert
- delete
- update
- references
- usage: the right to use that element in one's own declarations
- trigger: the right to define triggers on that relation
- execute: the right to execute a piece of code
- under: the right to create subtypes of a given type

Granting Privileges

- SQL Grant statement can let a user "copy" a privilege to another user
- SQL Statement Syntax:
  ```
  GRANT <privilege list> ON <db element>
  TO <user list> [WITH GRANT OPTION];
  ```
- privilege list: an option is ALL PRIVILEGES, that means all the privileges that the grantor may legally grant on the db element in question.
- db element: usually a relation (base table or view). If it is another kind of element, the name of the element is preceded by the type of that element.
- The special user PUBLIC means all users.
- SQL Statement Example:
  ```
  Grant select, update On HR.Employees To usera, userb With Grant Option;
  Grant delete On HR.Employees To userb;
  ```

Revoking Privileges

- a granted privilege can be revoked at any time.
- SQL statement:
  ```
  REVOKE <privilege list> ON <db element>
  FROM <user list> CASCADE|RESTRICT;

  REVOKE GRANT OPTION FOR <privilege list> ON <db element>
  FROM <user list> CASCADE|RESTRICT;
  ```
- Example:
  ```
  Revoke Grant Option For select On HR.Employees From usera Cascade;
  Revoke update On HR.Employees From usera Restrict;
  ```
- In the second statement: The core privileges themselves remain, but the option to grant them to others is removed.
- CASCADE: revoke any privileges that were granted only because of the revoked privileges.
- RESTRICT: if the privilege has passed on by the user, the revoke with RESTRICT option would fail. You'll be forced to use CASCADE option.

View

- Base table: created by create table statement, physically exists, persistent, won't change because other relation's change
- Virtual Views: relations defined by a query over other relations
- virtual views are not stored, but can be queried as if they existed.
- Views can also be materialized.
- Declaring view:

  ```
  CREATE VIEW view_name (attribute list) AS (view-definition-SQL);
  Create View Dept_Budget (dname, totalSalary)
  AS (select dname, sum(salary) as totalSalary
        from  Departments join Emps on did = workdept
        group by dname);
  ```

- Querying view: the same as a base table. During the query processing time, the view would be replaced by its definition in order to execute the query.

  ```
  select dname from dept_budget where totalSalary > 100000;
  ```

- removing view:

  ```
  DROP VIEW view_name;
  ```

Sequence

Syntax of creating a sequence:

```
CREATE SEQUENCE <sequence_name>
   Start With <integer>
   Increment By <integer>
   Order | NoOrder
   Cycle | NoCycle
   Maxvalue <integer> | NoMaxvalue
   Minvalue <integer> | NoMinvalue;
```

Example:

```
Create Sequence AutoProjectNo
   Start With 1000
   Increment by 1
   Order
   NoCycle
   No Maxvalue
   MinValue 1000;

Create Sequence ConfirmationNo
   Start With 1000
   Increment by 4
   NoOrder
   Cycle
   Maxvalue 9999
   MinValue 10000;
```

Trigger

- An example of trigger:

```
CREATE OR REPLACE TRIGGER  "projectNumber"
    before insert on Projects
        for each row
            when (NEW.projectNo is null)
    begin
         select AutoProjectNo.nextval into :NEW.projectNo from dual;
    end;
/
```

- firing Point: Before/After
- Options: insert/delete/update
- on <table-name>
- For each Row or For each statement
- when (bool condition)
- Begin body End;
  body can include multiple sql statements.