Data Advanced

Schema’s

DE HOGESCHOOL MET HET NETWERK

Elfde-Liniestraat 24, 3500 Hasselt
www.pxl.be

- Schema’s
- Tablespace
- Objecten

Schema’s

- Een schema is verbonden aan een user
- Schema aanmaken gebeurt **NIET** met het CREATE SCHEMA statement
- Schema wordt aangemaakt **wanneer de user wordt aangemaakt**
- Een schema gebruikt tablespace(s)

CREATE SCHEMA

- Wordt **NIET** gebruikt om een schema aan te maken
- **WEL** om een bestaand schema op te vullen met objecten
- Kan ook met aparte CREATE statements
- **Je kan geen schema maken voor een andere gebruiker!**
- Naam van het schema moet gelijk zijn aan de gebruikersnaam

```
CREATE SCHEMA AUTHORIZATION schema_name
    [create_table_statement]
    [create_view_statement]
    [grant_statement];
```

CREATE SCHEMA - voorbeeld

```
CREATE SCHEMA AUTHORIZATION JANE
    CREATE TABLE products
      ( product_id number(10) not null,
        product_name varchar2(50) not null,
        category varchar2(50),
        CONSTRAINT products_pk PRIMARY KEY (product_id)
       );
    CREATE TABLE suppliers
      ( supplier_id number(10) not null,
        supplier_name varchar2(50) not null,
        city varchar2(25),
        CONSTRAINT suppliers_pk PRIMARY KEY (supplier_id)
       );
```

SCHEMA vs USER vs DATABASE

- Een user is een account om te verbinden met een database
- Een schema is verbonden met een user
- Een database is het geheel van alle users, schema’s, systeemtabellen, e.d.

User aanmaken

- Een gebruiker wordt aangemaakt met het CREATE USER command

  CREATE USER <username> IDENTIFIED BY "<password>";

- Stel dat we de user *Guus* willen aanmaken met paswoord *p@ssw0rd*

  CREATE USER guus identified by “p@ssw0rd”;

- Inloggen met deze gebruiker lukt helaas nog niet

  Status : Failure -Test failed: ORA-01045: user GUUS lacks CREATE SESSION privilege; logon denied

- Reden is dat een nieuwe user geen enkel recht heeft om iets te doen!

GRANT privileges

- Om dit op te lossen moeten we de gebruiker rechten geven

  GRANT <privilege> TO <user>

- Om in te loggen moeten we het *Create session* privilege toekennen

  GRANT create session TO guus

- Geef de gebruiker rechten om een tabel aan te maken

  GRANT create table TO guus

- Maar toch krijgen we een foutmelding

  ORA-01950: no privileges on tablespace 'SYSTEM'

GRANT privileges

- Oplossing hiervoor is een quota geven op de tablespace
  - Vraag de gebruikte tablespace op (voer uit met de SYSTEM user!)

    ```
    select default_tablespace from dba_users
    where  username = 'GUUS';
    ```

    Het resultaat van deze query gebruiken als naam van de tablespace hieronder
  - Geef het gewenste quota (of ongelimiteerd)

    ```
    ALTER USER guus QUOTA unlimited ON system;
    ```
- Nu kan gebruiker *guus* tabellen aanmaken en data inserten
  - ```
    CREATE TABLE Klanten(naam VARCHAR(20));
    INSERT INTO Klanten VALUES ('Hogeschool');
    INSERT INTO Klanten VALUES ('PXL');
    ```
- Daarnaast kan je ook rechten geven om views, procedures, sequences, … aan te maken
  - GRANT create view, create procedure, create sequence to guus;
    ```

GRANT privileges

- Let op: rechten geven om een DROP command uit te voeren bestaat niet
- Reden: database users hebben altijd alle privileges op hun eigen objecten
- Gevaar: eender welke applicatie die deze database user gebruikt om te verbinden met de DB kan SQL injection doen!
- Oplossing: maak een tweede user aan die enkel *create session* mag doen

  Het aanmaken en rechten toekennen kan ook in één statement

  GRANT create session to guus_app identified by “St3rkP@ssw00rd”;
- Deze application user moet nog rechten krijgen op de tabel(len) van gebruiker *guus.* Herhaal dit voor ELK gewenst object!

  GRANT select, insert, update, delete on guus.klant to guus_app;

GRANT privileges

- Ondanks het gebruik van een application user is onze DB nog steeds niet waterdicht beveiligd!
- Iedereen met toegang tot het netwerk kan als *guus* verbinden.
- Oplossing: account locken
  ALTER USER guus ACCOUNT LOCK;
- Maar deze oplossing is niet werkbaar. Soms moeten we zelf als *guus* inloggen, dus account unlocken, werk uitvoeren en niet vergeten terug te locken, enz.

GRANT privileges

- Nog een veel grote probleem is dat we hackers een zicht geven op de gebruikersnamen. Oracle meldt namelijk of een account gelocked is. ORA-28000: the account is locked
- Hackers kunnen dus je systeem doorlopen op eventueel gekende usernames en default(?) paswoorden proberen.
- Oplossing is schema-only accounts
  - Buiten scope van de cursus (gaat niet in versie 11G)

REVOKE privileges

- Een toegewezen privilege kan ook terug weggenomen worden
- Gebruik hiervoor het REVOKE command
  REVOKE <privilege> FROM <user>
- Voor systeem privileges voor je bijvoorbeeld onderstaande uit:
  REVOKE create table FROM guus;
- Voor object privileges bijvoorbeeld:
  REVOKE select on guus.klant FROM guus_app;

DROP user

- Een gebruiker kan je makkelijk verwijderen met het DROP USER command
  DROP USER <username>;
- Dit kan enkel als de gebruiker niet verbonden is met de database!
- Als een gebruiker objecten heeft aangemaakt in zijn schema (tabellen, e.d.), kan deze ook niet verwijderd worden.
- Eventueel kan dan de optie CASCADE worden meegegeven
  DROP USER guus CASCADE;
- Wees voorzichtig hiermee, dit verwijdert ALLE data van de gebruiker, zonder enige waarschuwing!

Roles

- Een role is een groep van privileges
- Kan gebruikt worden als je telkens dezelfde privileges wil toepassen op gebruikers
- Voorbeeld: de rol “Application developers” waarin je de rechten create table en create procedures geeft

```
CREATE ROLE role_name
[ NOT IDENTIFIED |
IDENTIFIED {BY password | USING [schema.] package | EXTERNALLY | GLOBALLY } ;
```

Roles

- Vervolgens kan je table privileges en/of functie privileges (EXECUTE) toekennen aan de rol

  GRANT privileges ON object TO role_name

Privilege	Description
SELECT	Ability to perform SELECT statements on the table.
INSERT	Ability to perform INSERT statements on the table.
UPDATE	Ability to perform UPDATE statements on the table.
DELETE	Ability to perform DELETE statements on the table.
REFERENCES	Ability to create a constraint that refers to the table.
ALTER	Ability to perform ALTER TABLE statements to change the table definition.
INDEX	Ability to create an index on the table with the create index statement.
ALL	All privileges on table.

Roles

- Laatste stap is de rol toekennen aan gebruikers

  GRANT role_name TO user_name;

- Een role kan enabled/disabled worden

  SET ROLE

  (role_name [IDENTIFIED BY password] | ALL [EXCEPT role1, role2, ...] | NONE);

- Wanneer een gebruiker inlogt worden alle default rollen enabled
- Alle non-default rollen moeten enabled worden (met het SET ROLE statement)
- Een default rol is altijd enabled wanneer men inlogt (SET ROLE statement is hiervoor niet nodig)

Roles

- Om een rol als default in te stellen gebruikt je het ALTER USER statement

  ALTER USER user_name

  DEFAULT ROLE

  (role_name | ALL [EXCEPT role1, role2, ...] | NONE);

- Om een rol te verwijderen gebruik je het DROP ROLE statement

  DROP ROLE role_name;

Roles - voorbeeld

- Rol aanmaken
  CREATE ROLE app_dev IDENTIFIED BY test123;
- Rechten geven op tabel medewerkers
  GRANT select, insert, update, delete ON medewerkers TO app_dev;
- Rechten afnemen
  REVOKE delete ON medewerkers FROM app_dev;
- Of alle rechten afnemen op een object
  REVOKE all ON medewerkers FROM app_dev;
- Rol toewijzen aan gebruiker
  GRANT app_dev TO john;
- Rol activeren
  SET ROLE app_dev IDENTIFIED BY test123;

Roles - voorbeeld

- Default rol instellen voor gebruiker John
  ALTER USER john
  DEFAULT ROLE app_dev;
- Alle rollen van gebruiker John als default instellen
  ALTER USER john
  DEFAULT ROLE all;
- Alle rollen van gebruiker John als default instellen, behalve app_dev
  ALTER USER john
  DEFAULT ROLE all
  EXCEPT app_dev;
- Rol verwijderen
  DROP ROLE app_dev;

Oefeningen!