DDL - Create, Alter, Drop; DAS - Grant, Revoke

11. marca 2015

Skupiny DDL príkazov

1. CREATE - vytvorenie DB objektu
2. ALTER - modifikácia DB objektu
3. DROP - zrušenie DB objektu

Databázové objekty

- TABLE
- INDEX
- SEQUENCE
- SYNONYM
- VIEW
- FUNCTION
- PROCEDURE
- TRIGGER

- DATABASE
- USER
- TABLESPACE
- ...

Dátové typy

Skalárne dátové typy:

- Reťazec
  - CHAR (dlzka)
  - VARCHAR2 (dlzka)
- Číslo
  - NUMBER
  - NUMBER(dlzka)
  - NUMBER(dlzka, pocet_des)
- Dátum / čas
  - DATE
  - TIMESTAMP
  - INTERVAL

- **Veľké dátové typy:**
  - BLOB - binary large object
  - CLOB - character large object
  - XMLTYPE - xml objetkový typ
- **Užívateľom definované**
  - Kolekcie
  - Recordy
  - Objekty

Vytvorenie tabuľky

Syntax

```
CREATE TABLE nazov_tabulky
(
   nazov_stlpca    datovy_typ [DEFAULT default_hodnota]  [NOT NULL]  row_constraint,
   ...
   table_constraint
)
```

- row_constraint:
  - `PRIMARY KEY`
  - `UNIQUE`
  - `REFERENCES nazov_tabulky`
  - `CHECK (nazov_stlpca podmienka) [DISABLE]`
- table_constraint:
  - `[ CONSTRAINT nazov_constraintu ] PRIMARY KEY ( zoznam_PK_stlpcov )`
  - `[ CONSTRAINT nazov_constraintu ] UNIQUE ( zoznam_PK_stlpcov )`
  - `[ CONSTRAINT nazov_constraintu ] FOREIGN KEY ( zoznam_FK_stlpcov )`
    `REFERENCES nazov_PK_tabulky [(zoznam_PK_stlpcov)]`
  - `[CONSTRAINT nazov_constraintu ] CHECK (nazov_stlpca podmienka) [DISABLE]`

Príklad vytvorenie tabuľky

```
create table ou
( rod_cislo    char(11)  primary key,
  meno         varchar2(20) not null,
  priezvisko   varchar2(20) not null,
  login        varchar2(20) not null  UNIQUE  ,
  dat_zapisu   date         DEFAULT  sysdate ,
  typ_osoby    char(1)      CHECK ( typ_osoby IN ( 'S','U','C', 'Z'))
);
```

```
SQL> desc ou
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ROD_CISLO                                 NOT NULL CHAR(11)
 MENO                                      NOT NULL VARCHAR2(20)
 PRIEZVISKO                                NOT NULL VARCHAR2(20)
 LOGIN                                     NOT NULL VARCHAR2(20)
 DAT_ZAPISU                                         DATE
 TYP_OSOBY                                          CHAR(1)
```

Všimnite si:

- že primárny kľúč je automaticky NOT NULL
- a v popise tabuľky sa integritné obmedzenia neukážu.

Príklad vytvorenie tabuľky - pokračovanie

- ```
  SQL> insert into ou (rod_cislo, meno, priezvisko, login, typ_osoby )
    2  values ( '935201/1144', 'Jana', 'Novakova', 'novakova', 'X');
  insert into ou (rod_cislo, meno, priezvisko, login, typ_osoby )
  *
  ERROR at line 1:
  ORA-02290: check constraint (VAJSOVA.SYS_C00363213) violated
  ```

- Ak zadáme prípustnú hodnotu pre stĺpec typ_osoby, príkaz insert už prejde.

  ```
    1  insert into ou (rod_cislo, meno, priezvisko, login, typ_osoby )
    2* values ( '935201/1144', 'Jana', 'Novakova', 'novakova', 'S')
  SQL> /

  SQL> select * from ou;

  ROD_CISLO   MENO                 PRIEZVISKO           LOGIN           DAT_ZAPIS T
  ----------- -------------------- -------------------- --------------- --------- -
  935201/1144 Jana                 Novakova             novakova        04-MAR-15 S
  ```

Všimnime si:
že bola doplnená hodnota v stĺpci dat_zapisu, aj keď sme ju nevypĺňali.
  ```

Default hodnoty

- Ak stĺpec s DEFAULT hodnotou nepridáme do insertu - do tabuľky bude vložená default hodnota.

```
insert into ou (rod_cislo, meno, priezvisko, login, typ_osoby )
  values ( '935201/1144', 'Jana', 'Novakova', 'novakova', 'S');
```

- Ak zadáme do stĺpca konkrétnu hodnotu, nezafunguje DEFAULT hodnota.

```
insert into ou
  values ( '790511/2255', 'Peter','Velky', 'velky', to_date('01.01.2015', 'DD.MM.YYYY') , 'U');
```

- Ak exaktne zadáme do stĺpca NULL hodnotu, nezafunguje DEFAULT hodnota.

```
insert into ou
  values ( '941216/5566','Martin', 'Biely', 'biely',  NULL , 'S');
```

```
SQL> select login, dat_zapisu from ou;

LOGIN                DAT_ZAPIS
-------------------- ---------
novakova             04-MAR-15
velky                01-JAN-15
biely
```

CREATE TABLE - PRIMARY KEY

- Definícia priamo pri stĺpci.

```
CREATE TABLE tabA
(
  id        integer  NOT NULL   primary key ,
  stlpec    VARCHAR2(15)
);
```

- Definícia pomocou tabuľkového obmedzenia.

```
CREATE TABLE tabA
(
  id        integer  NOT NULL,
  stlpec    VARCHAR2(15) ,
   primary key (id)
);
```

CREATE TABLE - PRIMARY KEY - kompozitný primárny kľúč.

```
CREATE TABLE tabB
(
  id1      integer  NOT NULL  primary key,
  id2      integer  NOT NULL  primary key,
  stlpec   VARCHAR2(15)
);
```

CREATE TABLE - PRIMARY KEY - kompozitný primárny kľúč.

```
CREATE TABLE tabB
(
  id1      integer  NOT NULL  primary key,
  id2      integer  NOT NULL  primary key,
  stlpec   VARCHAR2(15)
);

                              *
 ERROR  at line 4:
ORA-02260: table can have only one primary key
```

Poznámka: Nie je možné aby mala tabuľka 2 primárne kľúče.

CREATE TABLE - PRIMARY KEY - kompozitný primárny kľúč.

```
CREATE TABLE tabB
(
  id1      integer  NOT NULL  primary key,
  id2      integer  NOT NULL  primary key,
  stlpec   VARCHAR2(15)
);

                              *
 ERROR  at line 4:
ORA-02260: table can have only one primary key
```

Poznámka: Nie je možné aby mala tabuľka 2 primárne kľúče.

```
CREATE TABLE tabB
(
  id1      integer  NOT NULL,
  id2      integer  NOT NULL,
  stlpec   VARCHAR2(15) ,
   primary key ( id1, id2 )
);
```

CREATE TABLE - FOREIGN KEY

1. Musí byť vytvorená tabuľka s primárnym kľúčom, na ktorú sa chceme odkazovať.

```
CREATE TABLE os_udaje
( rod_cislo CHAR(11)  primary key,
  ...
);
```

```
CREATE TABLE os_udaje
( rod_cislo CHAR(11) ,
  ...,
  primary key (rod_cislo)
);
```

2. V príkaze na vytvorenie tabuľky s cudzím kľúčom, musí byť najprv definícia stĺpcov cudzieho kľúča, ktoré musia byť rovnakého dátového typu a dĺžky ako primárny kľúč.

3. Môžeme doplniť príkaz na vytvorenie cudzieho kľúča.

```
CREATE TABLE student
(
  os_cislo  number not null primary key,
  rod_cislo CHAR(11) not null references os_udaje,
  ...
);
```

```
CREATE TABLE student
(
  os_cislo   number  NOT NULL primary key,
  rod_cislo CHAR(11) NOT NULL,
  ...,
  foreign key (rod_cislo) references os_udaje
);
```

CREATE TABLE - FOREIGN KEY

4. V prípade kompozitného kľúča je potrebné buď uviesť stĺpce cudzieho kľúča v rovnakom poradí ako bol vytvorený primárny kľúč, alebo uviesť aj stĺpce primárneho kľúča, na ktorý sa odkazuje daný cudzí kľúč.

```
CREATE TABLE student
(
  os_cislo       number    NOT NULL primary key,
  st_odbor       NUMBER(3) NOT NULL,
  st_zameranie   NUMBER(3) NOT NULL,
  ...,
  foreign key (st_odbor, st_zameranie)
    references st_odbory( st_odbor, st_zameranie)
);
```

Úprava tabuľky

Syntax

```
ALTER TABLE nazov_tabulky
  {
     ADD ( { definicia_stlpca | tabulkovy_constraint } )
     |
     MODIFY ( { modifikacia_stlpca  } )
     |
     DROP ( { COLUMN nazov_stlpca | CONSTRAINT nazov_constraintu  } )
   }
```

ALTER TABLE - ADD

- Doplnenie stĺpcov do tabuľky.

```
SQL> ALTER TABLE ou
  2  ADD ( dat_zmeny DATE DEFAULT sysdate,
  3        uziv      VARCHAR2(20) );
```

- Doplnenie obmedzenia.

```
SQL> ALTER TABLE ou
  2   ADD CONSTRAINT unique_login UNIQUE ( login );
```

- Doplnenie kontroly dĺžky loginu.

```
SQL> ALTER TABLE ou
  2   ADD CHECK ( length(trim(login)) > 3 );
```

- Pridanie kontroly cudzieho kľúča.

```
SQL> ALTER TABLE st
  2  ADD  FOREIGN KEY ( rod_cislo ) REFERENCES ou;
```

ALTER TABLE - MODIFY

- Oprava stĺpca typ_osoby z NULL na NOT NULL.

```
SQL> ALTER TABLE ou MODIFY ( typ_osoby  NOT NULL );
```

- Doplnenie default hodnoty pre stĺpec typ_osoby.

```
SQL> ALTER TABLE ou MODIFY ( typ_osoby DEFAULT 'S' );
```

- Oprava stĺpca typ_osoby z NOT NULL na NULL.

```
SQL> ALTER TABLE ou MODIFY ( typ_osoby NULL );
```

- Oprava stĺpca typ_osoby - nastavenie default hodnoty a zároveň NOT NULL.

```
SQL> ALTER TABLE ou MODIFY ( typ_osoby DEFAULT 'U' NOT NULL);
```

- Zmena dátového typu stĺpca priezvisko. (nie všetky zmeny sú povolené)

```
SQL> ALTER TABLE ou MODIFY ( priezvisko varchar2(25));
```

ALTER TABLE - DROP

- Odstránenie stĺpca z tabuľky.

```
SQL> ALTER TABLE ou DROP COLUMN typ_osoby;
```

- Odstránenie constraintu z tabuľky.

```
SQL> ALTER TABLE ou DROP CONSTRAINT SYS_C00364696;
```

ALTER TABLE - DROP

- Vypnutie kontroly constraintu.

```
alter table  ou DISABLE CONSTRAINT  SYS_C00364696;
```

- Opätovné zapnutie kontroly constraintu.

```
alter table  ou ENABLE CONSTRAINT  constraint_name;
```

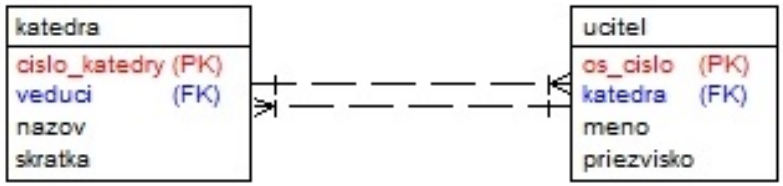
- Odstránenie constraintu z tabuľky.

```
SQL> ALTER TABLE ou DROP CONSTRAINT SYS_C00364696;
```

Názov constraintu je možné zistiť zo systémovej tabuľky `user_constraint`.
`user_constraints(#constraint_name, constraint_type, table_name, r_cosntraint_name)`

Poradie príkazov

- Cudzí kľúč je možné vytvoriť až po vytvorení primárneho kľúča, čo môže byť problém, prípadne nemožné.



- Preto je zvykom vytvoriť najprv všetky tabuľky, a potom pomocou príkazu ALTER table sa doplnia cudzie kľúče.

```
create table ucitel
(  os_cislo   char(5)      NOT NULL primary key,
   meno       varchar2(15) NOT NULL,
   priezvisko varchar2(30) NOT NULL,
   katedra    number  );

create table katedra
( cislo_katedry  number          NOT NULL,
  nazov_katedry  varchar2(30)    NOT NULL,
  veduci         char(5)         NOT NULL,
  skratka        char(4)         NOT NULL,
  primary key (cislo_katedry)
);

alter table ucitel  add ( foreign key (katedra) references katedra );
alter table katedra add ( foreign key (veduci) references ucitel );
```

Upozornenie

```
create table "tab_pom"
(
   id       integer        not null,
   meno     varchar2(10)   not null
);
```

- SQL> select table_name from tabs;

  ```
  TABLE_NAME
  ------------------------------
  tab_pom
  OS_UDAJE
  PREDMET
  PREDMET_BOD
  STUDENT
  ...
  ```

- SQL> desc tab_pom

  ```
  ERROR:
  ORA-04043: object tab_pom does not exist

  SQL> select * from tab_pom ;
  select * from tab_pom
                *
  ERROR at line 1:
  ORA-00942: table or view does not exist
  ```

Upozornenie - pokračovanie

```
SQL> desc "tab_pom"
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(38)
 MENO                                      NOT NULL VARCHAR2(10)

SQL> insert into "tab_pom" values ( 1 , 'Karol');

1 row created.

SQL> select * from "tab_pom" ;

        ID MENO
---------- ----------
         1 Karol
```

Create table z výsledkov selectu

1.
```
Create table Tab as
 select meno, priezvisko,  count(*) as pocet
   from os_udaje
   group by meno, priezvisko;
```

```
SQL> desc tab
 Name                      Null?    Type
 ------------------------- -------- ---------------
 MENO                      NOT NULL VARCHAR2(15)
 PRIEZVISKO                NOT NULL VARCHAR2(15)
 POCET                              NUMBER
```

Create table z výsledkov selectu

1.
```
Create table Tab as
 select meno, priezvisko,  count(*) as pocet
   from os_udaje
   group by meno, priezvisko;
```

```
SQL> desc tab
 Name                      Null?    Type
 ------------------------- -------- ---------------
 MENO                      NOT NULL VARCHAR2(15)
 PRIEZVISKO                NOT NULL VARCHAR2(15)
 POCET                              NUMBER
```

2.
```
Create table Tab as
 select meno, priezvisko,  rocnik+1 as novy_rocnik
   from os_udaje JOIN student using(rod_cislo);
```

```
SQL> desc tab
Name                      Null?    Type
------------------------- -------- ---------------
MENO                      NOT NULL VARCHAR2(15)
PRIEZVISKO                NOT NULL VARCHAR2(15)
NOVY_ROCNIK                        NUMBER
```

Create table z výsledkov selectu

1.
```
Create table Tab as
 select meno, priezvisko,  count(*) as pocet
   from os_udaje
   group by meno, priezvisko;
```

```
SQL> desc tab
 Name                      Null?    Type
 ------------------------- -------- ---------------
 MENO                      NOT NULL VARCHAR2(15)
 PRIEZVISKO                NOT NULL VARCHAR2(15)
 POCET                              NUMBER
```

2.
```
Create table Tab as
 select meno, priezvisko,  rocnik+1 as novy_rocnik
   from os_udaje JOIN student using(rod_cislo);
```

```
SQL> desc tab
 Name                      Null?    Type
 ------------------------- -------- ---------------
 MENO                      NOT NULL VARCHAR2(15)
 PRIEZVISKO                NOT NULL VARCHAR2(15)
 NOVY_ROCNIK                        NUMBER
```

3.
```
Create table Tab as
 select user as pouzivatel, sysdate as datum
   from dual;
```

```
SQL> desc tab
 Name                      Null?    Type
 ------------------------- -------- ---------------
 POUZIVATEL                         VARCHAR2(30)
 DATUM                              DATE
```

Zrušenie tabuľky

Syntax

```
DROP TABLE nazov_tabulky [CASCADE CONSTRAINTS];
```

Tabuľky je možné dropnúť len v správnom poradí (najprv foreign key, až potom tabuľku primárneho kľúča), alebo doplniť klauzulu CASCADE CONSTRAINTS.

Zrušenie tabuľky - príklad

```
SQL> DROP TABLE st;

SQL> DROP TABLE st CASCADE constraints;
```

Index

Vytvorenie indexu

```
CREATE [UNIQUE] INDEX nazov_indexu ON nazov_tabulky ( nazov_stlpca [ {ASC | DESC}] [,...]);
```

Zrušenie indexu

```
DROP INDEX nazov_indexu;
```

- Nie je možné vytvoriť dva indexy s rovnakou množinou stĺpcov indexu. (vrátane poradia)
- S vytvorením primárneho a unikátneho kľúča je automaticky vytvorený aj index.
- Zoznam existujúcich indexov je možné získať z tabuľky

  `user_indexes( #INDEX_NAME, INDEX_TYPE, TABLE_OWNER, TABLE_NAME, UNIQUENESS, ...)`

Vytvorenie indexu

- Príklad vytvorenia indexu s duplicitami.

```
CREATE INDEX  ind_st_rocnik ON student ( rocnik );
```

- Príklad vytvorenia indexu s duplicitami, pričom stĺpec `skrok` bude utriedený naopak.

```
CREATE INDEX  ind_zp1  ON zap_predmety ( skrok DESC, cis_predm );
```

- Vytvorenie unikátneho kľúča.

```
CREATE UNIQUE INDEX ou_login ON os_udaje (login);
```

View

Syntax - vytvorenie

```
CREATE [ OR REPLACE ] VIEW nazov_pohladu
AS
   SELECT-prikaz
  [ WITH
    { READ ONLY
       |
       CHECK OPTION }
  ] ;
```

Syntax - zrušenie vytvorenie

```
DROP VIEW nazov_pohladu;
```

View - príklad

- Vytvorenie pohľadu

```
CREATE OR REPLACE VIEW studenti
AS
   SELECT st.os_cislo, ou.meno, ou.priezvisko, st.st_skupina
     FROM os_udaje ou JOIN student st USING (rod_cislo)
     WHERE stav = 'S'
   WITH READ ONLY;
```

- Z pohľadu môžeme robiť selecty ako keby to bola tabuľka.

```
SQL> SELECT * FROM studenti ;

  OS_CISLO MENO            PRIEZVISKO      ST_SKU
---------- --------------- --------------- ------
    550020 Peter           Kapustny        5ZM031
    501333 Martin          Kluciar         5ZSD11
    501555 Marek           Durica          5ZP012
    501402 Marek           Durica          5ZM023
    501319 Branislav       Balaz           5ZIA21
    500429 Peter           Minarik         5ZSN23
    501469 Stanislav       Steinmuller     5ZP021
    500425 Jaroslav        Cipak           5ZSN23
    501512 Peter           Novak           5ZI031
```

Synonym

Syntax - vytvorenie synonyma

```
CREATE [OR REPLACE] [PUBLIC] SYNONYM nazov_synonym
  FOR nazov_db_objektu;
```

Syntax - zrušenie synonyma

```
DROP [PUBLIC] SYNONYM nazov_synonym
  FOR nazov_db_objektu;
```

Synonymum - príklad

- Vytvorenie synonyma pre tabuľku `valentik.osoba`.

```
SQL> CREATE SYNONYM osoba FOR valentik.osoba;
```

- Ďalej môžeme používať ako pôvodný objekt.

```
SQL> select meno, priezvisko from osoba ;

MENO            PRIEZVISKO
--------------- ---------------
Karol           Najnovsi
Karol           Estenovsi
Karol           Estenajnovsi
Karol           Estenovsi
```

Sequence

Syntax

```
CREATE SEQUENCE nazov_sekvencie
{
      INCREMENT BY krok
   |  START WITH   startovna_hodnota
   |  { MAXVALUE  maximalna_hodnota | NOMAXVALUE }
   |  { MINVALUE  minimalna_hodnota | NOMINVALUE }
   |  { CYCLE | NOCYCLE }
   |  { CACHE  pocet | NOCACHE }
}
```

Syntax

```
DROP SEQUENCE nazov_sekvencie;
```

Sequence - príklady

- Vytvorenie sekvencie.

```
SQL> create sequence sekv_id
  2  start with 1
  3  increment by 1;
```

- Získanie ďalšej hodnoty sekvencie

```
SQL> select sekv_id.nextval from dual;

   NEXTVAL
----------
         1
```

- Získanie posledne pridelenej hodnoty sekvencie v rámci rovnakej session.

```
SQL> select sekv_id.currval from dual;

   CURRVAL
----------
         1
```

Sequence - príklady

1.
```
create sequence sekv_id
       start with 100
       increment by 1
       maxvalue 105
        cycle ;
```

NEXTVAL: 100,101, 102, 103, 104, 105, 1 , 2, 3

Sequence - príklady

1.
```
create sequence sekv_id
      start with 100
      increment by 1
      maxvalue 105
       cycle ;
```

   NEXTVAL: 100,101, 102, 103, 104, 105, 1, 2, 3

2.
```
create sequence sekv_id
    start with  103
    increment by 1
     minvalue 100
    maxvalue 105
    cycle;
```

   NEXTVAL: 103, 104, 105, 100, 101, 102, 103, 104, 105, 100, ...

Sequence - príklady

1.
```
create sequence sekv_id
      start with 100
      increment by 1
      maxvalue 105
       cycle ;
```

NEXTVAL: 100,101, 102, 103, 104, 105, 1, 2, 3

2.
```
create sequence sekv_id
    start with 103
    increment by 1
    minvalue 100
    maxvalue 105
    cycle;
```

NEXTVAL: 103, 104, 105, 100, 101, 102, 103, 104, 105, 100, ...

3.
```
create sequence sekv_id start with 100 increment by 1 maxvalue 105
   nocycle ;
```

NEXTVAL: 100,101, 102, 103, 104, 105,

```
ERROR at line 1:
ORA-08004: sequence SEKV_ID.NEXTVAL exceeds MAXVALUE and cannot be instantiated
```

DAS - Data Access Statement

Syntax - pridelenie práv

```
GRANT { nazov_db_prava | nazov_role }
  TO { uzivatel | rola | PUBLIC };
GRANT  nazov_tab_prava ON nazov_tabulky
  TO { uzivatel | rola | PUBLIC };
```

Syntax - odobratie práv

```
REVOKE { nazov_prava | nazov_role }
  FROM { uzivatel | rola | PUBLIC };
REVOKE  nazov_tab_prava ON nazov_tabulky
  FROM { uzivatel | rola | PUBLIC };
```

Zoznam práv

Databázové práva

```
CONNECT
RESOURCE
CREATE ANY TABLE
CREATE ANY DIRECTORY
CREATE USER
...
```

Objektové práva

```
INSERT
DELETE
UPDATE
SELECT
ALL
```

Pridelenie a odobratie práv

- Pridanie základných práv, aby užívateľ `vajsova` mohol pracovať.

  ```
  GRANT connect, resource TO vajsova;
  ```

- Odobratie práv na vytváranie všetkých typov objektov užívateľovi `vajsova`.

  ```
  REVOKE resource FROM vajsova;
  ```

- Pridelenie práva na select z tabuľky osoba všetkým užívateľovi `matiasko`.

  ```
  GRANT select ON osoba TO matiasko;
  ```

  ```
  REVOKE select ON osoba TO matiasko;
  ```

- Pridelenie a odobratie všetkých práv na tabuľku osoba všetkým užívateľom.

  ```
  GRANT ALL ON osoba TO public;
  ```

  ```
  REVOKE ALL ON osoba FROM public;
  ```