DML - Insert, Delete, Update

3. marca 2015

Insert - syntax

```
INSERT INTO tabulka[(zoznam_stlpcov)]

  VALUES (zoznam_hodnot)
  |
  SELECT-prikaz
```

- Príkaz Insert má dve formy:
  - VALUES - vloženie LEN JEDNÉHO riadku pomocou priameho zadania nových hodnôt.
  - SELECT - vloženie viacerých riadkov predpripravených dát z pomocných tabuliek.
- Je potrebné zadať unikátne hodnoty stĺpcov primárneho kľúča.
- Je potrebné zadať hodnoty všetkých NOT NULL stĺpcov.
- Aby bolo možné znovu pospájať súvisiace dáta z rôznych tabuliek, je potrebné zadať hodnotu cudzieho kľúča.

INSERT - VALUES

```
SQL> desc os_udaje
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ROD_CISLO                                 NOT NULL CHAR(11)
 MENO                                      NOT NULL VARCHAR2(15)
 PRIEZVISKO                                NOT NULL VARCHAR2(15)
 ULICA                                              VARCHAR2(20)
 PSC                                                CHAR(5)
 OBEC                                               VARCHAR2(50)
```

- **Počet a poradie hodnôt je rovnaké ako je vytvorená tabuľka. Neznáme hodnoty nastavte na NULL, ak sa nejedná o NOT NULL stĺpec:**

```
INSERT into os_udaje
VALUES ('810502/5511','Peter', 'Maly',  null, null, null );
```

- **Hodnoty sú v poradí v akom sú vymenované stĺpce:**

```
INSERT INTO os_udaje(meno, priezvisko, rod_cislo)
VALUES ('Peter', 'Maly', '810502/5511');
```

INSERT - SELECT

- **Vloženie hodnôt z pomocnej tabuľky:**

```
INSERT INTO os_udaje( rod_cislo, priezvisko, meno )
SELECT rc, priezvisko, meno
  FROM prijimacky
  WHERE prijaty = 'A';
```

- **Kombinácia hodnôt z tabuľky a konštánt.**

```
INSERT INTO student( os_cislo, rod_cislo, st_odbor, st_zameranie, rocnik, stav, dat_zapisu )
SELECT os_cislo, rod_cislo, st_odbor, st_zameranie,  1, 'S', sysdate
  FROM prijimacky
  WHERE prijaty = 'A';
```

Vložte prvákom (Inf.) všetky povinné predmety prvého ročníka roku 2009

1. Budeme potrebovať získať všetky hodnoty pre NOT NULL stĺpce.

```
SQL> desc zap_predmety
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 OS_CISLO                                  NOT NULL NUMBER(38)
 CIS_PREDM                                 NOT NULL CHAR(4)
 SKROK                                     NOT NULL NUMBER(4)
 PREDNASAJUCI                              NOT NULL CHAR(5)
 ECTS                                      NOT NULL NUMBER(3)
 ZAPOCET                                            DATE
 VYSLEDOK                                           CHAR(1)
 DATUM_SK                                           DATE
```

2.
```
select st.os_cislo
  from student st
  where st.rocnik = 1
    and st.st_odbor = 100
    and st.st_zameranie = 0;
```

3.
```
select pb.cis_predm, pb.skrok, pb.garant, pb.ects
  from st_program stp JOIN predmet_bod pb ON ( stp.cis_predm = pb.cis_predm
           AND stp.skrok = pb.skrok )
  where stp.rocnik = 1
    and stp.skrok = 2009
    and stp.st_odbor = 100
    and stp.st_zameranie = 0
    and stp.typ_povin = 'P';
```

4.
```
select st.os_cislo,cis_predm, skrok, pb.garant, pb.ects
  from student st,
       st_program stp JOIN predmet_bod pb USING ( cis_predm, skrok )
  where st.rocnik = 1
    and st.st_odbor = 100
    and st.rocnik = stp.rocnik
    and stp.skrok = 2009
    and st.st_odbor = stp.st_odbor
    and st.st_zameranie = stp.st_zameranie
    and stp.typ_povin = 'P';
```

5.
```
insert into zap_predmety
      ( os_cislo, cis_predm, skrok, prednasajuci, ects)
select st.os_cislo,cis_predm, skrok, pb.garant, pb.ects
  from student st,
       st_program stp JOIN predmet_bod pb USING ( cis_predm, skrok )
  where st.rocnik = 1
    and st.st_odbor = 100
    and st.rocnik = stp.rocnik
    and stp.skrok = 2009
    and st.st_odbor = stp.st_odbor
    and st.st_zameranie = stp.st_zameranie
    and stp.typ_povin = 'P';
```

Update - syntax

```
UPDATE tabulka
SET stlpec = hodnota
   [, stlpec = hodnota ]
WHERE podmienky;
```

- Príkazom Update môžeme zmeniť viaceré hodnoty LEN jednej tabuľky.
- Ak je podmienka na zmenu riadkov z inej tabuľky, je potrebné použiť vnorený Select.
- Ak máme nové hodnoty v pomocnej tabuľke, je potrebný select príkaz v časti SET. Aby nedošlo k strate dát, je potrebné doplniť aj podmienku, aby boli menené len riadky, ku ktorým máme hodnoty v pomocnej tabuľke.

- **Zmena hodnôt na základe podmienky z rovnakej tabuľky:**

```
UPDATE student
SET stav = 'K',
    ukoncenie = sysdate
where os_cislo = 5204;
```

- **Zmena hodnôt na základe podmienky z druhej tabuľky:**

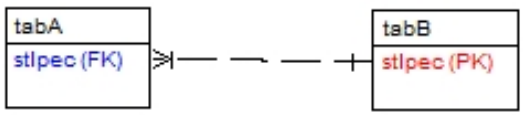
```
UPDATE student
SET stav = 'K',
    ukoncenie = sysdate
where os_cislo IN ( select os_cislo from ABSOLVENTI
                    where skrok = to_char( sysdate, 'YYYY') -1 );
```

- **Zmena hodnôt na hodnoty, ktoré sú v pomocnej tabuľke:**

```
UPDATE student
SET   st_skupina = ( select skupina from st_pom
                     where student.os_cislo = st_pom.os_cislo)
where EXISTS ( select skupina from st_pom
                    where student.os_cislo = st_pom.os_cislo);
```

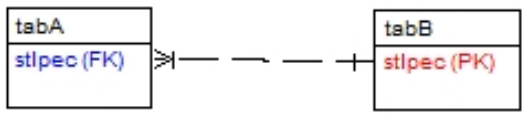
Množina udaná pomocou vnoreného selectu - Jednoduchý primárny kľúč.

[1,1]



alebo

[1,1]




```
UPDATE tabA
SET .....
WHERE tabA.stlpec [NOT] IN
             ( select tabB.stlpec
                 FROM tabB
              )
```

```
UPDATE tabA
SET .....
WHERE [ NOT ] EXISTS
             ( select 'x'
                 FROM tabB
                 WHERE tabA.stlpec = tabB.stlpec
              )
```

Množina udaná pomocou vnoreného selectu.

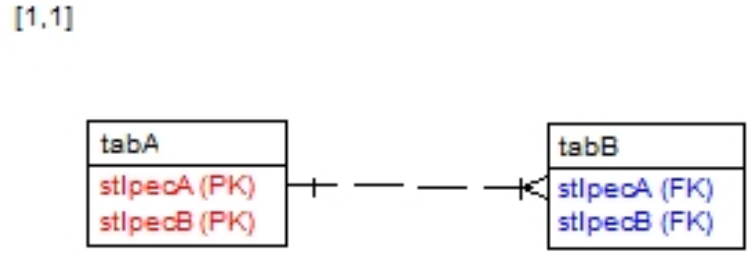
- IN

```
select ou.meno, ou.priezvisko
  from os_udaje ou JOIN student st USING (rod_cislo)
  where st.os_cislo NOT IN
       ( select zp.os_cislo
           from zap_predmety zp
           where zp.cis_predm = 'BI06'
       );
```

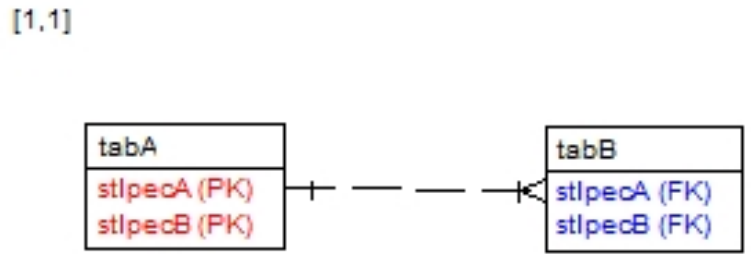
- EXISTS

```
select ou.meno, ou.priezvisko
  from os_udaje ou JOIN student st USING (rod_cislo)
  where  NOT EXISTS
       ( select 'x'
           from zap_predmety zp
           where zp.cis_predm = 'BI06'
             and zp.os_cislo = st.os_cislo
       );
```

Množina udaná pomocou vnoreného selectu - Kompozitný primárny kľúč.



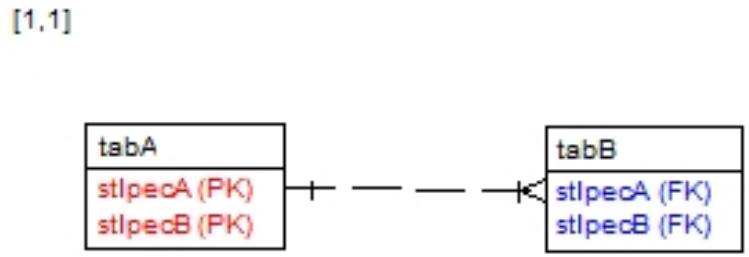
Množina udaná pomocou vnoreného selectu - Kompozitný primárny kľúč.




```
UPDATE tabA
SET .....
WHERE  tabA.stlpecA [NOT] IN
              ( select  tabB.stlpecA
                  FROM tabB
               )
   AND  tabA.stlpecB [NOT] IN
              ( select  tabB.stlpecB
                  FROM tabB
               )
```

```
UPDATE tabA
SET .....
WHERE   [ NOT ] EXISTS
              ( select 'x'
                  FROM tabB
                  WHERE  tabA.stlpecA = tabB.stlpecA
                    AND tabA.stlpecB = tabB.stlpecB
               )
```

Množina udaná pomocou vnoreného selectu - Kompozitný primárny kľúč.



ZLE

```
UPDATE tabA
SET .....
WHERE  tabA.stlpecA [NOT] IN
            ( select  tabB.stlpecA
                FROM tabB
             )
   AND  tabA.stlpecB [NOT] IN
            ( select  tabB.stlpecB
                FROM tabB
             )
```

OK

```
UPDATE tabA
SET .....
WHERE  [ NOT ] EXISTS
            ( select 'x'
                FROM tabB
                WHERE  tabA.stlpecA = tabB.stlpecA
                    AND tabA.stlpecB = tabB.stlpecB
             )
```

Delete - syntax

```
DELETE FROM tabulka
WHERE podmienky;
```

- Slúži na vymazanie hodnôt LEN z jednej tabuľky.
- Ak je podmienka z inej tabuľky, musíme použiť vnorený select.

- **Vymazanie všetkých riadkov tabuľky.**

```
DELETE FROM zap_predmety;
```

- **Vymazanie riadkov na základe podmienky z rovnakej tabuľky.**

```
DELETE FROM zap_predmety
WHERE cis_predm = 'BI06';
```

- **Vymazanie riadkov na základe podmienky z inej tabuľky.**

```
DELETE FROM zap_predmety
WHERE cis_predm IN ( select cis_predm from predmet
                     where nazov like 'Zaklady data%');
```

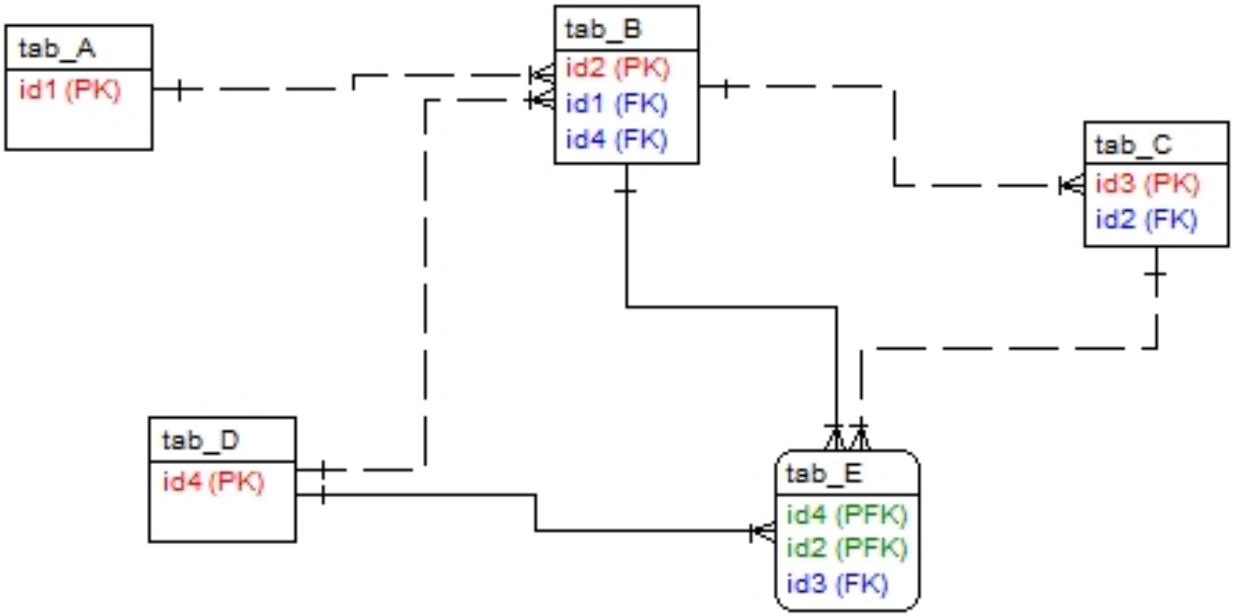
Foreign key - cudzí kľúč

Hodnota Foreign key (FK - cudzí kľúč) je rovná hodnote primary key (PK - primárny kľúč), alebo hodnotu NULL.

- Insert, Load - Najprv musí byť zadaná hodnota primárneho kľúča.
- Delete - Najprv vymazávame hodnoty cudzieho kľúča.

Poradie operácií

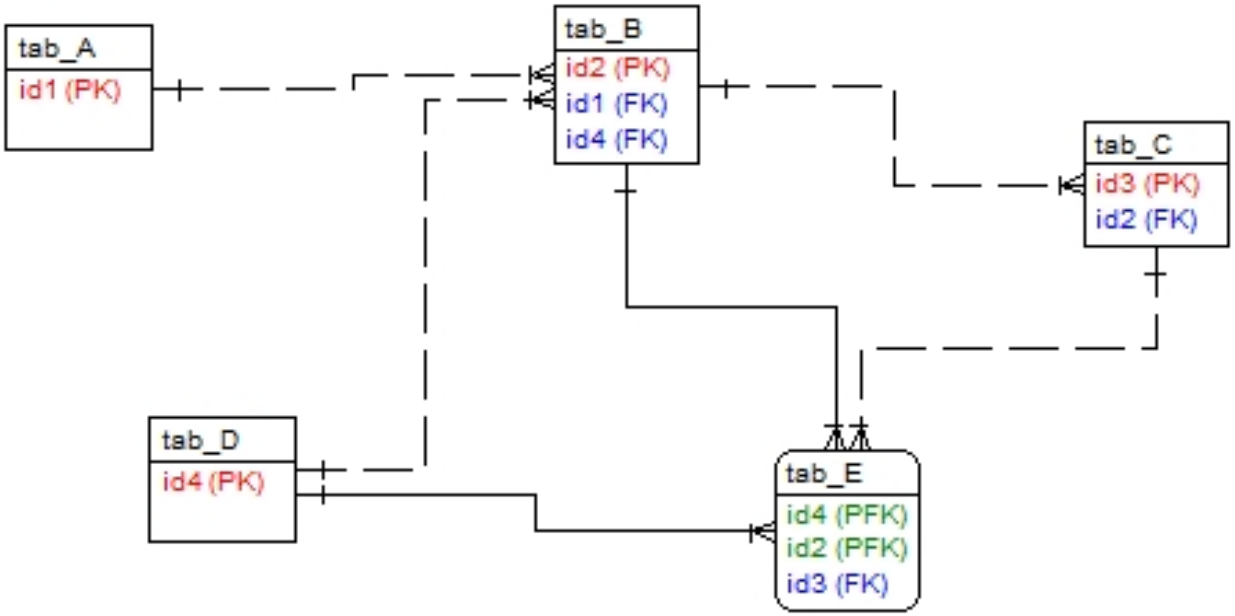
[1,1]



Poradie	**Insert**	**Delete**
tab_A,tab_B,tab_C,tab_D, tab_E		
tab_A,tab_D,tab_B,tab_C, tab_E		
tab_E,tab_C,tab_B,tab_D, tab_A		
tab_D,tab_A,tab_B,tab_C, tab_E		

Poradie operácií

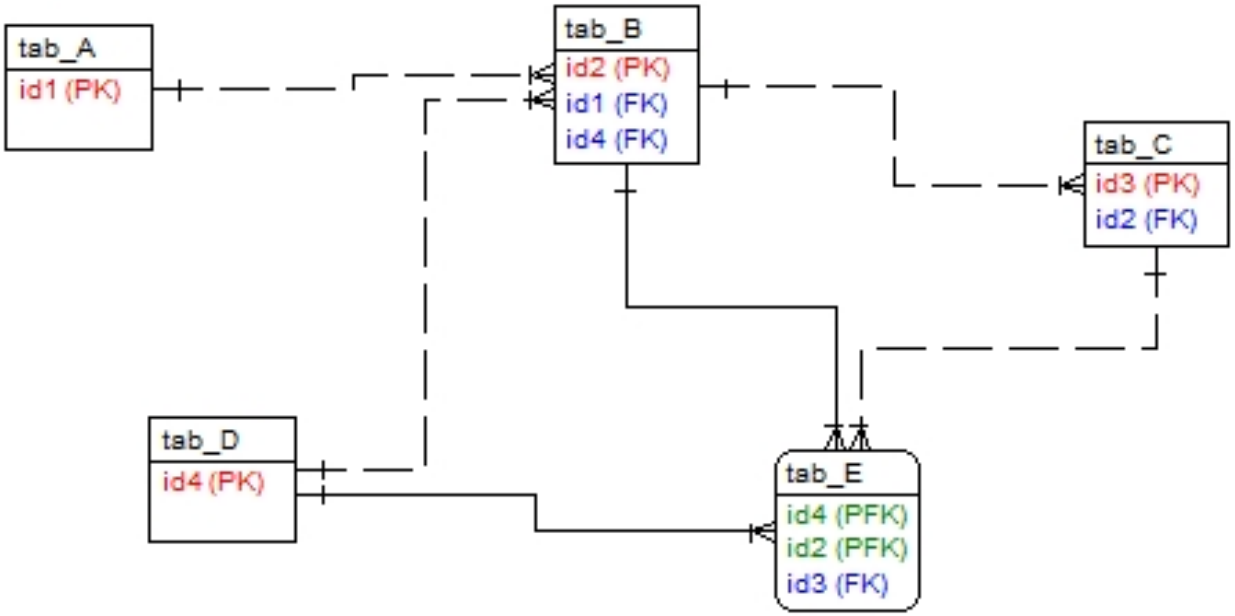
[1,1]



Poradie	**Insert**	**Delete**
tab_A,tab_B,tab_C,tab_D, tab_E	nie	nie
tab_A,tab_D,tab_B,tab_C, tab_E		
tab_E,tab_C,tab_B,tab_D, tab_A		
tab_D,tab_A,tab_B,tab_C, tab_E		

Poradie operácií

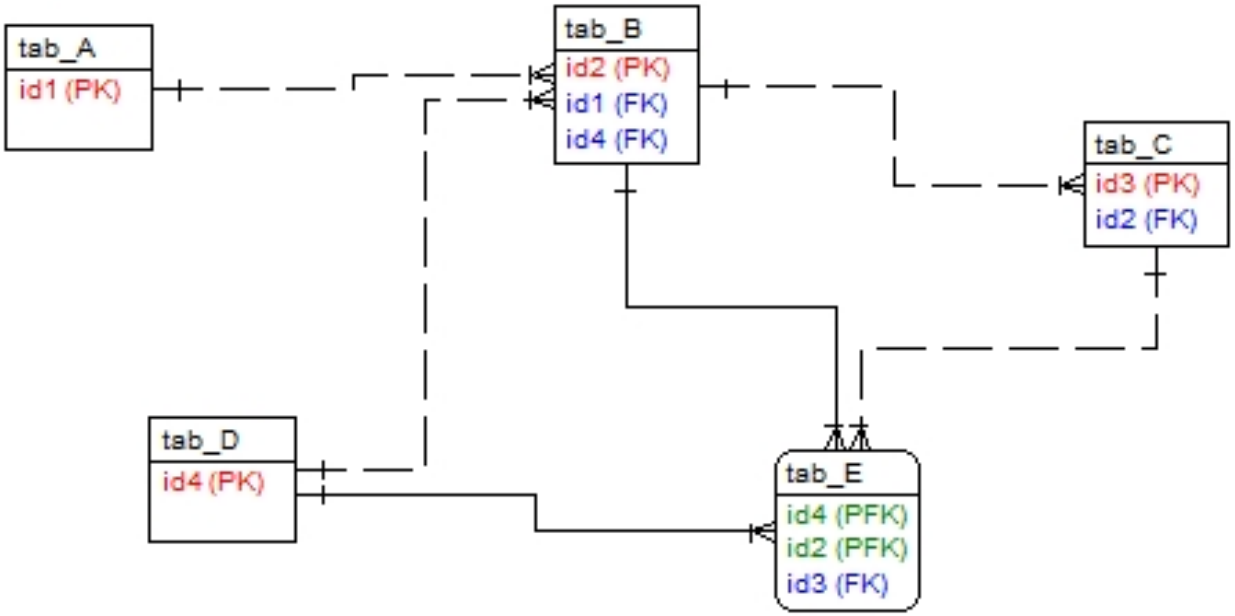
[1,1]



Poradie	**Insert**	**Delete**
tab_A,tab_B,tab_C,tab_D, tab_E	nie	nie
tab_A,tab_D,tab_B,tab_C, tab_E	ano	nie
tab_E,tab_C,tab_B,tab_D, tab_A		
tab_D,tab_A,tab_B,tab_C, tab_E		

Poradie operácií

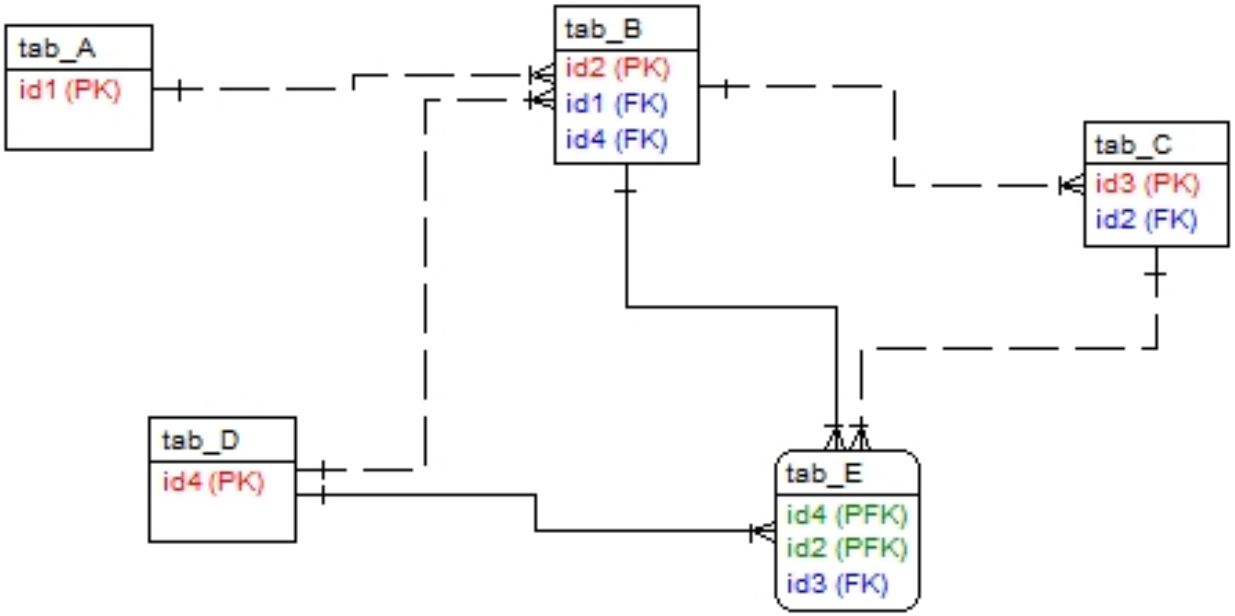
[1,1]



Poradie	**Insert**	**Delete**
tab_A,tab_B,tab_C,tab_D, tab_E	nie	nie
tab_A,tab_D,tab_B,tab_C, tab_E	ano	nie
tab_E,tab_C,tab_B,tab_D, tab_A	nie	ano
tab_D,tab_A,tab_B,tab_C, tab_E		

Poradie operácií

[1,1]



Poradie	**Insert**	**Delete**
tab_A,tab_B,tab_C,tab_D, tab_E	nie	nie
tab_A,tab_D,tab_B,tab_C, tab_E	ano	nie
tab_E,tab_C,tab_B,tab_D, tab_A	nie	ano
tab_D,tab_A,tab_B,tab_C, tab_E	ano	nie

Príkazy ukončenia transakcie

- COMMIT - potvrdenie transakcie - t.j. zmeny budú natrvalo uložené.
- ROLLBACK - neúspešné ukončenie transakcie - t.j. zmeny budú vrátené.
- Všetky DDL-príkazy sú potvrdením transakcie .

```
SQL> create table pom ( cislo integer );

Table created.

SQL> insert into pom values ( 1 );
SQL> insert into pom values ( 2 );
```

```
SQL> create table pom ( cislo integer );

Table created.

SQL> insert into pom values ( 1 );
SQL> insert into pom values ( 2 );

SQL> commit;

Commit complete.
```

```
SQL> create table pom ( cislo integer );

Table created.

SQL> insert into pom values ( 1 );
SQL> insert into pom values ( 2 );

SQL> commit;

Commit complete.
```

```
SQL> insert into pom values ( 3 );

SQL> select * from pom;

     CISLO
----------
         1
         2
         3
```

```
SQL> create table pom ( cislo integer );

Table created.

SQL> insert into pom values ( 1 );
SQL> insert into pom values ( 2 );

SQL> commit;

Commit complete.
```

```
SQL> insert into pom values ( 3 );

SQL> select * from pom;

     CISLO
----------
         1
         2
         3

SQL> rollback;

Rollback complete.
```

```
SQL> create table pom ( cislo integer );

Table created.

SQL> insert into pom values ( 1 );
SQL> insert into pom values ( 2 );

SQL> commit;

Commit complete.
```

```
SQL> insert into pom values ( 3 );

SQL> select * from pom;

     CISLO
----------
         1
         2
         3

SQL> rollback;

Rollback complete.
```

```
SQL> select * from pom;

     CISLO
----------
         1
         2
```

```
SQL> create table pom1 ( cislo integer );

SQL> insert into pom1 values ( 11 );
SQL> insert into pom1 values ( 12 );
```

```
SQL> create table pom1 ( cislo integer );

SQL> insert into pom1 values ( 11 );
SQL> insert into pom1 values ( 12 );

SQL> create table pom2 ( text varchar2(50));
```

```
SQL> create table pom1 ( cislo integer );

SQL> insert into pom1 values ( 11 );
SQL> insert into pom1 values ( 12 );

SQL> create table pom2 ( text varchar2(50));

SQL> rollback;

Rollback complete.
```

```
SQL> create table pom1 ( cislo integer );

SQL> insert into pom1 values ( 11 );
SQL> insert into pom1 values ( 12 );

SQL> create table pom2 ( text varchar2(50));

SQL> rollback;

Rollback complete.
```

```
SQL> select * from pom1;

     CISLO
----------
        11
        12
```