v. 5717a

Jak si na PC vypěstovat

SQL

server a jak ho používat
(snadno a rychle)

2017

by: Ing. Jan Steringa

Instalace

postup je použitelný pro MySQL v 5.7.17 (c) Oracle Corporation

Pro absolutní kontrolu nad instalačním procesem nepoužívám distribuci s instalátorem, ale ZIP archiv.

Po stažení archivu (mysql-5.7.17-win32.zip) rozbalit do adresáře
c:\mysql

Vytvořit adresář, kde budou vytvořeny databázové soubory
c:\mysql\data

Vzorový konfigurační soubor c:\mysql\my-default.ini
zkopírovat a přejmenovat na
c:\mysql\my.ini

Konfigurace

V c:\mysql\bin\my.ini editovat řádky 18 až 20 pro nastavení adresáře s instalací serveru, adresáře pro uložení databází a IP port, na kterém bude server komunikovat.

```
basedir = c:/mysql
datadir = c:/mysql/data
port = 3306
```

(databázový server představuje aplikace mysqld.exe)

Na příkazovém řádku (konzoli Windows) spustit MYSQL server
C:\mysql\bin*mysqld --defaults-file=c:\mysql\my.ini --initialize-insecure --console*
čímž se vytvoří databázový systém a založí účet root (bez hesla).
Průběh je vypisován na konzoli. mysqld se ukončí.

Spuštění serveru

Server lze po prvotní konfiguraci spustit
C:\mysql\bin*mysqld --defaults-file=c:\mysql\my.ini --console*
mysqld se neukončí.

Na jednom stroji může běžet v jednom okamžiku několik instancí serverů MySQL, které lze spustit s různými konfiguračními soubory (parametr ***--defaults-file***). Servery musí běžet na různých IP portech a s různými databázemi - viz parametr datadir v konfiguračním souboru.

Každá instance serveru může obsahovat několik databází a každá databáze může obsahovat několik tabulek, obsahující data.

Ukončení serveru

Ukončení databázového serveru:
C:\mysql\bin\mysqladmin -h 127.0.0.1 -P 3306 -u root -p shutdown

parametry na příkazové řádce:
-h host 127.0.0.1
-P port
-u user
-p password (bude následovat výzva)
-p123 přímo zadané heslo 123

Provoz serveru jako systémové služby

(vyžaduje spustit příkazový řádek jako správce)

instalace služby, manuálně spouštěné, pojmenované MySQL3306
C:\MySQL\bin\mysqld --install-manual MySQL3306
--defaults-file=c:\mysql\my.ini

deinstalce služby
C:\MySQL\bin\mysqld --remove MySQL3306

spuštění služby
net start mysql3306

zastavení služby
net stop mysql3306

Konzole MySQL

Terminálový (konzolový) klient MySQL je aplikace mysql.exe
umístěná v adresáři c:\mysql\bin

Příkazy lze psát na více řádků dokud zadání není ukončeno znakem ;
Sekvence **\c** na další řádce zruší vstup rozepsaného příkazu
Sekvence **\G** (bez ;) způsobí formulářový výstup (select * from user \G)

SQL příkazy lze uložit do dávkového souboru (commands.txt) a spouštět příkazem:
c:\mysql\bin\mysql -u root -p123 -t < commands.txt > c:\mysql\out.htm
nebo
c:\mysql\bin\mysql -u root -p123 -H < commands.txt > c:\mysql\out.txt
Výsledek je uložen do soboru out.txt popř. out.htm
Parametr **-t** zajistí tabulkové formátování výstupu.
Parametr **-H** zajistí formátování HTML.

Pokud se terminálová session zakousne a je potřeba sestřelení procesu lze zjistit aktuální sessions
C:\mysql\bin>mysqladmin -u root -p123 processlist
C:\mysql\bin>mysqladmin -u root -p123 stat

nebo komplexně jedním příkazem
C:\mysql\bin>mysqladmin -u root -p123 proc stat

```
mysqladmin: [Warning] Using a password on the command line interface can be insecure.
+----+------+-----------------+--------+---------+------+----------+------------------+
| Id | User | Host            | db     | Command | Time | State    | Info             |
+----+------+-----------------+--------+---------+------+----------+------------------+
| 5  | root | localhost:64984 | asterx | Sleep   | 21   |          |                  |
| 7  | root | localhost:64988 |        | Query   | 0    | starting | show processlist |
+----+------+-----------------+--------+---------+------+----------+------------------+
Uptime: 93  Threads: 2  Questions: 17  Slow queries: 0  Opens: 106  Flush tables: 1  Open tables: 99
Queries per second avg: 0.182
```

ukončení procesu prostřednictvím jeho identifikátoru ID
C:\mysql\bin>mysqladmin -u root kill 5

Toto lze aplikovat i přímo z konzole MySQL příkazy:
show processlist;
kill 14;

```
mysql> show processlist;
+----+------+-----------------+--------+---------+------+----------+------------------+
| Id | User | Host            | db     | Command | Time | State    | Info             |
+----+------+-----------------+--------+---------+------+----------+------------------+
|  9 | root | localhost:65004 | asterx | Sleep   |   87 |          | NULL             |
| 11 | root | localhost:65009 | NULL   | Query   |    0 | starting | show processlist |
+----+------+-----------------+--------+---------+------+----------+------------------+
2 rows in set (0.00 sec)

mysql> kill 9;
Query OK, 0 rows affected (0.00 sec)
```

Změna hesla na "123" (nejen) účtu root, který nemá heslo:
mysqladmin -u root password 123

Pokud již účet heslo má (např. 123456),
lze jej zadat bezprostředně za parametr -p (bez mezery)
mysqladmin -u root -p123456 password 123

Vynucení dialogu ke změně hesla při nezadání hesel v příkazu:
mysqladmin -u root -p password

Aktualizace práv, znovunačtení tabulek:
mysqladmin -u root -p123 flush-privileges

Připojení k databázi dokud účet root nemá heslo:
mysql -h 127.0.0.1 -P 3306 -u root -p123
parametry na příkazové řádce:
-h host 127.0.0.1
-P port
-u user
-p heslo bude zadáno po výzvě
-p123 přímo zadané heslo 123, bez mezery mezi -p a heslem

založení nového uživatele - **jan** s heslem **hello**
create user jan identified by 'hello';
- heslo je hashováno funkcí password()

smazání uživatele
drop user jan;

pro okamžitou akceptaci nových práv musí být proveden příkaz
flush privileges;

přidělení práv pro uživatele **jan**
grant all on asterx.* to jan;
přidělí se všechna práva - **all**
pro všechny objekty databáze asterx **(asterx.*)**

přidělení práv jen pro jednu konkrétní tabulku
grant all on asterx.osoby to jan;

odebrání práv pro **mazání** datových vět pro uživatele **jan**
revoke delete on asterx.osoby from jan;

Databáze v konzoly MySQL

vypiš databáze v systému:
show databases;

vytvoření databáze "asterx" (nová a prázdná):
create database asterx;

přepni se do databáze ve které se nyní bude pracovat:
use asterx;
a zvolená databáze lze ověřit:
select database();

Zvolit databázi lze i v přihlašovacím příkazu:
C:\mysql\bin\mysql -u root -p123 asterx

vypiš tabulky ve zvolená (aktuální) databázi:
show tables;

likvidace databáze:
drop database asterx;

likvidace tabulky (nejen jejího obsahu):
drop table osoby;

vytvoření nové tabulky

```
create table osoby (
  pk int auto_increment primary key,
  jmeno char(25) not null,
  prijmeni char(40) not null,
  bydliste int
  ) engine=innodb DEFAULT charset=cp1250;
```

tabulka "osoby" bude obsahovat sloupce (pk, jmeno, prijmeni, bydliste)

"pk" a "bydliste" bude číselný typ
"jmeno" a "prijmeni" bude řetězec znaků dlouhý 25 a 40 znaků

auto_increment - pro vložené řádky bez vyplněné této hodnoty bude hodnota automaticky vyplněna - automatické číslování, vzestupné

primary key - hodnoty musí být v rámci tabulky unikátní, neopakovatelné

engine=innodb DEFAULT charset=cp1250 - informace pro db o typu tabulky a použité znakové sadě pro písmena národních abeced

strukturu vytvořené tabulky lze ověřit:
describe osoby;
a vypsat generující příkaz:
show create table osoby \G

vlož záznam (datovou větu) do tabulky

insert into osoby (jmeno,prijmeni,bydliste) values
("jan","steringa",1);

do tabulky osoby vlož do zmíněných položek data, položka "pk" byla při vytváření tabulky nastavena jako **auto_increment** a tak se bude vyplňovat sama

změna datové věty
update osoby set jmeno='jacob' where prijmeni='steringa';

takto lze změnit i heslo uživatele sytému s použitím hashovací funkce, jelikož heslo je v systémové tabulce v nečitelném tvaru:
use mysql;
update user set authentication_string=password('123') where
user='root';
flush privileges;

úpravy vytvořené tabulky

alter table osoby add rc int;
přidání položky

alter table osoby rename os;
přejmenování položky

alter table osoby auto_increment=1;
nastavení autoinkrementace

vymazání datové věty (záznamu) z tabulky

delete from osoby;
vymaže celý obsah tabulky, ne však tabulku jako takovou !

delete from osoby where pk=1;
mazání datových vět s podmínkou - **where**
-smaže záznam, kde datová položka model obsahuje hodnotu "1"

delete from osoby where ((jmeno='jan') and (pk > 5));
složená podmínka - výmaz se provede za předpokladu platnosti obou podmínek, závorky slouží pouze k zpřehlednění - nejsou povinné, ale nanejvýš vhodné .... je to takhle prostě více *cool* !

SELECT data mining - dolování dat z databáze

select 1+1;
spočti 1+1 a vypiš výsledek

```
mysql> select 1+1;
+-----+
| 1+1 |
+-----+
|   2 |
+-----+
1 row in set (0.00 sec)
```

select version(), current_date, current_time;

```
+-----------+--------------+--------------+
| version() | current_date | current_time |
+-----------+--------------+--------------+
| 5.7.17    | 2017-02-12   | 18:36:27     |
+-----------+--------------+--------------+
1 row in set (0.00 sec)
```

select password("Hello world");
select md5("Hello world");
zahesluj a zobraz požadovaný řetězec "Hello world"

```
mysql> select password("Hello world");
+-------------------------------------------+
| password("Hello world")                   |
+-------------------------------------------+
| *10682B6D54CE72511B299AE5A0C1EAAADEA71349 |
+-------------------------------------------+
1 row in set (0.00 sec)

mysql> select md5("Hello world");
+----------------------------------+
| md5("Hello world")               |
+----------------------------------+
| 3e25960a79dbc69b674cd4ec67a72c62 |
+----------------------------------+
1 row in set (0.00 sec)
```

některé příkazy v následujících příkladech jsou funkční pouze na následujících vzorových datech:

tabulka matrix

pk	spz	model	prodejce	kupujici	cena_n	cena_p	datum_n	datum_p
1	cbj 44-45	1	1	NULL	10000	NULL	2007-02-14	NULL
2	cbp 74-15	2	2	NULL	15000	NULL	2007-02-14	NULL
3	cbr 47-45	1	3	NULL	20000	NULL	2007-03-18	NULL
4	cbh 54-35	4	3	NULL	32000	NULL	2007-08-18	NULL
5	cbk 45-25	7	3	NULL	13000	NULL	2007-05-05	NULL
6	cbe 49-75	9	3	NULL	5000	NULL	2007-04-07	NULL
7	cbj 22-11	4	3	NULL	8000	NULL	2007-04-04	NULL
8	cbm 41-27	3	4	NULL	50000	NULL	2007-07-22	NULL
9	cbj 57-25	4	1	NULL	68000	NULL	2007-04-14	NULL

tabulka modely

pk	*vyrobce*	*model*
1	*skoda*	*105 s*
2	*skoda*	*120 s*
3	*skoda*	*125 l*
4	*skoda*	*136 l*
5	*skoda*	*110 r*

tabulka zakaznik

pk	*jmeno*	*prijmeni*	*bydliste*
1	*jan*	*novacek*	*a.michny 7 Budejovice*
2	*petr*	*blaha*	*a.michny 7 Budejovice*
3	*karel*	*malik*	*a.michny 7 Budejovice*
4	*ivan*	*kollar*	*a.michny 7 Budejovice*
5	*david*	*matulka*	*a.michny 7 Budejovice*

Vytvoření vzorových dat

SQL soubor s příkazy **reko.sql** je spuštěn jako dávka příkazem DOSu **c:\mysql\bin\mysql -h 127.0.0.1 -u root -p123 -t -vvv < reko.sql > reko.out.txt**

Dosový příkaz je efektivní uložit do dávkového souboru BAT a spouštět jím dávku SQL příkazů. Výstup SQL dávky je přesměrován z obrazovky do souboru **reko.out.txt**

```
# database creation

create database if not exists reko;
drop database reko;
create database reko;
use reko;

# table constructor

create table matrix (
  pk int auto_increment primary key,
  spz char(10),
  model int not null,
  prodejce int not null,
  kupujici int,
  cena_n int,
  cena_p int,
  datum_n date,
  datum_p date
  ) ENGINE=MyISAM DEFAULT CHARSET=cp1250;

create table zakaznik (
  pk int primary key,
  jmeno char(25) not null,
  prijmeni char(40) not null,
  bydliste char(40)
  ) ENGINE=MyISAM DEFAULT CHARSET=cp1250;

create table modely (
  pk int primary key,
  vyrobce char(25) not null,
  model char(40) not null
  ) ENGINE=MyISAM DEFAULT CHARSET=cp1250;


# test-data import

insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbj 44-45",1,1,10000,"2007-02-14");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbp 74-15",2,2,15000,"2007-02-14");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbr 47-45",1,3,20000,"2007-03-18");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbh 54-35",4,3,32000,"2007-08-18");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbk 45-25",7,3,13000,"2007-5-5");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbe 49-75",9,3,5000,"2007-4-7");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbj 22-11",4,3,8000,"2007-4-4");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbm 41-27",3,4,50000,"2007-7-22");
insert into matrix (spz,model,prodejce,cena_n,datum_n) values ("cbj 57-25",4,1,68000,"2007-4-14");

insert into zakaznik (pk,jmeno,prijmeni,bydliste) values (1,"jan","novacek","a.michny 7 Budejovice");
insert into zakaznik (pk,jmeno,prijmeni,bydliste) values (2,"petr","blaha","a.michny 7 Budejovice");
insert into zakaznik (pk,jmeno,prijmeni,bydliste) values (3,"karel","malik","a.michny 7 Budejovice");
insert into zakaznik (pk,jmeno,prijmeni,bydliste) values (4,"ivan","kollar","a.michny 7 Budejovice");
insert into zakaznik (pk,jmeno,prijmeni,bydliste) values (5,"david","matulka","a.michny 7 Budejovice");

insert into modely (pk,vyrobce, model) values (1,"skoda","105 s");
insert into modely (pk,vyrobce, model) values (2,"skoda","120 s");
insert into modely (pk,vyrobce, model) values (3,"skoda","125 l");
insert into modely (pk,vyrobce, model) values (4,"skoda","136 l");
insert into modely (pk,vyrobce, model) values (5,"skoda","110 r");
```

select * from matrix;
vypiš celou tabulku "matrix"

```
mysql> select * from matrix;
+----+-----------+-------+----------+----------+--------+--------+------------+---------+
| pk | spz       | model | prodejce | kupujici | cena_n | cena_p | datum_n    | datum_p |
+----+-----------+-------+----------+----------+--------+--------+------------+---------+
|  1 | cbj 44-45 |     1 |        1 |     NULL |  10000 |   NULL | 2007-02-14 | NULL    |
|  2 | cbp 74-15 |     2 |        2 |     NULL |  15000 |   NULL | 2007-02-14 | NULL    |
|  3 | cbr 47-45 |     1 |        3 |     NULL |  20000 |   NULL | 2007-03-18 | NULL    |
|  4 | cbh 54-35 |     4 |        3 |     NULL |  32000 |   NULL | 2007-08-18 | NULL    |
|  5 | cbk 45-25 |     7 |        3 |     NULL |  13000 |   NULL | 2007-05-05 | NULL    |
|  6 | cbe 49-75 |     9 |        3 |     NULL |   5000 |   NULL | 2007-04-07 | NULL    |
|  7 | cbj 22-11 |     4 |        3 |     NULL |   8000 |   NULL | 2007-04-04 | NULL    |
|  8 | cbm 41-27 |     3 |        4 |     NULL |  50000 |   NULL | 2007-07-22 | NULL    |
|  9 | cbj 57-25 |     4 |        1 |     NULL |  68000 |   NULL | 2007-04-14 | NULL    |
+----+-----------+-------+----------+----------+--------+--------+------------+---------+
9 rows in set (0.00 sec)
```

select count(*) from matrix;
vypiš počet datových vět (záznamů) v tabulce

```
mysql> select count(*) from matrix;
+----------+
| count(*) |
+----------+
|        9 |
+----------+
1 row in set (0.00 sec)
```

select pk, prodejce from matrix;
vypiš jen požadované sloupce (**pk, prodejce)** z tabulky "matrix"

```
mysql> select pk,prodejce from matrix ;
+----+----------+
| pk | prodejce |
+----+----------+
|  1 |        1 |
|  2 |        2 |
|  3 |        3 |
|  4 |        3 |
|  5 |        3 |
|  6 |        3 |
|  7 |        3 |
|  8 |        4 |
|  9 |        1 |
+----+----------+
9 rows in set (0.00 sec)
```

select * from modely where model like "%20_s";

výběr dat, které splňují podmínku **where**
použití wildcards (zástupné znaky) - v podmínce musí být **like**
- znak "%" nahrazuje řetězec znaků
- znak "_" nahrazuje jediný znak

pokud potřebujeme vyhledávat přímo tyto znaky (" _ %) použijeme před nimi backslash "\" .... **like "\%20"** platí pro %20

```
mysql> select * from modely where model like "%20_s";
+----+---------+-------+
| pk | vyrobce | model |
+----+---------+-------+
|  2 | skoda   | 120 s |
+----+---------+-------+
1 row in set (0.00 sec)
```

operátoty v podmínce WHERE - pro porovnání číselných položek
= , > , < , >= , <= , <> , is null

select distinct model from matrix;

výstupem jsou pouze rozdílné hodnoty ... za každou hodnotu zobrazí pouze jednoho zástupce

```
mysql> select distinct model from matrix;
+-------+
| model |
+-------+
|     1 |
|     2 |
|     4 |
|     7 |
|     9 |
|     3 |
+-------+
6 rows in set (0.00 sec)
```

```
select matrix.pk, matrix.spz, zakaznik.prijmeni "prodejce"
       from matrix, zakaznik
       where matrix.prodejce=zakaznik.pk;
```

příklad joinu - vypiš požadované sloupečky z požadovaných tabulek

použit tzv.tečkovaný tvar - tabulka.záznam

nadpis pro sloupeček **zakaznik.prijmeni** bude nadepsán "prodejce"

sloupec **matrix.prodejce** obsahuje čisla, které mají být v tomto selectu nahrazeny záznamy z tabulky "zakaznik" ... **zakaznik.prijmeni**

- je vytvořeno logické propojení **matrix.prodejce=zakaznik.pk**

- propojení je platné jen pro okamžik výpisu

```
mysql> select matrix.pk, matrix.spz, zakaznik.prijmeni "prodejce"
       from matrix, zakaznik
       where matrix.prodejce=zakaznik.pk;
+----+-----------+----------+
| pk | spz       | prodejce |
+----+-----------+----------+
|  1 | cbj 44-45 | novacek  |
|  9 | cbj 57-25 | novacek  |
|  2 | cbp 74-15 | blaha    |
|  3 | cbr 47-45 | malik    |
|  4 | cbh 54-35 | malik    |
|  5 | cbk 45-25 | malik    |
|  6 | cbe 49-75 | malik    |
|  7 | cbj 22-11 | malik    |
|  8 | cbm 41-27 | kollar   |
+----+-----------+----------+
9 rows in set (0.00 sec)
```

```
select matrix.pk, matrix.spz, modely.model "model"
       from matrix, modely
       where matrix.model=modely.pk;
```

ještě jednou podobný select

```
mysql> select matrix.pk, matrix.spz, modely.model "model"
       from matrix, modely
       where matrix.model=modely.pk;
+----+-----------+-------+
| pk | spz       | model |
+----+-----------+-------+
|  1 | cbj 44-45 | 105 s |
|  3 | cbr 47-45 | 105 s |
|  2 | cbp 74-15 | 120 s |
|  8 | cbm 41-27 | 125 l |
|  4 | cbh 54-35 | 136 l |
|  7 | cbj 22-11 | 136 l |
|  9 | cbj 57-25 | 136 l |
+----+-----------+-------+
7 rows in set (0.00 sec)
```

..... kam se ale poděl záznam matrix.pk 5 a 6 ???
- není vypsán, protože **matrix.model** který nemá odpovídající protějšek mezi záznamy **modely.pk** se nepropojí a nevypíše

```
select matrix.pk, matrix.spz, modely.model "model"
  from matrix left join (modely)
    on matrix.model=modely.pk;
```

provedení joinu s přesahem - vypíší se i záznamy, které nemají v druhé tabulce "své protějšky" - tyto záznamy mají v propojovaných datech prázdný datový záznam - hodnota NULL (datové vakuum)

```
mysql> select matrix.pk, matrix.spz, modely.model "model"
       from matrix
       left join (modely)
         on matrix.model=modely.pk;
+----+-----------+-------+
| pk | spz       | model |
+----+-----------+-------+
|  1 | cbj 44-45 | 105 s |
|  2 | cbp 74-15 | 120 s |
|  3 | cbr 47-45 | 105 s |
|  4 | cbh 54-35 | 136 l |
|  5 | cbk 45-25 | NULL  |
|  6 | cbe 49-75 | NULL  |
|  7 | cbj 22-11 | 136 l |
|  8 | cbm 41-27 | 125 l |
|  9 | cbj 57-25 | 136 l |
+----+-----------+-------+
9 rows in set (0.00 sec)
```

```
select matrix.pk, matrix.spz, modely.model "model"
  from matrix left join (modely)
  on matrix.model=modely.pk
where modely.model is null;
```

vypíší se jen záznamy bez svých protějšků - kde při spojení vzniklo vakuum, podmínka is null

výraz "= null" nebude fungovat

```
mysql> select matrix.pk, matrix.spz, modely.model "model"
       from matrix
       left join (modely)
         on matrix.model=modely.pk
       where modely.model is null;
+----+-----------+-------+
| pk | spz       | model |
+----+-----------+-------+
|  5 | cbk 45-25 | NULL  |
|  6 | cbe 49-75 | NULL  |
+----+-----------+-------+
2 rows in set (0.00 sec)
```

```
select matrix.pk, matrix.spz
  from matrix
  where matrix.model in (select modely.pk from modely)
```

hledání záznamů, které nemají své protějšky v jiné tabulce
tentokrát verze s vnořeným selectem

- pokud není záznam v seznamu **in** (vygenerovaným vnořeným selectem) nebude vypsán

je to možná přehlednější zápis než s "joinem s přesahem", ale pro db mnohem pomalejší ... pokud zpracováváme 10000 a více záznamů

```
mysql> select matrix.pk, matrix.spz from matrix where matrix.model in (select modely.pk from modely);
+----+-----------+
| pk | spz       |
+----+-----------+
|  1 | cbj 44-45 |
|  2 | cbp 74-15 |
|  3 | cbr 47-45 |
|  4 | cbh 54-35 |
|  7 | cbj 22-11 |
|  8 | cbm 41-27 |
|  9 | cbj 57-25 |
+----+-----------+
7 rows in set (0.01 sec)
```

```
select matrix.pk, matrix.spz
  from matrix
  where matrix.model not in (select modely.pk from modely);
```

negace předchozí logiky

```
mysql> select matrix.pk, matrix.spz from matrix where matrix.model not in (select modely.pk from modely);
+----+-----------+
| pk | spz       |
+----+-----------+
|  5 | cbk 45-25 |
|  6 | cbe 49-75 |
+----+-----------+
2 rows in set (0.00 sec)
```

```
select pk, spz from matrix where pk in (1,3,5);
```

další příklad na IN - pracuj jen se záznamy uvedenými v seznamu

```
mysql> select pk, spz from matrix where pk in (1,3,5);
+----+-----------+
| pk | spz       |
+----+-----------+
|  1 | cbj 44-45 |
|  3 | cbr 47-45 |
|  5 | cbk 45-25 |
+----+-----------+
3 rows in set (0.00 sec)
```

select pk, spz from matrix where pk between 2 and 5;

vymezení podmínky rozsahem hodnot - datové věty musí mít záznam v datové položce **pk** v rozsahu mezi 2 - 5

```
mysql> select pk, spz from matrix where pk between 2 and 5;
+----+-----------+
| pk | spz       |
+----+-----------+
|  2 | cbp 74-15 |
|  3 | cbr 47-45 |
|  4 | cbh 54-35 |
|  5 | cbk 45-25 |
+----+-----------+
4 rows in set (0.03 sec)
```

select prodejce, count(*) from matrix group by prodejce;
vypiš různé prodejce a počet záznamů, kde se vyskytují

```
mysql> select prodejce, count(*) from matrix group by prodejce;
+----------+----------+
| prodejce | count(*) |
+----------+----------+
|        1 |        2 |
|        2 |        1 |
|        3 |        5 |
|        4 |        1 |
+----------+----------+
4 rows in set (0.00 sec)
```

select prodejce, count(*)
from matrix
group by prodejce
having count(*) >= 2;

vypiš různé prodejce a počet záznamů, kde se vyskytují, ale jen ty prodejce kteří se vyskytují 2 a vícekrát

```
mysql> select prodejce, count(*) from matrix group by prodejce having count(*) >= 2;
+----------+----------+
| prodejce | count(*) |
+----------+----------+
|        1 |        2 |
|        3 |        5 |
+----------+----------+
2 rows in set (0.00 sec)
```

naplnění tabulky daty z externího souboru

soubor ve formátu CSV - comma separated values, data oddělené čárkou

```
******************** file: data.csv ****************************
10;porsche;115 z;
11;porsche;12 b;
12;mercedes;a 2200;
13;mercedes;a 2100;
14;mercedes;c 6255;
15;mercedes;d 6565;

************************************************************
```

```
load data local
  infile "e:/data.csv"
  into table modely fields
  terminated by ";";
```

```
mysql> load data local
       infile "e:/data.csv"
       into table modely
       fields terminated by ";";

Query OK, 6 rows affected, 6 warnings (0.00 sec)
Records: 6  Deleted: 0  Skipped: 0  Warnings: 6
```

parametr **"local"** - data jsou na disku klienta, jinak se soubor hledá na serveru

data v souboru musí na každém řádku končit příslušným znakem, např. středníkem ";", při exportu dat z excelu tento znak většinou chybí a musí se ručně do textového souboru doplnit

poslední záznam v datovém souboru by měl být (prázdná řádka), při více prázdných řádkách jsou vkládány do db nulové záznamy

naplnění tabulky daty ze selectu

novou tabulku **mx** lze vytvořit jako stín stávající tabulky **matrix**
create table mx like matrix;
(data se nepřenesou)

naplnění daty pomocí selectu nad jinou tabulkou
insert into mx select * from matrix where pk is not null;

```
mysql> insert into mx select * from matrix where pk is not null;
Query OK, 9 rows affected (0.09 sec)
Records: 9  Duplicates: 0  Warnings: 0
```

transakce

transakce lze použít u tabulek, které mají obsluhu innodb enginem

show table status;
vypíše vlastnosti tabulek z aktuální databáte

alter table matrix engine = innodb;
alter table matrix engine = myisam;
pro tabulku "matrix" přepne engine (mechanizmus obsluhy) db stroje

jako výchozí je nastaven při startu nové session autocommit, ten je potřeba pro transakční zpracování vypnout a nastartovat transakční mechanizmus

set autocommit=0;
start transaction;

nyní jsou innodb tabulky schopny transakčního zpracování
(ne všechny operace podléhají transakčnímu přístupu, např. drop table nelze vrátit)

výsledek příkazu insert nebo delete lze vrátit příkazem
rollback;
nebo potvrdit příkazem
commit;

pro jemnější ošetření lze vkládat v průběhu zpracování transakce zarážky, ke kterým se lze vrátit
(pokud se vrátíme k nějaké zarážce, mladší zarážky se zruší)

vlož zarážku, vrať se k zarážce, smaž zarážku
savepoint stav_a;
rollback to savepoint stav_a;
release savepoint stav_a;

triggery

hlídací pes v databázi, číhající na definovaný povel

```
delimiter //
create trigger tr_01 before insert on modely for each row
  begin
    if new.pk < 4 then
      set new.pk = new.pk + 10;
    elseif new.pk > 4 then
      set new.pk = new.pk + 20;
    end if;
  end; //
delimiter ;
```

delimiter // - povel bloku, středníky nejsou brány (až do odvolání) za ukončení příkazu

// blok ukončen

delimiter; - zrušení sekvence // jako delimiteru

tr_01 - název triggeru

before / after - doba zapůsobení triggeru

insert /update / delete - hlídaná operace, povel pro hlídacího psa

new.položka / old.položka - položka po / před hlídanou operací

drop trigger reko.tr_01;
smazáni triggeru tr_01 v databázi reko

show triggers;
zobrazení aktivních triggerů v databázi

```
mysql> show triggers;
+---------+--------+--------+-----------------------------------------------------------------------------
-----------------------------------------+--------+---------+----------+----------------+
| Trigger | Event  | Table  | Statement
                                    | Timing | Created | sql_mode | Definer        |
+---------+--------+--------+-----------------------------------------------------------------------------
-----------------------------------------+--------+---------+----------+----------------+
| tr_01   | INSERT | modely | begin
   if new.pk < 4 then
     set new.pk = new.pk + 10;
   elseif new.pk > 4 then
     set new.pk = new.pk + 20;
    end if;
  end | BEFORE | NULL    |          | root@localhost |
+---------+--------+--------+-----------------------------------------------------------------------------
-----------------------------------------+--------+---------+----------+----------------+
1 row in set (0.02 sec)
```