

# Skrypt

## Wprowadzenie do bazy MySQL

**Zbigniew Gołębiewski**

Wersja 3.10

Grójec 2017

## Spis treści

1.	INSTALACJA i KONFIGURACJA SERWERA MySQL 5.7.....	5
2.	FOLDERY BAZY MySQL W SYSTEMIE WINDOWS .....	10
3.	URUCHOMIENIE I ZATRZYMANIE USŁUGI SERWERA MySQLw SYSTEMIE WINDOWS .....	10
1.	Polecenie wyszukujące nazwę usługi MySQL w systemie Windows: .....	10
2.	Uruchamianie usługi w systemie Windows .....	10
3.	LOGOWANIE DO BAZY DANYCH.....	11
4.	PIERWSZE KROKI W BAZIE MySQL.....	12
A.	Wyświetlenie aktualnej godziny, użytkownika i wersji.....	12
B.	Polecenie show .....	13
C.	Podstawowe operacje w bazie.....	14
i.	Wyświetlanie listy baz na serwerze .....	14
ii.	Zastosowanie wybranej bazy .....	14
iii.	Wyświetlanie listy tabel danej bazy .....	15
iv.	Wyświetlanie opisu wybranej tabeli .....	15
5.	Magia polecenie mysqladmin .....	17
A.	Ustawienie pierwszego hasła dla roota .....	17
B.	Zmiana hasła dla dowolnego konta .....	17
C.	Sprawdzenie czy serwer MySQL działa .....	18
D.	Sprawdzenie wersji serwer MySQL.....	18
E.	Sprawdzenie status serwera MySQL.....	18
F.	Sprawdzenie wartości zmiennych serwera MySQL.....	19
G.	Sprawdzenie procesów serwera MySQL.....	19
I.	Uruchamianie, zatrzymywanie serwera relikacji slave .....	19
6.	Partycjonowanie tabeli .....	20
A.	Co to jest partycjonowanie? .....	20
B.	Testowanie partycji.....	20
C.	Typy partycjonowania.....	21
D.	Tworzenie partycji.....	21
	Składnia polecenia: .....	21
7.	Użytkownicy i uprawnienia .....	23
A.	Zarządzanie kontem MySQL.....	24
I.	Tworzenie użytkownika – create user .....	24
B.	Język DCL (Data Control Language).....	24
I.	Nadawanie uprawnień -- grant .....	24

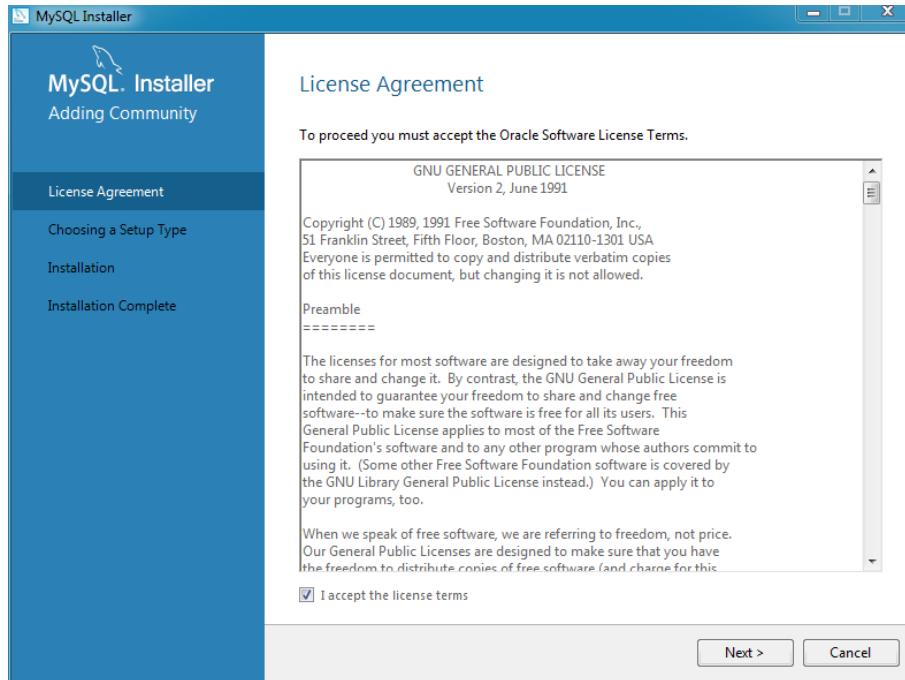
II.	Odbieranie uprawnień -- revoke .....	26
III.	Sprawdzenie uprawnień użytkownika .....	26
7.	Typy danych w bazie MySQL.....	27
8.	Język DDL.....	28
I.	Tworzenie bazy danych.....	28
II.	Usuwanie bazy danych.....	28
III.	Tworzenie tabeli bazy danych.....	28
I.	Typy danych w/g tabeli z punktu 7. ....	29
II.	Atrybuty kolumny: .....	29
III.	OPCJE TABELI.....	29
IV.	Realizacja kluczy obcych w MySQL: .....	30
IV.	Wyświetlenie sposobu tworzenia tabeli: .....	32
V.	Usuwanie tabeli bazy danych.....	33
VI.	Aktualizowanie tabeli bazy danych.....	33
VII.	Czyszczenie tabeli bazy danych.....	37
9.	Testowanie tabel - polecenie mysqlcheck .....	37
10.	Silniki bazy MySQL.....	37
A.	Ustalanie dostępnych silników bazy danych.....	38
B.	Reprezentacja tabeli na dysku .....	39
C.	Silniki MySQL.....	40
D.	Zmiana silnika baz danych.....	40
E.	Sprawdzenie czy dana tabela posiada określony silnik.....	41
11.	Automatyzacja pracy a MySQL, czyli skrypty .....	42
A.	Uruchamianie skryptów .....	43
12.	WSTAWIANIE, AKTUALIZACJA I USUWANIE DANYCH z BAZY MySQL .....	44
A.	Wstawianie danych do tabeli.....	44
I.	Polecenie INSERT .....	44
II.	Polecenie INSERT i SELECT .....	44
III.	Załadowanie danych z pliku przy pomocy polecenia load data local infile.....	44
IV.	Załadowanie danych z pliku z pliku przy pomocy polecenia mysqlimport .....	44
V.	Przykłady dla polecenia INSERT: .....	45
VI.	Przykład dla polecenia mysqlimport .....	46
VII.	Przykład dla polecenia load data local infule .....	47
B.	Aktualizacja danych w bazie .....	47
C.	Usuwanie danych z bazy .....	48
13.	KOPIA ZAPASOWA w MySQL.....	48

A.	Wykonywanie kopii zapasowej .....	48
I.	Możemy również wykonać kopię wszystkich baz: .....	48
II.	Możemy również wykonać kopię kilku baz jednocześnie:.....	48
III.	Możemy również wykonać kopię wybranej tabeli bazy: .....	49
B.	Przywracanie kopii zapasowej .....	49
14.	Dzienniki zdarzeń serwera MySQL.....	49
15.	TESTY WYDAJNOŚCI w MySQL .....	50
A.	MySQL Enterprise Monitor .....	50
B.	MySQL Workbench .....	51
C.	MySQL Utilites.....	54
D.	Explain.....	55
E.	mysqldumpslow.pl .....	58
F.	MySQL Tuner.....	58
G.	Polecenie mytop .....	59
H.	Indeksy .....	60
16.	PERSPEKTYWY .....	60
A.	Tworzenie widoków .....	60
B.	UWAGI:.....	61
C.	Widoki niemodyfikowalne .....	61
D.	SPRAWDZENIE PORAWNOŚCI WIDOKU .....	61
E.	Usuwanie widoków.....	61
17.	INDEKSY.....	62
A.	Rodzaje indeksów .....	62
B.	Widoki wykorzystywane są, aby: .....	62
C.	Gdzie stosować: .....	63
D.	Gdzie nie należy stosować: .....	63
E.	TWORZENIE INDEKSÓW .....	63
F.	USUWANIE INDEKSÓW .....	64
18.	REPLIKACJA.....	64

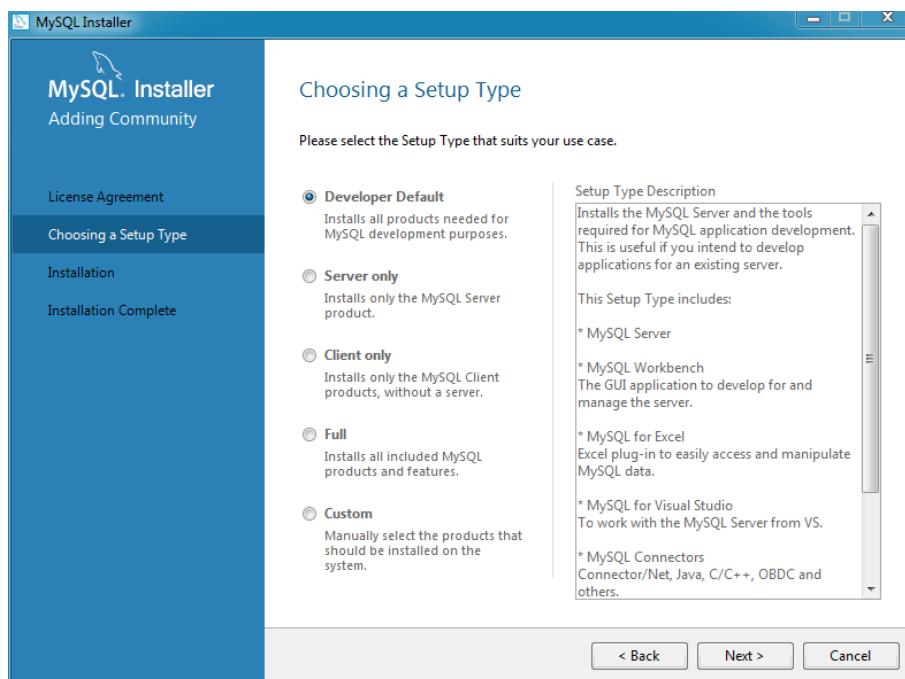
## 1. INSTALACJA i KONFIGURACJA SERWERA MySQL 5.7

### A. Uruchomienie instalatora

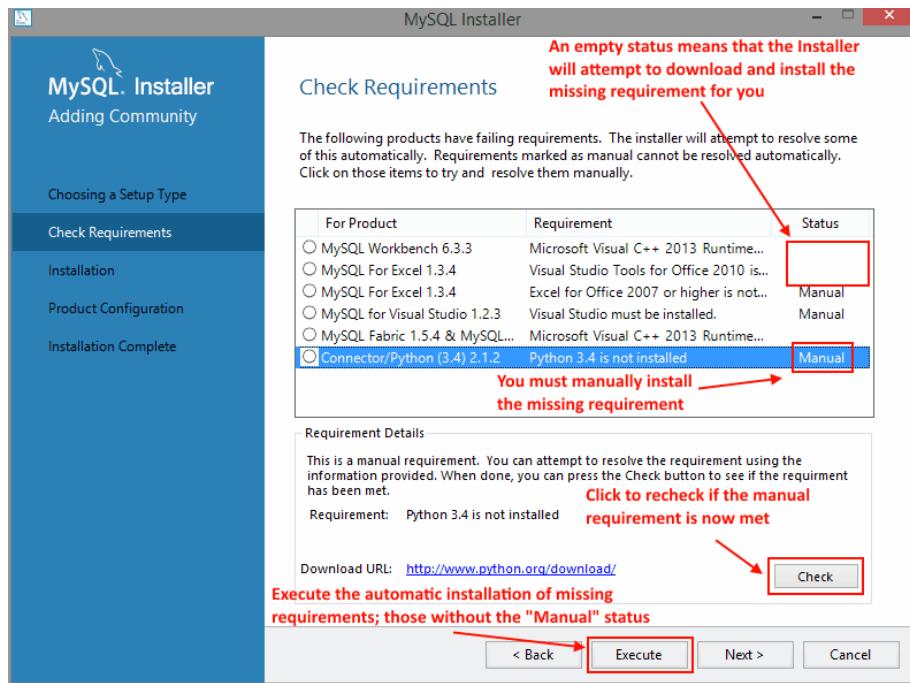
### B. Akceptujemy warunki licencji



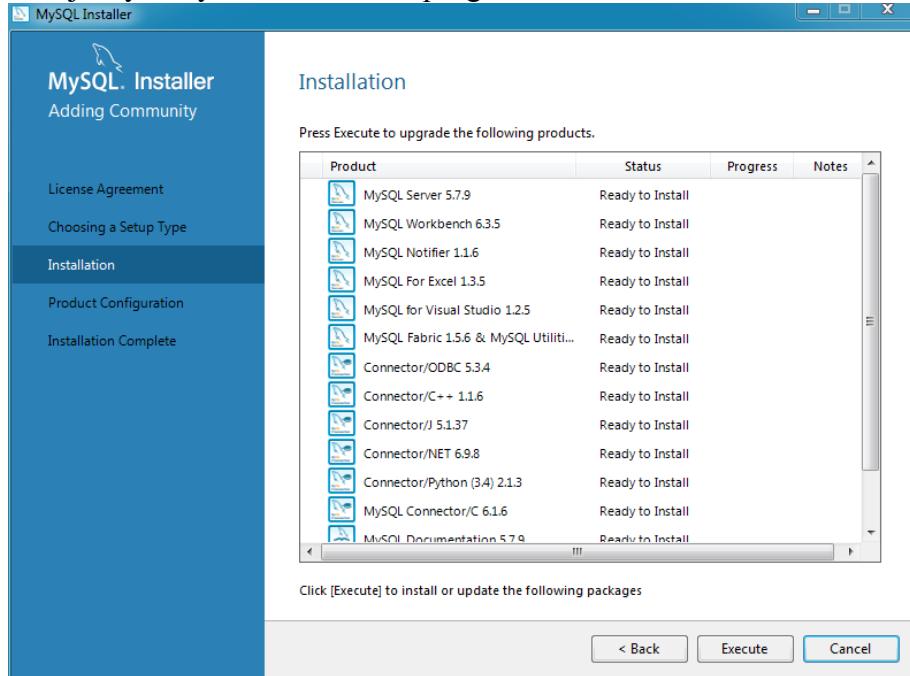
### C. Wybieramy typ instalacji



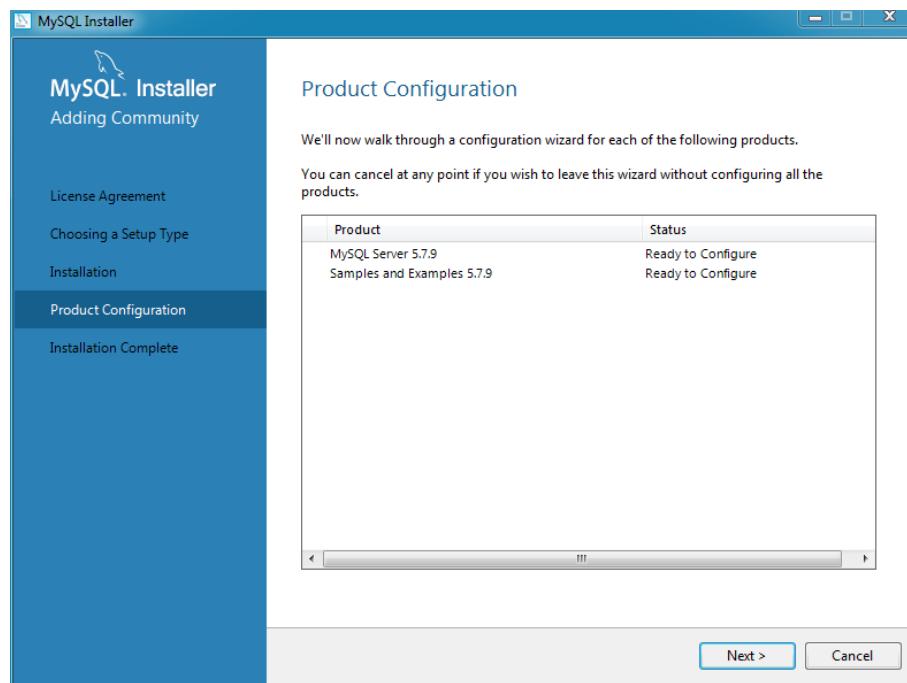
## D. Sprawdzenie wymagań przez instalator



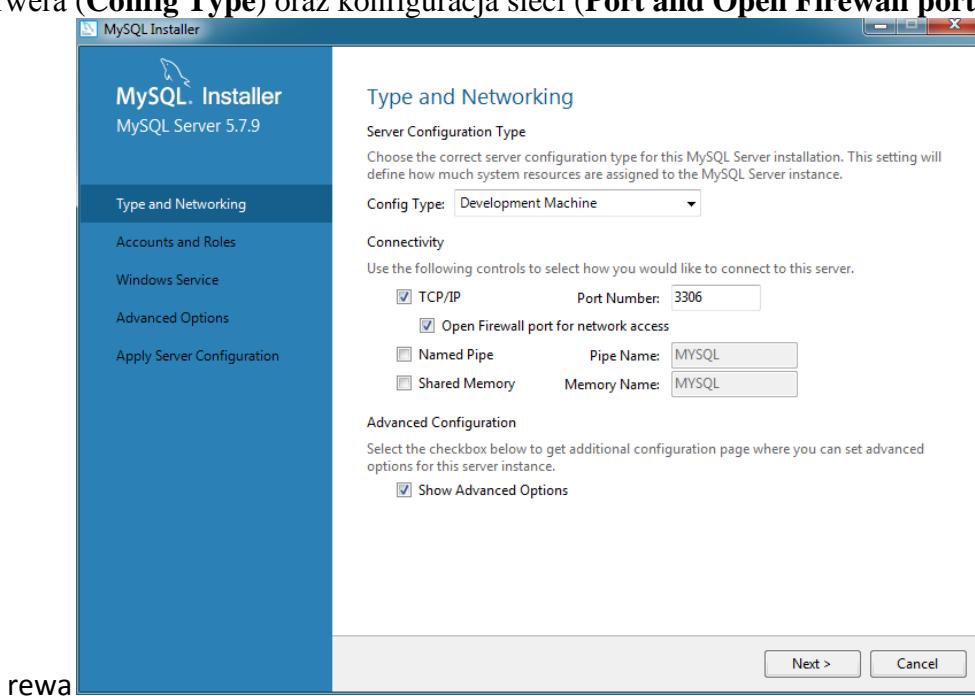
## E. Pobieranie i instalacja wybranych składników oprogramowania



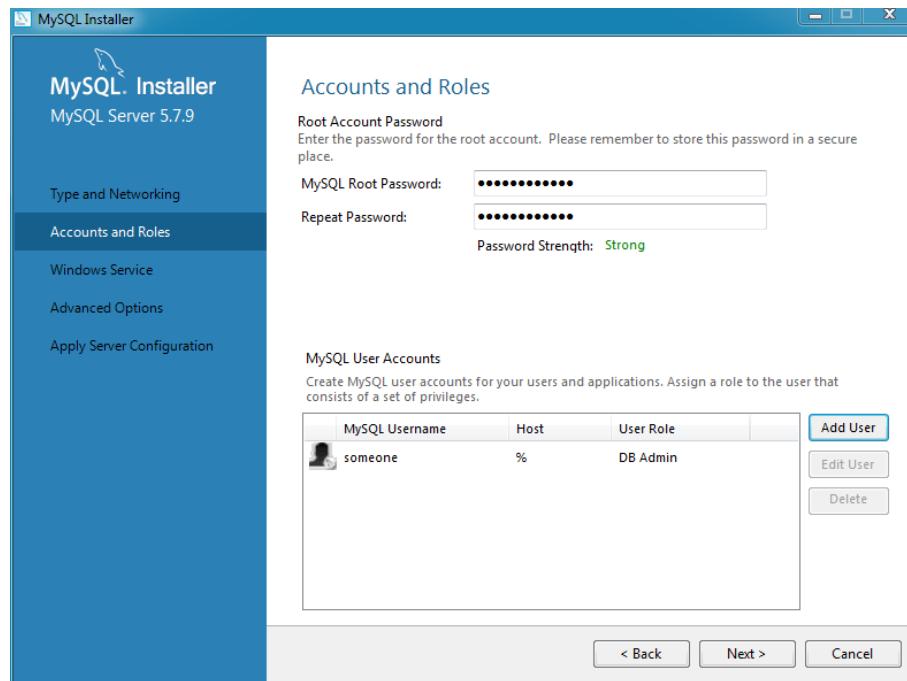
## F. Przygotowanie serwer MySQL do wstępnej konfiguracji



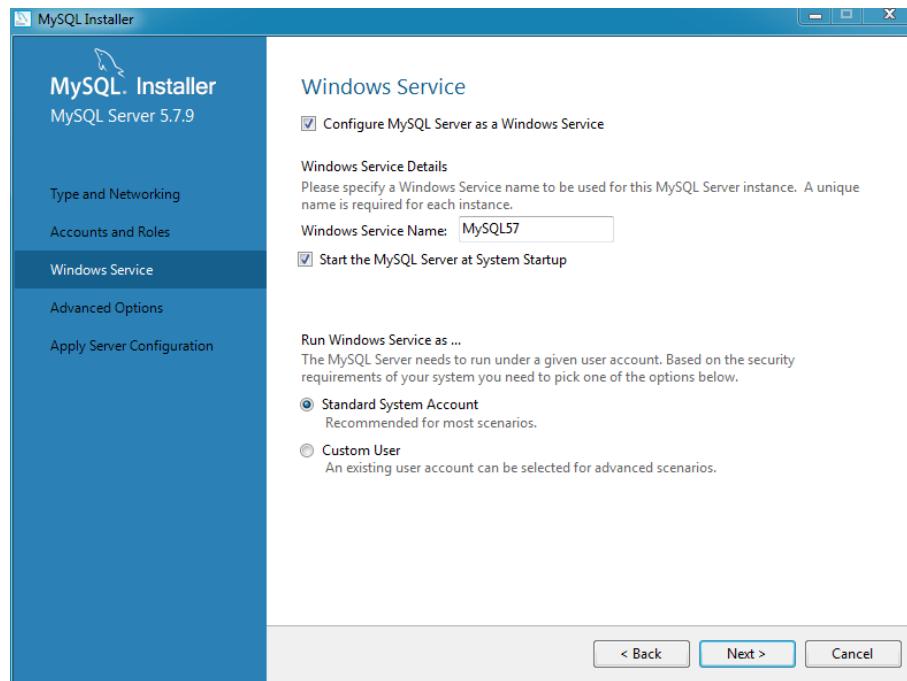
## G. Wybór typu serwera (**Config Type**) oraz konfiguracja sieci (**Port and Open Firewall port**)



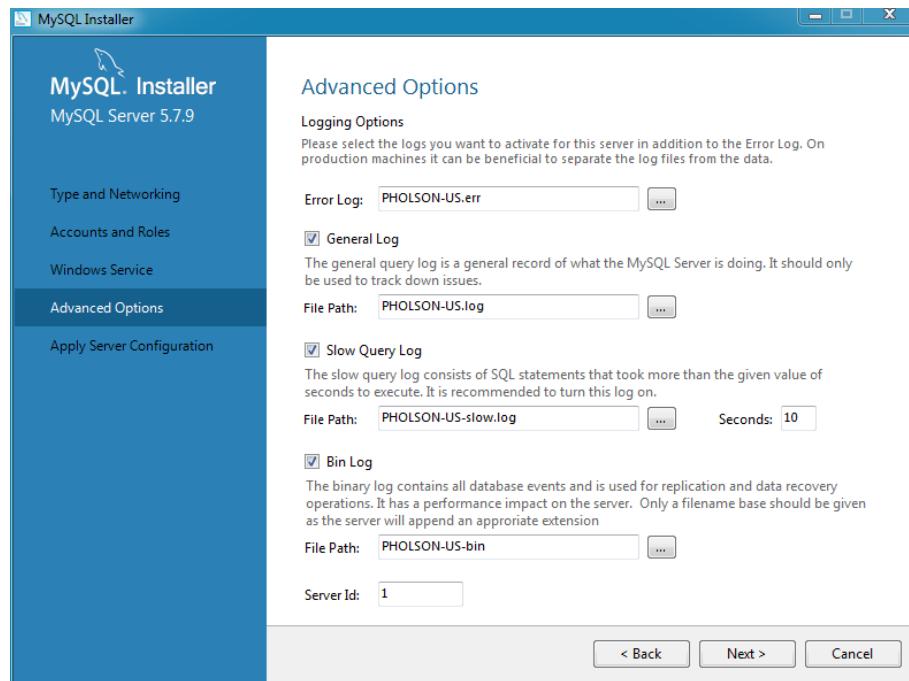
## H. Utworzenie hasła do konta root-a oraz utworzenie dodatkowych kont na serwerze



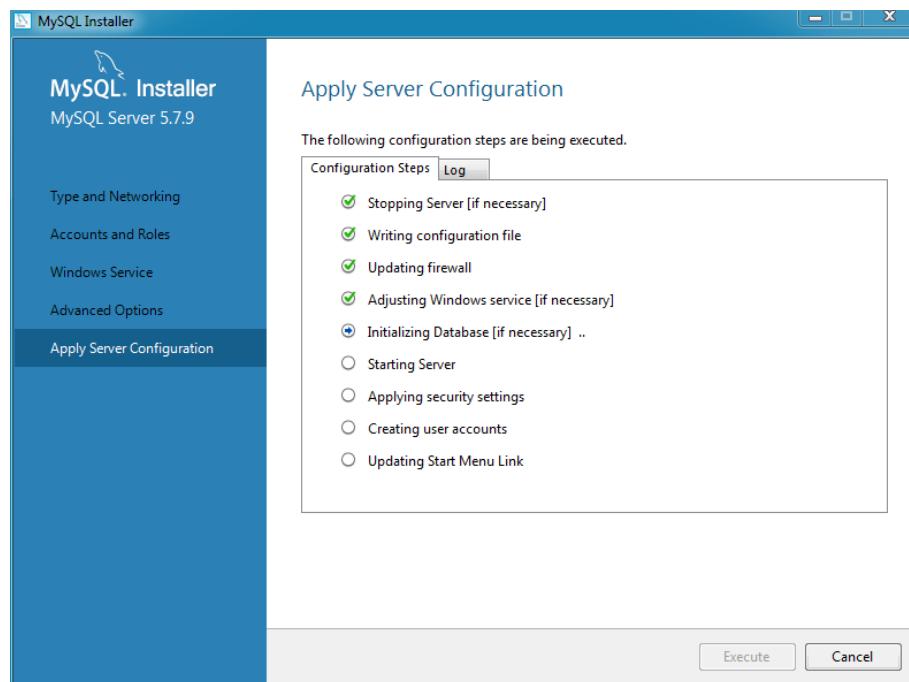
## I. Konfiguracja usługi w systemie Windows (**Windows Service Name i Start the MySQL Server at System Startup**)



## J. Konfiguracja plików logów dla bazy MySQL



## K. Zastosowanie wprowadzonych ustawień i uruchomienie serwera



## **2. FOLDERY BAZY MySQL W SYSTEMIE WINDOWS**

Domyślana ścieżka instalacji **MySQL 5.7** w systemie Windows (32-bitowy), to:

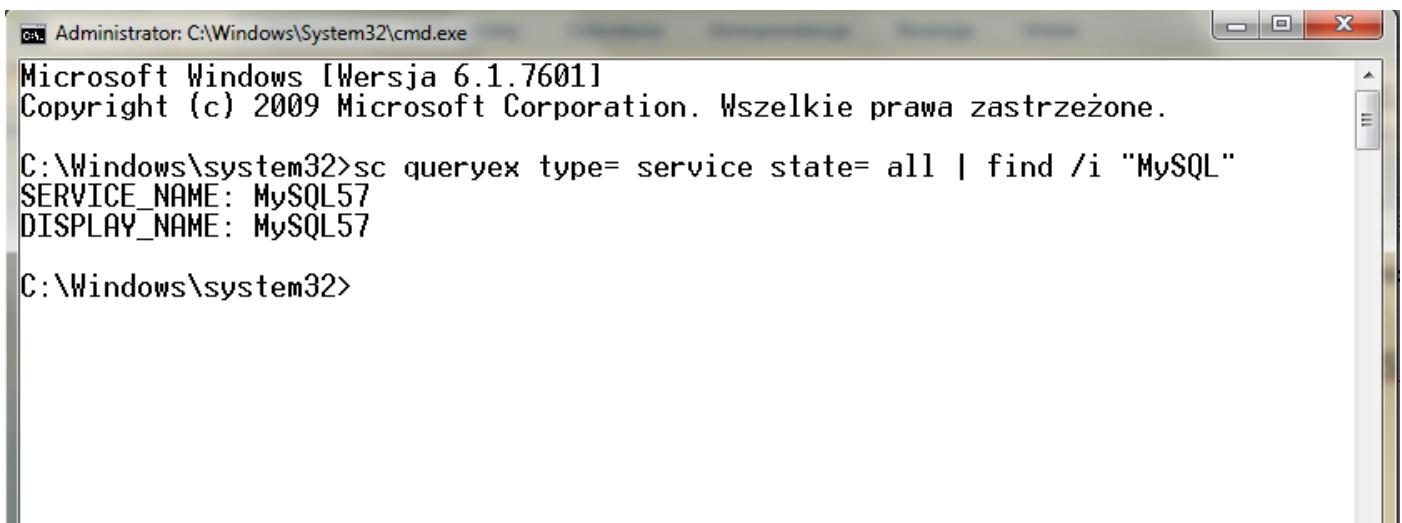
C:\Program Files\MySQL\MySQL Server 5.7.

Lp.	Foldery	Opis
1.	bin	mysqld server oraz inne narzędzia serwer
2.	examples	przykładowa baza i skrypty
3.	include	pliki nagłówkowe języka C++
4.	lib	biblioteki
5.	share	error message w różnych wersjach językowych, ustawienia poszczególnych języków
6.	%PROGRAMDATA%\MySQL\MySQL Server 5.7\	Folder gdzie przechowywane są logi serwera oraz foldery baz systemowych i bazy użytkownika

## **3. URUCHOMIENIE I ZATRZYMANIE USŁUGI SERWERA MySQL w SYSTEMIE WINDOWS**

### **1. Polecenie wyszukujące nazwę usługi MySQL w systemie Windows:**

```
sc queryex type= service state= all | find /i "MySQL"
```



```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Wersja 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Windows\system32>sc queryex type= service state= all | find /i "MySQL"
SERVICE_NAME: MySQL57
DISPLAY_NAME: MySQL57

C:\Windows\system32>
```

Poprawne wykonanie powyższego polecenia w konsoli cmd systemu Windows

### **2. Uruchamianie usługi w systemie Windows**

Do uruchomienia, zatrzymania lub restartu usługi używamy polecenia net.

Składnia podstawowa tego polecenia to:

**net TRYB NazwaUsługi,**

gdzie TRYB to jedna z opcji: start, stop lub restart, NazwaUsługi – usługa, którą chce uruchomić, zatrzymać lub wykonać restart.

Uruchomienie usługi MySQL57 w konsoli cmd\*:

**net start MySQL57**

Zatrzymywanie usługi MySQL57 w konsoli cmd\*:

**net stop MySQL57**

\* Konsola cmd musi być uruchomiona w trybie administratora.

### **3. LOGOWANIE DO BAZY DANYCH**

Składnia polecenia do logowania:

**mysql -h NazwaKomputer -u NazwaUżytkownika -p**

Możemy zastosować też formy alternatywne:

*-h NazwaKomputera : --host=NazwaKomputera lub adres IP*

*-u NazwaUżytkownika : -- user=NazwaUżytkownika*

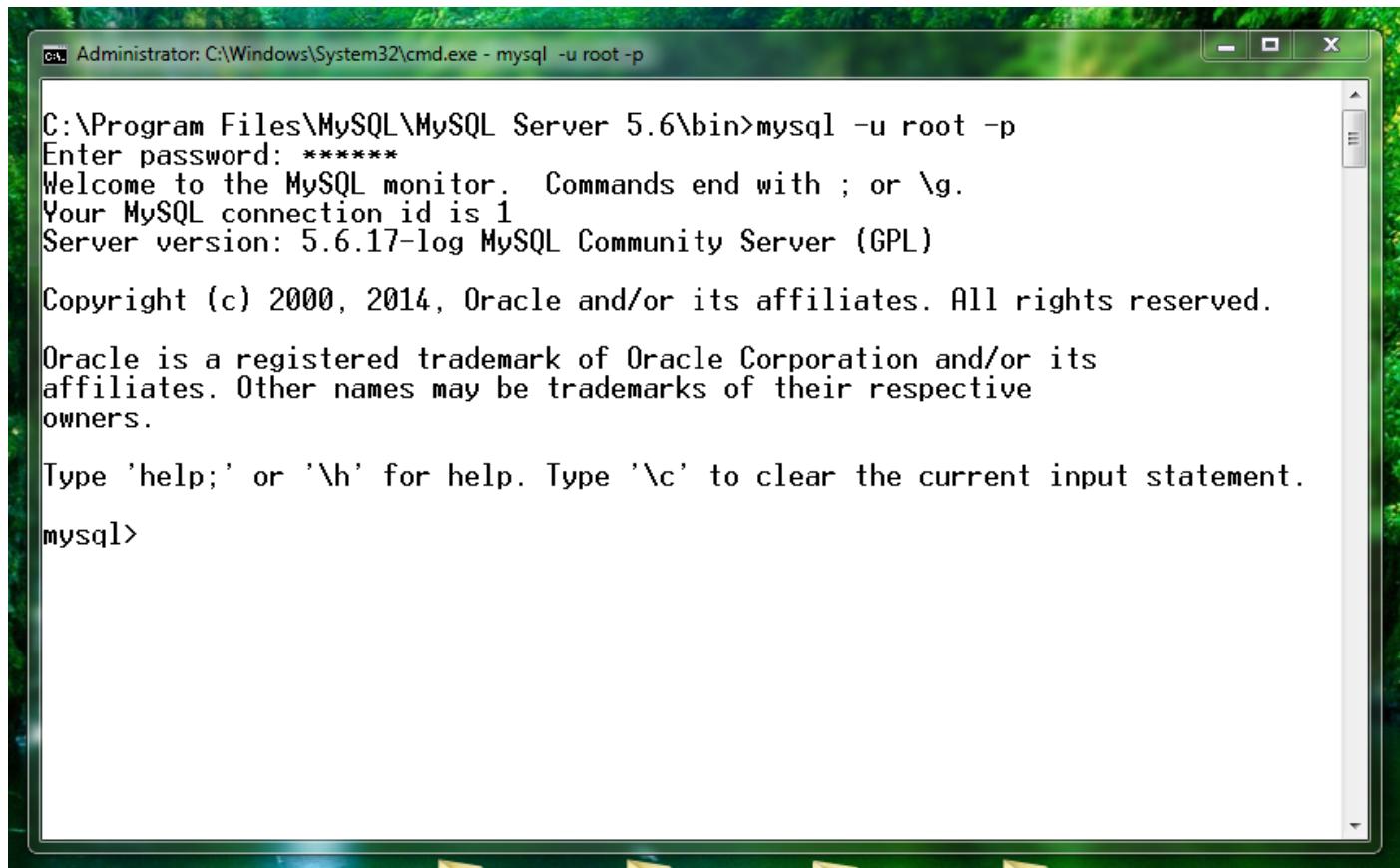
*-p Hasło: --password=Hasło niezalecane, bo można łatwo odczytać hasło*

Jeżeli konto użytkownik nie posiada hasła , to podczas logowania pomijamy parametr p.

Przykład:

**mysql -u root -p**

W tym przypadku użytkownik root zostanie poproszony o podanie hasła. W czasie wpisywania hasła pojawiają się gwiazdki.



C:\Program Files\MySQL\MySQL Server 5.6\bin>mysql -u root -p  
Enter password: \*\*\*\*\*  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 1  
Server version: 5.6.17-log MySQL Community Server (GPL)  
  
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql>

Poprawne zalogowanie do serwera MySQL

Zakończenie pracy z bazą MySQL to polecenie **bye** lub **quit**.

#### 4. PIERWSZE KROKI W BAZIE MySQL

##### A. Wyświetlenie aktualnej godziny, użytkownika i wersji

Wyświetlenie bieżącej daty i godziny:      **select now();**

Wyświetlenie bieżącego użytkownika:      **select user();**

Wyświetlenie wersji MySQL:      **select version();**

Powysze proste zapytania mogą być zastąpione przez zapytania złożone, np.:

- **select now(), user();**
- lub
- **select now(), user(), version();**

```

mysql> select now(), user(), version();
+-----+-----+-----+
| now() | user() | version() |
+-----+-----+-----+
| 2014-12-02 21:07:08 | root@localhost | 5.6.17-log |
+-----+-----+-----+
1 row in set (0.06 sec)

```

lub

- `select now(), user(), version() \G`

```

mysql> select now(), user(), version() \G
***** 1. row *****
    now(): 2014-12-02 21:08:56
    user(): root@localhost
version(): 5.6.17-log
1 row in set (0.00 sec)

```

## B. Polecenie show

Show ma wiele form, które dostarczają informacji o bazach danych, tabele, kolumny, czy informacje o stanie o serwerze. Więcej informacji można znaleźć na stronie :

<http://dev.mysql.com/doc/refman/5.7/en/show.html>

Najczęściej stosowane wersje polecenia show:

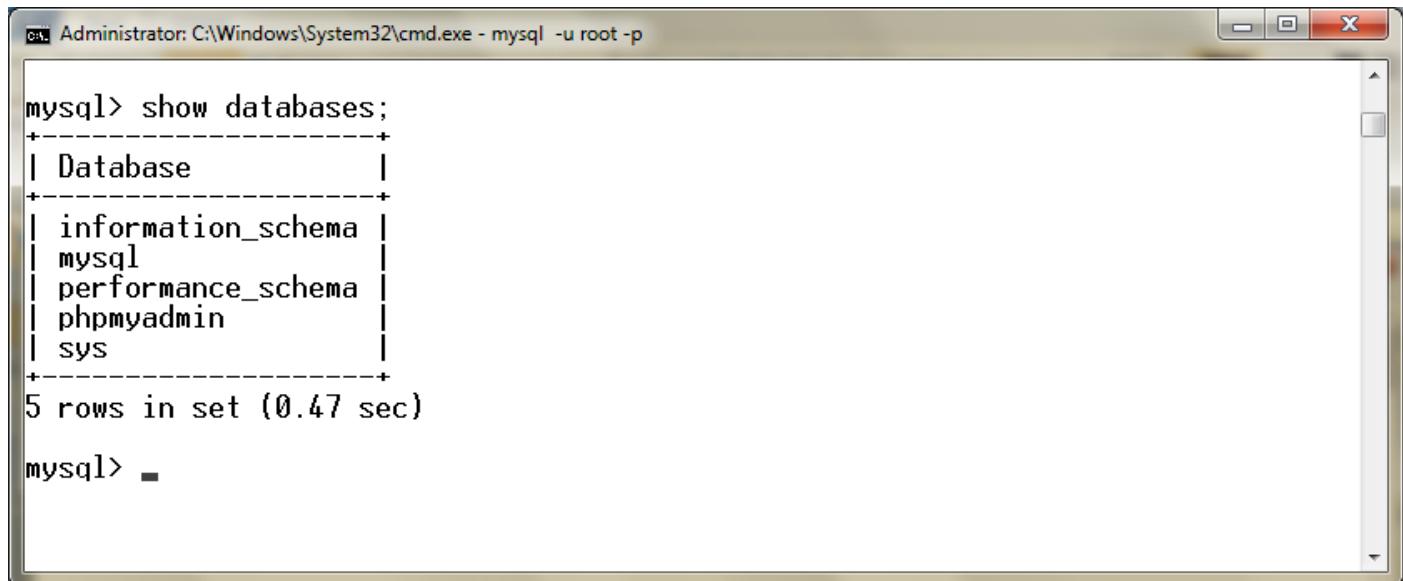
- `show databases;` - wyświetlanie baz danych
- `show grants for NazwaUżytkownika` - wyświetlanie informacji o użytkowniku
- `show errors;` - informacje o błędach
- `show privileges \G` – lista dostępny uprawnień
- `show open tables;` - lista otwartych table w systemie bazy
- `show processlist;` - wyświetlenie listy procesów
- `show tables;` - lista tabel wybranej bazie
- `show engine NazwaSilnika status;` - wyświetlanie informacji na temat danego silnika bazy danych
- `show triggers;` - informacje o wyzwalaczach
- `show variables;` - informacje na temat zmiennych systemowych
- `show warnings;` - informacje o ostrzeżeniach w bazie

## C. Podstawowe operacje w bazie

### i. Wyświetlanie listy baz na serwerze

Wykonujemy polecenie

```
show databases;
```



```
Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p

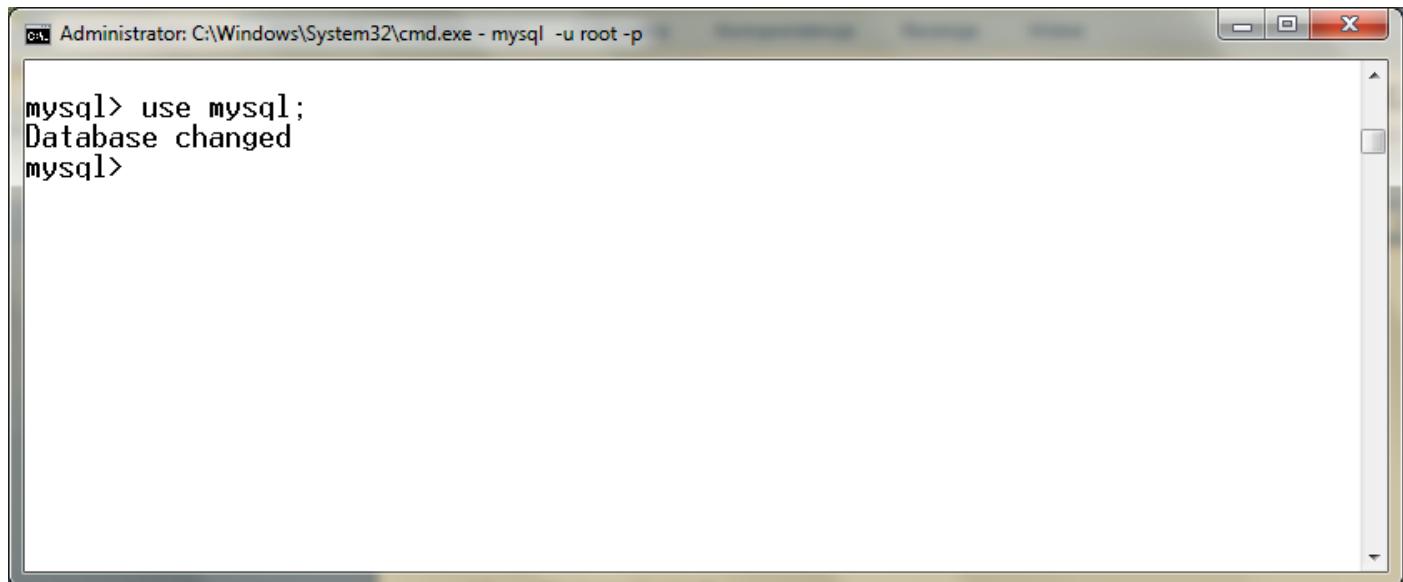
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| phpmyadmin |
| sys |
+-----+
5 rows in set (0.47 sec)

mysql> _
```

### ii. Zastosowanie wybranej bazy

Wykonujemy polecenie

```
use NazwaBazyDanych;
```



```
Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p

mysql> use mysql;
Database changed
mysql>
```

### iii. Wyświetlanie listy tabel danej bazy

Wykonujemy polecenie

`show tables;`

The screenshot shows a Windows Command Prompt window titled "Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p". The command entered is "mysql> show tables;". The output is a table titled "Tables\_in\_mysql" containing 31 rows of table names. The table structure is as follows:

Tables_in_mysql
columns_priv
db
engine_cost
event
func
general_log
gtid_executed
help_category
help_keyword
help_relation
help_topic
innodb_index_stats
innodb_table_stats
ndb_binlog_index
plugin
proc
procs_priv
proxies_priv
server_cost
servers
slave_master_info
slave_relay_log_info
slave_worker_info
slow_log
tables_priv
time_zone
time_zone_leap_second
time_zone_name
time_zone_transition
time_zone_transition_type
user

At the bottom of the output, it says "31 rows in set (0.06 sec)".

### iv. Wyświetlanie opisu wybranej tabeli

Wykonujemy polecenie:

`describe NazwaTabeli`

lub

`SHOW [FULL] COLUMNS FROM NazwaTabeli\G`

```

Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p
mysql> describe db;
+-----+-----+-----+-----+-----+-----+-----+
| Field          | Type           | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+-----+
| Host           | char(60)       | NO   | PRI  |          |       |
| Db             | char(64)       | NO   | PRI  |          |       |
| User           | char(32)       | NO   | PRI  |          |       |
| Select_priv    | enum('N','Y') | NO   |      |          | N     |
| Insert_priv   | enum('N','Y') | NO   |      |          | N     |
| Update_priv   | enum('N','Y') | NO   |      |          | N     |
| Delete_priv   | enum('N','Y') | NO   |      |          | N     |
| Create_priv   | enum('N','Y') | NO   |      |          | N     |
| Drop_priv     | enum('N','Y') | NO   |      |          | N     |
| Grant_priv    | enum('N','Y') | NO   |      |          | N     |
| References_priv | enum('N','Y') | NO   |      |          | N     |
| Index_priv    | enum('N','Y') | NO   |      |          | N     |
| Alter_priv    | enum('N','Y') | NO   |      |          | N     |
| Create_tmp_table_priv | enum('N','Y') | NO   |      |          | N     |
| Lock_tables_priv | enum('N','Y') | NO   |      |          | N     |
| Create_view_priv | enum('N','Y') | NO   |      |          | N     |
| Show_view_priv | enum('N','Y') | NO   |      |          | N     |
| Create_routine_priv | enum('N','Y') | NO   |      |          | N     |
| Alter_routine_priv | enum('N','Y') | NO   |      |          | N     |
| Execute_priv   | enum('N','Y') | NO   |      |          | N     |
| Event_priv     | enum('N','Y') | NO   |      |          | N     |
| Trigger_priv   | enum('N','Y') | NO   |      |          | N     |
+-----+-----+-----+-----+-----+-----+-----+
22 rows in set (0.67 sec)

mysql> -

```

```

Administrator: C:\Windows\System32\cmd.exe - startmysql.bat
mysql> SHOW COLUMNS FROM platnosci\G
***** 1. row *****
Field: NumerKlienta
Type: int(11)
Null: NO
Key: PRI
Default: NULL
Extra:
***** 2. row *****
Field: Numer
Type: varchar(50)
Null: NO
Key: PRI
Default: NULL
Extra:
***** 3. row *****
Field: DataPlatnosci
Type: date
Null: NO
Key:
Default: NULL
Extra:
***** 4. row *****
Field: kwota
Type: double
Null: NO
Key:
Default: NULL
Extra:
4 rows in set (0.05 sec)

mysql> -

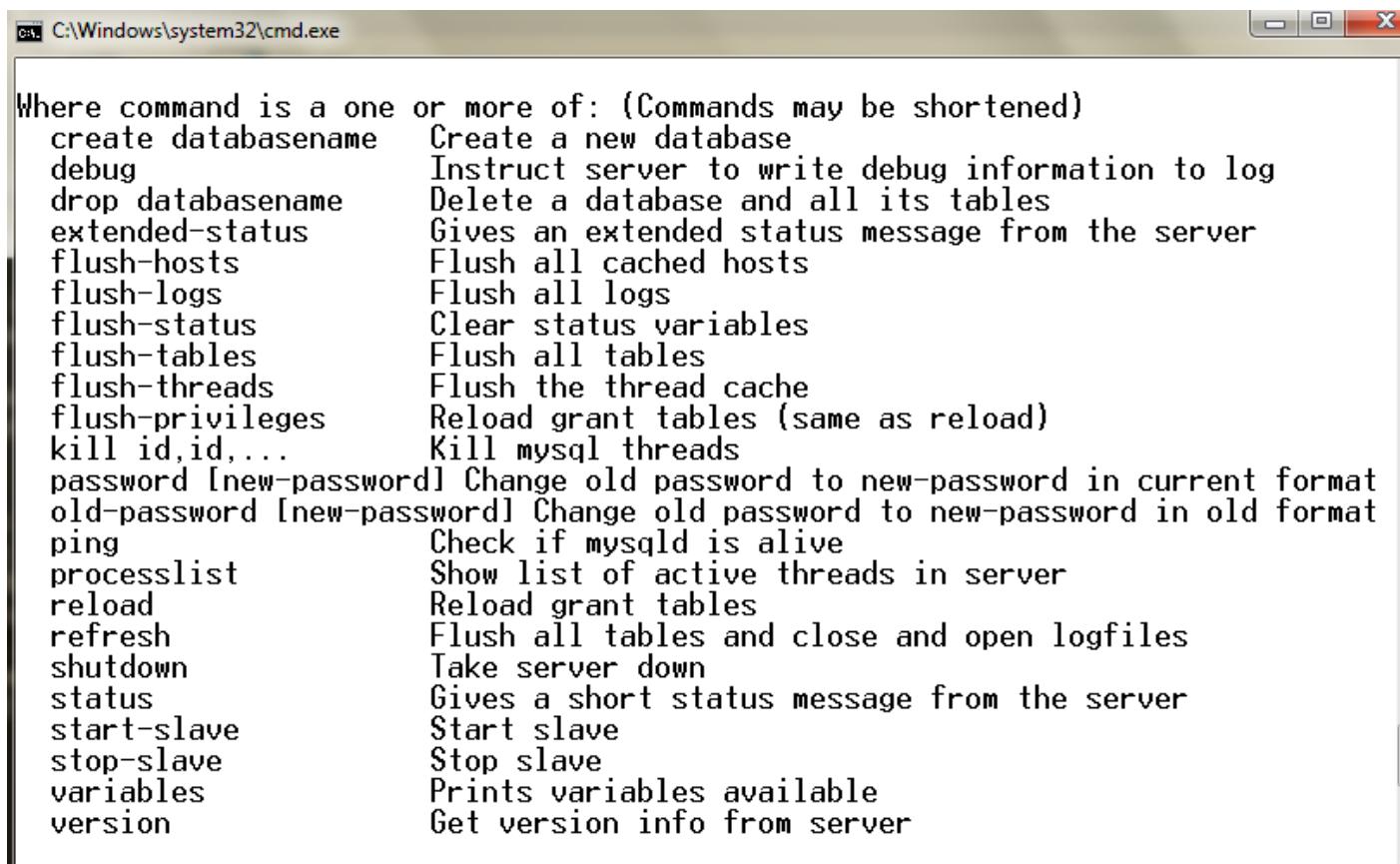
```

## 5. Magia polecenie mysqladmin

Składania polecenia:

```
mysqladmin --user =root -pHASLO command
```

Przykładowe opcje command:



```
C:\Windows\system32\cmd.exe

Where command is one or more of: (Commands may be shortened)
create database [name]      Create a new database
debug                      Instruct server to write debug information to log
drop database [name]        Delete a database and all its tables
extended-status             Gives an extended status message from the server
flush-hosts                 Flush all cached hosts
flush-logs                  Flush all logs
flush-status                Clear status variables
flush-tables                Flush all tables
flush-threads               Flush the thread cache
flush-privileges            Reload grant tables (same as reload)
kill id,id,...              Kill mysql threads
password [new-password]    Change old password to new-password in current format
old-password [new-password] Change old password to new-password in old format
ping                        Check if mysqld is alive
processlist                 Show list of active threads in server
reload                      Reload grant tables
refresh                     Flush all tables and close and open logfiles
shutdown                    Take server down
status                      Gives a short status message from the server
start-slave                 Start slave
stop-slave                  Stop slave
variables                   Prints variables available
version                     Get version info from server
```

### A. Ustawienie pierwszego hasła dla roota

Należy wykonać polecenie:

```
mysqladmin -u root password HASLO_DO_MYSQL_DLA_ROOT
```

Pamiętajmy aby przeładować uprawnienia:

```
mysqladmin -u root -p flush-privileges
```

### B. Zmiana hasła dla dowolnego konta

Ustanowienie hasła dla każdego innego użytkownika odbywa się w taki sam sposób jak dla Roota czyli:

```
mysqladmin -u USER_NAME password HASLO_DO_MYSQL
```

## C. Sprawdzenie czy serwer MySQL działa

Wykonujemy polecenie

```
mysqladmin -u root -p ping
```

```
C:\Program Files\MySQL\MySQL Server 5.7\bin>mysqladmin -u root -p ping
Enter password: *****
mysqld is alive

C:\Program Files\MySQL\MySQL Server 5.7\bin>
```

## D. Sprawdzenie wersji serwer MySQL

Wykonujemy polecenie

```
mysqladmin -u root -p version
```

```
C:\Program Files\MySQL\MySQL Server 5.7\bin>mysqladmin -u root -p version
Enter password: *****
mysqldadmin Ver 8.42 Distrib 5.7.9, for Win32 on AMD64
Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Server version      5.7.9-log
Protocol version   10
Connection          localhost via TCP/IP
TCP port            3306
Uptime:             5 hours 21 min 47 sec

Threads: 1  Questions: 13  Slow queries: 0  Opens: 109  Flush tables: 1  Open tables: 102  Queries per second avg: 0.000

C:\Program Files\MySQL\MySQL Server 5.7\bin>
```

## E. Sprawdzenie statusu serwera MySQL

Wykonujemy polecenie

```
mysqladmin -u root -p status
```

```
C:\> Administrator: C:\Windows\System32\cmd.exe
C:\> C:\Program Files\MySQL\MySQL Server 5.7\bin>mysqladmin -u root -p status
Enter password: *****
Uptime: 19398 Threads: 1 Questions: 15 Slow queries: 0 Opens: 109 Flush tables: 1 Open tables: 102 Queries per second avg: 0.000
C:\> C:\Program Files\MySQL\MySQL Server 5.7\bin>
```

## F. Sprawdzenie wartości zmiennych serwera MySQL

Wykonujemy polecenie

```
mysqladmin - u root -p variables
```

## G. Sprawdzenie procesów serwera MySQL

Wykonujemy polecenie

```
mysqladmin - u root -p processlist
```

```
C:\> Administrator: C:\Windows\System32\cmd.exe
C:\> C:\Program Files\MySQL\MySQL Server 5.7\bin>mysqladmin -u root -p processlist
Enter password: *****
+-----+-----+-----+-----+-----+-----+
| Id | User | Host           | db | Command | Time | State    | Info
+-----+-----+-----+-----+-----+-----+
| 13 | root | localhost:52062 |   | Query   | 0    | starting | show processlist
+-----+-----+-----+-----+-----+-----+
C:\> C:\Program Files\MySQL\MySQL Server 5.7\bin>
```

## I. Uruchamianie, zatrzymywanie serwera relikacji slave

Wykonujemy polecenie

```
mysqladmin - u root -p stop-slave
```

lub

```
mysqladmin - u root -p start-slave
```

## 6. Partycjonowanie tabeli

### A. Co to jest partycjonowanie?

Partycjonowanie jest fizycznym dzieleniem tabeli na części, do których jednak nie mamy bezpośredniego dostępu (nie ma możliwości wykonania zapytania SELECT tylko na wskazanej partycji, ale możemy np. wskazać, którą partycję chcemy wyczyścić). Każda partycja jest zarządzana przez silnik bazodanowy, ma własne indeksy, które zostały zdefiniowane na tabeli.

### B. Testowanie partycji

Checąc dokonywać partycjonowania tabel w MySQL należy sprawdzić czy dana wersja obsługuje partycje. Sposoby sprawdzenia czy możliwe jest partycjonowanie tabel:

a.

```
SHOW VARIABLES LIKE '%partition%';
```

Uwaga: Zmienna **have\_partitioning** nie występuje w wersji wyższej niż 5.6.1

b.

```
SHOW PLUGINS;
```

Plugin Name	Status	Schema	Version	Checksum	License
INNODB_SYS_FOREIGN	ACTIVE	INFORMATION SCHEMA	NULL	NULL	GPL
INNODB_SYS_FOREIGN_COLS	ACTIVE	INFORMATION SCHEMA	NULL	NULL	GPL
INNODB_SYS_TABLESPACES	ACTIVE	INFORMATION SCHEMA	NULL	NULL	GPL
INNODB_SYS_DATAFILES	ACTIVE	INFORMATION SCHEMA	NULL	NULL	GPL
INNODB_SYS_VIRTUAL	ACTIVE	INFORMATION SCHEMA	NULL	NULL	GPL
MyISAM	ACTIVE	STORAGE ENGINE	NULL	NULL	GPL
MRG_MYISAM	ACTIVE	STORAGE ENGINE	NULL	NULL	GPL
PERFORMANCE_SCHEMA	ACTIVE	STORAGE ENGINE	NULL	NULL	GPL
ARCHIVE	ACTIVE	STORAGE ENGINE	NULL	NULL	GPL
BLACKHOLE	ACTIVE	STORAGE ENGINE	NULL	NULL	GPL
FEDERATED	DISABLED	STORAGE ENGINE	NULL	NULL	GPL
partition	ACTIVE	STORAGE ENGINE	NULL	NULL	GPL
ngram	ACTIVE	FTPARSER	NULL	NULL	GPL

C.

```
SELECT PLUGIN_NAME as Name,
       PLUGIN_VERSION as Version,
       PLUGIN_STATUS as Status
  FROM INFORMATION_SCHEMA.PLUGINS
 WHERE PLUGIN_TYPE='STORAGE ENGINE';

mysql> SELECT PLUGIN_NAME as Name,
      ->   PLUGIN_VERSION as Version,
      ->   PLUGIN_STATUS as Status
      ->  FROM INFORMATION_SCHEMA.PLUGINS
      -> WHERE PLUGIN_TYPE='STORAGE ENGINE';
+-----+-----+-----+
| Name | Version | Status |
+-----+-----+-----+
| binlog | 1.0 | ACTIVE |
| CSV | 1.0 | ACTIVE |
| MEMORY | 1.0 | ACTIVE |
| InnoDB | 5.7 | ACTIVE |
| MyISAM | 1.0 | ACTIVE |
| MRG_MYISAM | 1.0 | ACTIVE |
| PERFORMANCE_SCHEMA | 0.1 | ACTIVE |
| ARCHIVE | 3.0 | ACTIVE |
| BLACKHOLE | 1.0 | ACTIVE |
| FEDERATED | 1.0 | DISABLED |
| partition | 1.0 | ACTIVE |
+-----+-----+-----+
11 rows in set (0.00 sec)
```

### C. Typy partycjonowania:

- **RANGE** - dla każdej partycji zdefiniowane jest wyrażenie, które zwraca wartość typu integer;
- **LIST** - jest bardzo podobne do powyższego sposobu partycjonowania - każda partycja musi być zdefiniowana. Różnica wynika z tego, że wyrażenie definiujące partycję jest listą wartości typu integer.
- **HASH** - używane głównie do równomiernego rozkładania wierszy w partycji między z góry określona liczbą ścieżek. Wyrażenie definiująca partycję musi zwracać wartość typu integer.
- **KEY** - bardzo podobna metoda do HASH, ale tutaj to MySQL narzuca wyrażenie partycjonujące przy pomocy funkcji hashującej.
- **Subpartitioning (composite partitioning czyli złożone partycjonowanie)** - jest dalszym podziałem partycji na jeszcze mniejsze partycje. (czytaj: <https://dev.mysql.com/doc/refman/5.5/en/partitioning-subpartitions.html>)

### D. Tworzenie partycji:

Składnia polecenia:

```
CREATE TABLE NazwaTabeli  
    (Kolumny Typ)  
    ENGINE = Nazwa Silnika  
    PARTITION BY TypPartycji  
    (OPIS PARTYCJONOWANIA)
```

Przykłady:

- a. Partycja typu range

```
Create Table PartycjaRange  
(  
Id int,  
Name varchar (20),  
DataZatrudnienia date  
)  
PARTITION BY RANGE( YEAR(DataZatrudnienia) )  
(  
PARTITION p0 VALUES LESS THAN (1990),  
PARTITION p1 VALUES LESS THAN (1995),  
PARTITION p2 VALUES LESS THAN (2000),  
PARTITION p3 VALUES LESS THAN (2005)  
);
```

- b. Partycja typu list

```
Create Table PartycjaList  
(  
Id int,  
Name varchar (20),  
DataZatrudnienia date  
)  
PARTITION BY List( YEAR(DataZatrudnienia) )  
(  
PARTITION p0 VALUES IN (1990,1991,1992,1993,1994),  
PARTITION p1 VALUES IN (1995,1996,1997,1998),  
PARTITION p2 VALUES IN (1999,2000),  
PARTITION p4 VALUES IN (2001,2002,2003),  
PARTITION p5 VALUES IN (2005)  
);
```

```

mysql> desc partycjalist;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Id    | int(11) | YES |   | NULL |   |
| Name  | varchar(20) | YES |   | NULL |   |
| DataZatrudnienia | date | YES |   | NULL |   |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

```

Utworzone partycje fizycznie zobaczymy jedynie na dysku, gdzie przechowywane są bazy:

Nazwa	Data modyfikacji	Typ	Rozmiar
db.opt	2017-11-28 00:49	Plik OPT	1 KB
partycjalist#p#p0.ibd	2017-11-28 00:56	Plik IBD	96 KB
partycjalist#p#p1.ibd	2017-11-28 00:56	Plik IBD	96 KB
partycjalist#p#p2.ibd	2017-11-28 00:56	Plik IBD	96 KB
partycjalist#p#p4.ibd	2017-11-28 00:56	Plik IBD	96 KB
partycjalist#p#p5.ibd	2017-11-28 00:56	Plik IBD	96 KB
partycjalist.frm	2017-11-28 00:56	Plik FRM	9 KB
partycjarange#p#p0.ibd	2017-11-28 00:51	Plik IBD	96 KB
partycjarange#p#p1.ibd	2017-11-28 00:51	Plik IBD	96 KB
partycjarange#p#p2.ibd	2017-11-28 00:51	Plik IBD	96 KB
partycjarange#p#p3.ibd	2017-11-28 00:51	Plik IBD	96 KB
partycjarange.frm	2017-11-28 00:51	Plik FRM	9 KB

## 7. Użytkownicy i uprawnienia

Na serwerze MySQL istnieje 6 tabel związanych z uprawnieniami, znajdują się w bazie mysql:

Tabela uprawnień	Zawartość tabeli
<b>user</b>	Dane użytkowników, którzy mogą nawiązać połączenie z serwerem i ich globalne uprawnienia
<b>db</b>	Uprawnienia bazy danych
<b>tables_priv</b>	Uprawnienia tabel
<b>columns_priv</b>	Uprawnienia kolumn
<b>procs_priv</b>	Uprawnienia procedur składowanych
<b>proxies_priv</b>	Uprawnienia użytkowników proxy

## A. Zarządzanie kontem MySQL

Istnieją trzy polecenia pozwalające na przeprowadzenie wysokiego poziomu operacji na kontach MySQL:

- `create user`
- `drop user`
- `rename user`

### I. Tworzenie użytkownika – `create user`

Składnia polecenia:

```
create user 'NazwaUżytkownika'@'NazwaKomputera' identified by'Hasło';
```

Przykład:

```
create user 'student'@'%' identified by 'student';
```

```
create user 'student'@'localhost' identified by 'student';
```

```
create user 'student'@'192.168.1.3' identified by 'student';
```

```
create user 'student'@'192.168.1.%' identified by 'student';
```

```
create user 'student'@'192.168.1.0/255.255.255.0' identified by 'student';
```

```
create user 'student'@'localhost' identified by 'student' PASSWORD EXPIRE NEVER;
```

```
create user 'student'@'localhost' identified by 'student' PASSWORD EXPIRE INTERVAL 180 DAY;
```

```
create user 'jeffrey'@'localhost' REQUIRE SSL;
```

Podane przykłady tworzą konto student z hasłem student, ale różnią się sposobem dostępu do bazy danych.

## B. Język DCL (Data Control Language)

### I. Nadawanie uprawnień -- grant

Składnia polecenia:

```
grant uprawnienia [kolumny] ON poziom_uprawnien TO użytkownik
```

### Uprawnienia:

<b>Znaczenie</b>	
<b>ALL [PRIVILEGES]</b>	nadaje wszystkie uprawnienia z wyjątkiem GRANT OPTION
<b>ALTER</b>	zezwala na użycie instrukcji ALTER TABLE
<b>ALTER ROUTINE</b>	pozwala zmianę lub usunięcie składowanych funkcji i procedur
<b>CREATE</b>	zezwala na użycie instrukcji CREATE TABLE
<b>CREATE ROUTINE</b>	pozwala tworzenie składowanych funkcji i procedur
<b>CREATE TEMPORARY TABLES</b>	zezwala na użycie instrukcji CREATE TEMPORARY TABLE
<b>CREATE TABLESPACE</b>	zezwala na tworzenie, usuwanie i zmianę przestrzeni tabel
<b>CREATE USER</b>	zezwala na użycie instrukcji: CREATE USER, DROP USER, RENAME USER i REVOKE ALL PRIVILEGES
<b>CREATE VIEW</b>	pozwala na tworzenie widoków
<b>DELETE</b>	zezwala na użycie instrukcji DELETE
<b>DROP</b>	zezwala na użycie instrukcji DROP TABLE
<b>EVENT</b>	pozwala na pracę z harmonogramem zdarzeń
<b>EXECUTE</b>	pozwala na wykonywanie składowanych funkcji i procedur
<b>FILE</b>	zezwala na użycie instrukcji SELECT ... INTO OUTFILE i LOAD DATA INFILE
<b>INDEX</b>	zezwala na użycie instrukcji CREATE INDEX i DROP INDEX
<b>INSERT</b>	zezwala na użycie instrukcji INSERT
<b>LOCK TABLES</b>	zezwala na użycie instrukcji LOCK TABLES
<b>PROCESS</b>	zezwala na użycie instrukcji SHOW PROCESSLIST
<b>PROXY</b>	zezwala na użycie PROXY
<b>REFERENCES</b>	<b>Już nie stosowane</b>
<b>RELOAD</b>	zezwala na użycie instrukcji FLUSH
<b>REPLICATION CLIENT</b>	zezwala użytkownikowi na odpytywanie i wyszukiwanie serwera master i slave
<b>REPLICATION SLAVE</b>	zezwala użytkownikowi na odczytywaniu binarnych logów z servera master
<b>SELECT</b>	zezwala na użycie instrukcji SELECT
<b>SHOW DATABASES</b>	instrukcja SHOW DATABASES wyświetla wszystkie bazy
<b>SHOW VIEW</b>	pozwala na wykonanie zapytań SHOW CREATE VIEW w celu wyświetlenia definicji widoku
<b>SHUTDOWN</b>	zezwala na zamknięcie serwera mysql
<b>SUPER</b>	zezwala na wykonywanie operacji takich jak: CHANGE MASTER TO, KILL, PURGE BINARY LOGS, SET GLOBAL i mysqladmin debug
<b>TRIGGER</b>	pozwala na dodawanie i usuwanie wyzwalaczy
<b>UPDATE</b>	zezwala na użycie instrukcji UPDATE
<b>USAGE</b>	synonim dla "brak przywilejów"
<b>GRANT OPTION</b>	zezwala na dalsze przekazywanie uprawnień

## Poziom uprawnień

Uprawnienia	Poziom, na którym zastosowanie mają dane uprawnienia
<b>ON *.*</b>	Uprawnienia globalne: wszystkie bazy danych i wszystkie obiekty w bazach danych
<b>ON *</b>	Uprawnienia na poziomie domyślnej bazy danych. W przypadku braku domyślnej bazy danych zgłoszany jest błąd
<b>ON NazwaBazyDanych.*</b>	Uprawnienia na określonej bazie danych. Wszystkie obiekty bazy
<b>ON NazwaBazyDanych.NazwaTabeli</b>	Uprawnienia tabeli . Wszystkie kolumny tabeli
<b>ON NazwaTabeli</b>	Uprawnienia tabeli . Wszystkie kolumny tabeli w domyślnej bazie danych
<b>ON NazwaBazyDanych.NazwaProcedury</b>	Uprawnienia dla wskazanej procedury w określonej bazie danych
<b>ON Uzytkownik</b>	Uprawnienia proxy

## II. Odbieranie uprawnień -- revoke

Składnia polecenia:

```
revoke uprawnienia [kolumny] ON poziom_uprawnien FROM użytkownik
```

## III. Sprawdzenie uprawnień użytkownika

Sposób I:

```
SHOW GRANTS FOR nazwa_użytkownika@komputer;
```

The screenshot shows a Windows Command Prompt window titled "Administrator: C:\Windows\System32\cmd.exe - startmysql.bat". The MySQL prompt is visible, and the command "show grants for root@localhost;" is being run. The output displays the grants for the 'root' user on 'localhost', showing two rows of grants: one for all privileges on all databases and another for proxy privileges.

```
Administrator: C:\Windows\System32\cmd.exe - startmysql.bat
mysql> show grants for root@localhost;
+-----+
| Grants for root@localhost |
+-----+
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' WITH GRANT OPTION |
| GRANT PROXY ON ''@'' TO 'root'@'localhost' WITH GRANT OPTION |
+-----+
2 rows in set (0.00 sec)

mysql> -
```

Sposób II:

Wykonanie zapytania:

```
SELECT Host, User FROM mysql.user
```

lub

```
SELECT * FROM mysql.user \G
```

## 7. Typy danych w bazie MySQL

Typ danych	Długość	Opis
TINYINT	1 byte	zakres wartości od -128 do +127, bez znaku (UNSIGNED) 0 do 255
SMALLINT	2 bytes	zakres wartości od -32 768 do +32 767, bez znaku (UNSIGNED) 0 do 65 535
MEDIUMINT	3 bytes	zakres wartości od -8 388 608 do +8 388 607, bez znaku (UNSIGNED) 0 do 16 777 215
INT,INTEGER	4 bytes	zakres wartości od -2147483648 do +2 147 483 647, bez znaku (UNSIGNED) 0 do 4 294 967 295
BIGINT	8 bytes	zakres wartości od -9 223 372 036 854 775 808 do +9 223 372 036 854 775 807, bez znaku (UNSIGNED) 0 do 18 446 744 073 709 551 615
BIT, BOOL	1 byte	synonim dla TINYINT(1)
FLOAT		zakres wartości od -3.402823466E+38 do 3.402823466E+38
DOUBLE		zakres wartości od -1.7976931348623157E+308 do 1.7976931348623157E+308
DOUBLE PRECISION, REAL		synonimy dla typu DOUBLE
DECIMAL(m,d), DEC(m,d), NUMERIC(m,d)		zakres ustawia parametry "m" i "d", maksymalny zakres jest taki sam jak dla typu DOUBLE
DATE	'0000-00-00'	data w formacie "rok-miesiąc-dzień" względnie "RRRR-MM-DD" i w zakresie 1000-01-01 aż 9999-12-31
DATETIME	'0000-00-00 00:00:00'	data i czas w zakresie 1000-01-01 00:00:00 do 9999-12-31 23:59:59 (format = "RRRR-MM-DD HH:MM:SS")
TIMESTAMP(n)	'0000-00-00 00:00:00'	<ul style="list-style-type: none"> <li>- data i czas w zakresie 1970-01-01 00:00:00 do 2037-01-01 00:00:00 (zapisywanych jest zawsze wszystkich 14 liczb !)</li> <li>- forma t wyświetlenia (i dla zapytań) można ustawić przy pomocy parametru "m" o wartości 14 (lub bez wartości), 12, 10, 8, 6, 4, lub 2</li> <li>- "RRRRMMDDHHMMSS", "RRMMDDHHMMSS", "RRMMDDHHMM", "RRRRMMDD", "RRMMDD", "YYMM", "YY"</li> <li>- jeżeli do pola bazy danych tego typu nie zostanie zapisana żadna wartość, to MySQL sam uzupełni bieżący czas zmiany do danego wiersza</li> </ul>
TIME	'00:00:00'	zakres czasu wynosi od "-838:59:59" do "838:59:59" a format typu danych "HH:MM:SS"
YEAR(n)	0000	dla typu YEAR(4) będzie zakres 1901 do 2155, format = "RRRR", dla YEAR(2) będzie zakres 1970-2069
CHAR(m)		<ul style="list-style-type: none"> <li>- długość łańcucha "m" może być w zakresie 0-255 - jeżeli zapisywany łańcuch będzie krótszy niż jest ustawione, to zostanie automatycznie uzupełniony o spacje (ma zatem "stałą" wielkość)- CHAR (a więc bez "m") jest uważane za CHAR(1)</li> </ul>
VARCHAR(m)		długość łańcucha "m" może być w zakresie 0-255- jeżeli zapisywany łańcuch jest krótszy niż jest ustawione, to brakujące znaki nie są uzupełniane (łańcuch ma zatem zmienną długość), lecz dodatkowo jest zapisywana informacja o jego długości
TINYBLOB, TINYTEXT		Długość łańcucha wynosi maksymalnie 255 znaków
BLOB, TEXT		Długość łańcucha wynosi maksymalnie 65 535 znaków
MEDIUMBLOB, MEDIUMTEXT		Długość łańcucha wynosi maksymalnie 16 777 215 znaków
LONGBLOB, LONGTEXT		Długość łańcucha wynosi maksymalnie 4 294 967 295 znaków
ENUM('item1','item2',...)		<ul style="list-style-type: none"> <li>- tablica z góry przygotowanych łańcuchów (elementów) o maksymalnej ilości 65 535</li> </ul>

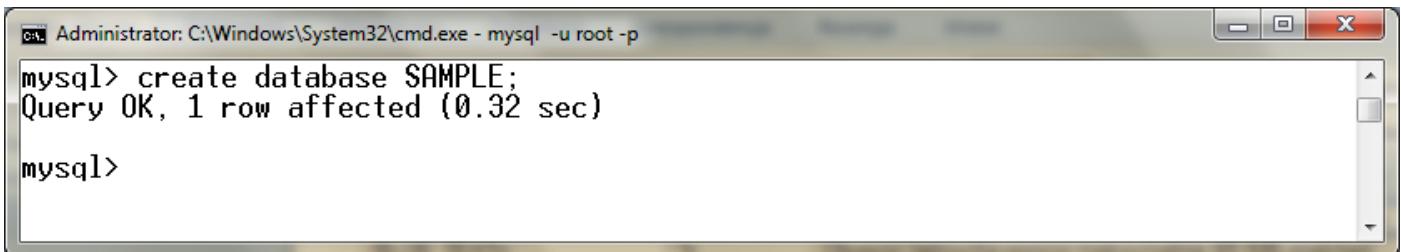
		<ul style="list-style-type: none"> <li>- w polu tabeli może się znaleźć tylko jeden z itemów, które zostały z góry przygotowane</li> <li>- zamiast nazwy 'item' można stosować również ich kolejność, a więc: 1 (zamiast 'item1'), 2 (zamiast 'item2')...</li> </ul>
SET('item1','item2',...)		<ul style="list-style-type: none"> <li>- tabela z góry przygotowanych łańcuchów (itemów) o maksymalnej ilości 64</li> <li>- w polu tabeli może się znaleźć kilka itemów, które są z góry przygotowane</li> </ul>

## 8. Język DDL

### I. Tworzenie bazy danych

Składnia polecenia:

```
create database NAZWA_BAZY;
```



```
Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p
mysql> create database SAMPLE;
Query OK, 1 row affected (0.32 sec)

mysql>
```

Jeżeli próbujemy nadać tworzonej bazie nazwę, która już istnieje, to wystąpi błąd. Można się przed tym błędem zabezpieczyć używając klauzuli IF NOT EXISTS.

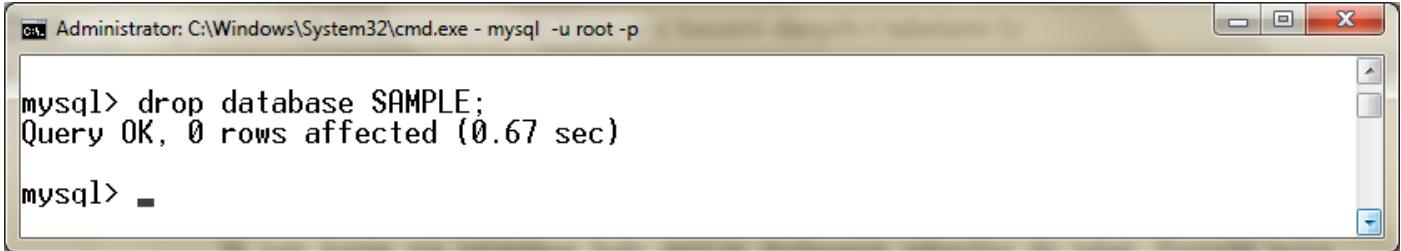
Przykład:

```
create database if not exists SAMPLE;
```

### II. Usuwanie bazy danych

Składnia polecenia:

```
drop database NAZWA_BAZY;
```



```
Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p
mysql> drop database SAMPLE;
Query OK, 0 rows affected (0.67 sec)

mysql> -
```

### III. Tworzenie tabeli bazy danych

Składnia polecenia:

```
create table [if not exists ]NAZWA_TABELI
```

```

(
    nazwa_pola_1 typ danych atrybut,
    nazwa_pola_2 typ danych atrybut,
    ...
)
[OPCJE];

```

nawiasami kwadratowymi obejmujemy elementy opcjonalne.

#### I. Typy danych w/g tabeli z punktu 7.

#### II. Atrybuty kolumny:

- **NOT NULL** - powoduje, że wartość w danym polu nie może być null.
- **AUTO\_INCREMENT** - auto numerowanie w kolumnie, każdy rekord automatycznie podczas insertu odtrzymuje kolejny numer. Uwaga: ten atrybut może być użyty tylko z zawęźaniem PRIMARY KEY
- **PRIMARY KEY** - powoduje, że dane w kolumnie nie mogą się powtarzać, służy do identyfikacji rekordu.
- **FOREIGN KEY** - odwołanie do klucza głównego z innej tabeli.
- **UNIQUE** - powoduje, że dane w kolumnie nie mogą się powtarzać.
- **DEFAULT** - domyślna wartość dla pola, w przypadku nie podania wartości dla kolumny w rekordzie zostanie zainsertowana wartość default.
- **UNSIGNED** - powoduje, że kolumna nie może przechowywać wartości na minusie przy czym zakres pozostaje taki sam, działa tylko dla typów przechowujących liczby całkowite.
- **ZEROFILL** - czyli zerowe wypełnienie, w przypadku gdy ilość liczb w polu będzie mniejsza niż ta zadeklarowana przy tworzeniu kolumny wartość pola będzie automatycznie "dopełniana" zerami na początku, działa tylko dla typów przechowujących liczby całkowite, automatycznie tworzy atrybut UNSIGNED.

#### III. OPCJE TABELI to:

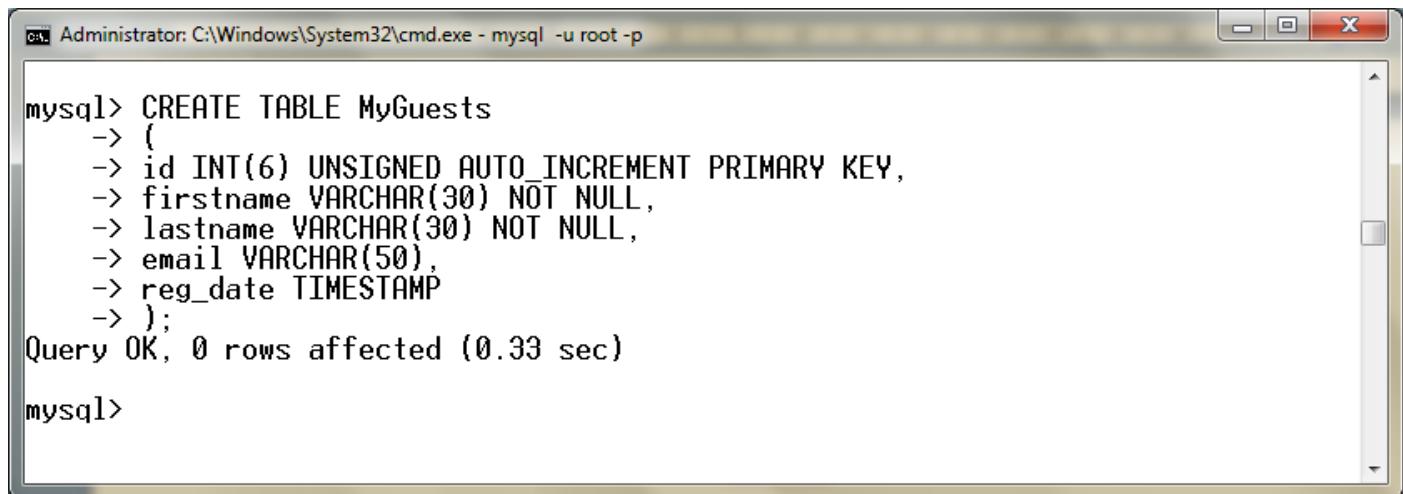
```

| ENGINETYPE [=] engine_name
| AUTO_INCREMENT [=] value
| AVG_ROW_LENGTH [=] value
| [DEFAULT] CHARACTER SET [=] charset_name
| CHECKSUM [=] {0 | 1}
| [DEFAULT] COLLATE [=] collation_name
| COMMENT [=] 'string'
| CONNECTION [=] 'connect_string'
| DATA DIRECTORY [=] 'absolute path to directory'
| DELAY_KEY_WRITE [=] {0 | 1}
| INDEX DIRECTORY [=] 'absolute path to directory'
| INSERT_METHOD [=] { NO | FIRST | LAST }
| MAX_ROWS [=] value
| MIN_ROWS [=] value
| PACK_KEYS [=] {0 | 1 | DEFAULT}
| PASSWORD [=] 'string'
| ROW_FORMAT [=] {DEFAULT|DYNAMIC|FIXED|COMPRESSED|REDUNDANT|COMPACT}

```

| UNION [=] (*tbl\_name*[,*tbl\_name*]...)

Przykład I:



```
Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p
mysql> CREATE TABLE MyGuests
-> (
-> id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
-> firstname VARCHAR(30) NOT NULL,
-> lastname VARCHAR(30) NOT NULL,
-> email VARCHAR(50),
-> reg_date TIMESTAMP
-> );
Query OK, 0 rows affected (0.33 sec)

mysql>
```

#### IV. Realizacja kluczy obcych w MySQL:

[CONSTRAINT [*symbol*]] FOREIGN KEY [*nazwa indeksu*]  
(*nazwa kolumny indeksowej*,...)

REFERENCES *nazwa tabeli* [(*nazwa kolumny indeksowej*,...)]

[ON DELETE *reference\_option*]

[ON UPDATE *reference\_option*]

#### Znaczenie:

CONSTRAINT – następuje zdefiniowanie ograniczenia klucza. Jeżeli pominiemy nazwę ograniczenia to silnik bazy MySQL utworzy ją sam.

Wskazówka:

Aby wyświetlić utworzone ograniczenia w danej bazie należy użyć zapytania:

```
SELECT constraint_name, table_schema, table_name, constraint_type
FROM information_schema.table_constraints
WHERE table_shema ='Nazwa bazy';
```

```
Administrator: C:\Windows\System32\cmd.exe - startmysql.bat

mysql> SELECT constraint_name, table_schema, table_name, constraint_type
-> FROM information_schema.table_constraints
-> Where table_schema = 'mojaxy';
+-----+-----+-----+-----+
| constraint_name | table_schema | table_name | constraint_type |
+-----+-----+-----+-----+
| PRIMARY        | mojaxy     | x          | PRIMARY KEY      |
| PRIMARY        | mojaxy     | y          | PRIMARY KEY      |
| y_ibfk_1       | mojaxy     | y          | FOREIGN KEY      |
+-----+-----+-----+-----+
3 rows in set (0.02 sec)

mysql> -
```

ON DELETE — przy usuwaniu krotki sprawdza stan tabel powiązanych i określa, co się z nimi dzieje.

ON UPDATE — przy modyfikacji krotki (atrybutu wchodzącego w skład klucza obcego) sprawdza stan tabel powiązanych i określa, co się z nimi dzieje.

*reference\_option:*

RESTRICT | CASCADE | SET NULL | NO ACTION

RESTRICT oznacza że usuwana tabela, kolumna, perspektywa nie może być w danej chwili używana przez jakikolwiek obiekt w bazie.

CASCADE — nakazuje zmianom na propagację kaskadową wzduż drzewa powiązanych tabel (potencjalnie bardzo niebezpieczne przy usuwaniu krotek!).

SET NULL — ustawia odpowiednie atrybuty powiązanych tabel, dotąd wskazujące na usuwany/modyfikowany element klucza obcego, na wartość NULL, jeśli definicja tabeli to dopuszcza.

NO ACTION — wyłącza mechanizm klucza obcego dla danej operacji.

Przykład II:

W przykładzie pokazano jak w momencie tworzenia tabeli utworzyć klucz główny (PRIMARY KEY) i klucz obcy (FOREIGN KEY):

```
CREATE TABLE X
(
  IdX TINYINT UNSIGNED NOT NULL PRIMARY KEY
);
```

```
CREATE TABLE Y
(
  IdY CHAR(1) NOT NULL PRIMARY KEY,
  IdX TINYINT UNSIGNED NOT NULL,
```

FOREIGN KEY (Idx) REFERENCES X (Idx) ON DELETE CASCADE ON UPDATE CASCADE );

```
Administrator: C:\Windows\System32\cmd.exe - startmysql.bat
mysql> desc X;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| IdX  | tinyint(3) unsigned | NO   | PRI | NULL    |       |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> desc Y;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| IdY  | char(1) | NO   | PRI | NULL    |       |
| IdX  | tinyint(3) unsigned | NO   | MUL | NULL    |       |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

#### Uwaga:

Klucze obce można tworzyć tylko dla bazy z silnikiem InnoDB, od MySQL 5.5 jest to domyślny silnik bazy.

#### Przykład III:

W przykładzie tym pokazano definiowanie klucza głównego oraz kodowanie znaków dla tabeli Biuro

```
CREATE TABLE Biuro (
    biuroNr VARCHAR(4) NOT NULL,
    ulica VARCHAR(25) NOT NULL,
    miasto VARCHAR(25) NOT NULL,
    kod VARCHAR(6),
    PRIMARY KEY (biuroNr)
)ENGINE = InnoDB
DEFAULT CHARSET=utf8 COLLATE=utf8_polish_ci;
```

## IV. Wyświetlenie sposobu tworzenia tabeli:

Składnia polecenia:

```
show create table NazwaTabeli\G
```

```
Administrator: C:\Windows\System32\cmd.exe - startmysql.bat
mysql> show create table platnosci\G
***** 1. row *****
  Table: platnosci
Create Table: CREATE TABLE `platnosci` (
  `NumerKlienta` int(11) NOT NULL,
  `Numer` varchar(50) NOT NULL,
  `DataPlatnosci` date NOT NULL,
  `kwota` double NOT NULL,
  PRIMARY KEY (`NumerKlienta`, `Numer`)
) ENGINE=MEMORY DEFAULT CHARSET=utf8
1 row in set (0.00 sec)

mysql> -
```

## V. Usuwanie tabeli bazy danych

Składnia polecenia:

```
drop table NAZWA_TABELI;
```

## VI. Aktualizowanie tabeli bazy danych

Jeżeli po utworzeniu tabeli konieczne są zmiany w strukturze tabeli należy zastosować polecenie:

```
alter table Nazwa_Tabeli OPERACJA
```

Szczegóły opcji OPERACJA można znaleźć na stronie: <http://dev.mysql.com/doc/refman/5.7/en/alter-table.html>.

Przykłady:

Do przykładów będzie wykorzystana przykładowa tabela Moja:

```
CREATE TABLE Moja (
    id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
    email VARCHAR(40) NULL,
    category SMALLINT NULL,
    is_active TINYINT NOT NULL DEFAULT '0',
    PRIMARY KEY (id)
);
```

```
Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p
mysql> CREATE TABLE Moja (
    -> id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
    -> email VARCHAR(40) NULL,
    -> category SMALLINT NULL,
    -> is_active TINYINT NOT NULL DEFAULT '0',
    -> PRIMARY KEY (id)
    -> );
Query OK, 0 rows affected (0.83 sec)

mysql> describe Moja
-> ;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id   | int(10) unsigned | NO  | PRI | NULL    | auto_increment |
| email | varchar(40)      | YES |     | NULL    |                |
| category | smallint(6)      | YES |     | NULL    |                |
| is_active | tinyint(4)       | NO  |     | 0       |                |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.03 sec)

mysql> -
```

- Zmiana nazwy tabeli

Alter table Moja Rename to Testowa;

```
Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p
mysql> alter table Moja rename to Testowa;
Query OK, 0 rows affected (0.34 sec)

mysql>
```

- Dodanie kolumny:

ALTER TABLE Testowa  
ADD plec VARCHAR(1) NOT NULL DEFAULT 'k';

```

Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p

mysql> ALTER TABLE Testowa
    -> ADD plec VARCHAR(1) NOT NULL DEFAULT 'k';
Query OK, 0 rows affected (1.04 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
mysql> describe Testowa
-> ;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra       |
+-----+-----+-----+-----+-----+-----+
| id   | int(10) unsigned | NO  | PRI | NULL    | auto_increment |
| email | varchar(40) | YES |     | NULL    |             |
| category | smallint(6) | YES |     | NULL    |             |
| is_active | tinyint(4) | NO  |     | 0       |             |
| plec | varchar(1) | NO  |     | k       |             |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.03 sec)

mysql>

```

Można równie dodać kolumnę na początku, wówczas należy na końcu polecenia dodawać **FIRST**.

Alter Table Testowa add T1 varchar(10) FIRST;

Jeżeli chcemy dodać kolumnę po określonej kolumnie, wówczas na końcu polecenia dodajemy **AFTER NazwaKolumny**.

Alter Table Testowa add T2 varchar(10) AFTER email;

```

Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p

mysql> describe Testowa;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra       |
+-----+-----+-----+-----+-----+-----+
| T1   | varchar(10) | YES |     | NULL    |             |
| id   | int(10) unsigned | NO  | PRI | NULL    | auto_increment |
| email | varchar(40) | YES |     | NULL    |             |
| T2   | varchar(10) | YES |     | NULL    |             |
| category | smallint(6) | YES |     | NULL    |             |
| is_active | tinyint(4) | NO  |     | 0       |             |
| plec | varchar(1) | NO  |     | k       |             |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)

mysql> -

```

- Modyfikacja w kolumny

Alter Table Testowa modify T2 varchar(20) not null;

```

Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p

mysql> Alter Table Testowa modify T2 varchar(20) not null;
Query OK, 0 rows affected (0.82 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> describe Testowa;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| T1    | varchar(10) | YES  |     | NULL    |          |
| id    | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| email | varchar(40)  | YES  |     | NULL    |          |
| T2    | varchar(20)  | NO   |     | NULL    |          |
| category | smallint(6) | YES  |     | NULL    |          |
| is_active | tinyint(4)  | NO   |     | 0       |          |
| plec  | varchar(1)   | NO   |     | k       |          |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)

```

- Zmiana nazwy kolumny

ALTER TABLE Testowa CHANGE COLUMN T2 T2\_NowaNazwa varchar(40);

```

Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p

mysql> ALTER TABLE Testowa CHANGE COLUMN T2 NowaNazwa varchar(40);
Query OK, 0 rows affected (0.97 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe Testowa;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id    | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| email | varchar(40)  | YES  |     | NULL    |          |
| NowaNazwa | varchar(40)  | YES  |     | NULL    |          |
| category | smallint(6) | YES  |     | NULL    |          |
| is_active | tinyint(4)  | NO   |     | 0       |          |
| plec  | varchar(1)   | NO   |     | k       |          |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.02 sec)

mysql> -

```

- Usuwanie kolumny

Alter Table Testowa drop T1;

```

Administrator: C:\Windows\System32\cmd.exe - mysql -u root -p

mysql> Alter Table Testowa drop T1;
Query OK, 0 rows affected (0.61 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe Testowa;
+-----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra       |
+-----+-----+-----+-----+-----+-----+
| id   | int(10) unsigned | NO  | PRI | NULL    | auto_increment |
| email | varchar(40) | YES |     | NULL    |             |
| T2   | varchar(20)  | NO  |     | NULL    |             |
| category | smallint(6) | YES |     | NULL    |             |
| is_active | tinyint(4)  | NO  |     | 0       |             |
| plec  | varchar(1)   | NO  | K   |         |             |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

```

## VII. Czyszczenie tabeli bazy danych

Jeśli chcielibyśmy wykasować wszystkie dane z tabeli, nie kasując struktury tabeli, należałoby się posłużyć poleceniem:

truncate NAZWA\_TABELI;

## 9. Testowanie tabel - polecenie mysqlcheck

Składnia polecenia:

mysqlcheck --u root -p OPCJE NazwaBAZY

- **OPCJE:**

- c (--check) – sprawdzanie poprawności
- o (--optimize) – optymalizacja
- r (--repair) – naprawa
- a (--analyze) - analiza

- **NazwaBAZY:** nazwa bazy lub –all-databases

## 10. Silniki bazy MySQL

Który silnik bazodanowy wybrać podczas tworzenia nowej bazy danych? To pytanie pada bardzo często na etapie projektowania nowej aplikacji czy systemu, który będzie korzystał z bazy MySQL.

Przed wybraniem silnika należy rozważyć jakie wady i zalety prezentuje każdy z nich. Inaczej na bazę danych będzie patrzył klient, któremu zależy na jej wydajności, a inaczej programista czy administrator, który będzie martwił się o kopie zapasowe danych czy stabilność całego serwera.

Silniki dostępne bazy danych dostępne w MySQL:

- ARCHIVE
- BLACKHOLE
- CSV
- FEDERATED
- InnoDB
- MEMORY
- MERGE
- MyISAM
- NDB

## A. Ustalanie dostępnych silników bazy danych

Dostępność silników bazy danych dla danego serwera tak naprawdę zależy od konkretnej wersji MySQL.

Aby przekonać się jakie silniki mamy dostępne na serwerze wykonujemy zapytanie:

**SHOW ENGINES;**

Engine	Support	Comment	Transactions	XA	Savepoints
InnoDB	DEFAULT	Supports transactions, row-level locking, and foreign keys	YES	YES	YES
MRG_MYISAM	YES	Collection of identical MyISAM tables	NO	NO	NO
MEMORY	YES	Hash based, stored in memory, useful for temporary tables	NO	NO	NO
BLACKHOLE	YES	/dev/null storage engine (anything you write to it disappears)	NO	NO	NO
MyISAM	YES	MyISAM storage engine	NO	NO	NO
CSV	YES	CSV storage engine	NO	NO	NO
ARCHIVE	YES	Archive storage engine	NO	NO	NO
PERFORMANCE_SCHEMA	YES	Performance Schema	NO	NO	NO
FEDERATED	NO	Federated MySQL storage engine	NULL	NULL	NULL

lub

**SHOW ENGINES \G**

```

Administrator: C:\Windows\System32\cmd.exe - startmysql.bat
mysql> show engines \G
***** 1. row *****
  Engine: InnoDB
  Support: DEFAULT
  Comment: Supports transactions, row-level locking, and foreign keys
Transactions: YES
    XA: YES
  Savepoints: YES
***** 2. row *****
  Engine: MRG_MYISAM
  Support: YES
  Comment: Collection of identical MyISAM tables
Transactions: NO
    XA: NO
  Savepoints: NO
***** 3. row *****
  Engine: MEMORY
  Support: YES
  Comment: Hash based, stored in memory, useful for temporary tables
Transactions: NO
    XA: NO
  Savepoints: NO
***** 4. row *****
  Engine: BLACKHOLE
  Support: YES
  Comment: /dev/null storage engine (anything you write to it disappears)
Transactions: NO
    XA: NO
  Savepoints: NO
***** 5. row *****
  Engine: MyISAM
  Support: YES
  Comment: MyISAM storage engine
Transactions: NO
    XA: NO
  Savepoints: NO
***** 6. row *****

```

## B. Reprezentacja tabeli na dysku

Za każdym razem gdy tworzymy tabelę, MySQL tworzy na dysku plik zawierający format tabeli (definicja tabeli). Format tabeli ma nazwę bazową taką samą jak nazwa tabeli oraz rozszerzenie *frm*. Plik ten jest niezmienny.

Poszczególne silniki baz danych mogą również tworzyć inne pliki unikalne dla tabeli i przeznaczone do przechowywania jej zawartości.

Silnik bazy danych	Pliki na dysku
InnoDB	ibd (dane i indeksy)
MyISAM	myd (dane), myi (indeksy)
CSV	csv (dane), csm (metadane)

## C. Silniki MySQL

### I. InnoDB:

#### Zalety

- obsługa transakcji
- gwarantuje integralność danych
- domyślny silnik od MySQL 5.5
- lepiej sprawuje się podczas replikacji typu master – slave
- automatyczna naprawa po awarii
- blokowanie na poziomie rekordów
- od wersji MySQL 5.6 obsługuje wyszukiwanie pełnego tekstu oraz indeksy FULLTEXT

#### Wady

- wolniejszy odczyt danych
- trudniejsze wykonywanie backupów

### II. MyISAM:

#### Zalety

- szybki odczyt z tabel
- prostsze wykonywanie kopii zapasowych
- odpowiedni dla tabel z małą ilością danych
- kompresja kluczy, gdy przechowywane są kolejne, podobne wartości ciągu tekstowego w indeksie
- wyszukiwanie pełnego tekstu oraz indeksy FULLTEXT
- więcej funkcji dla kolumn AUTO\_INCREMENT niż oferowane przez inne silniki baz danych
- przestrzenne typy danych i indeksy SPATIAL

#### Wady

- brak obsługi transakcji
- brak mechanizmów odpowiedzialnych za integralność danych
- przy dużych tabelach, długie czasy wykonywania **REPAIR TABLE** po awarii serwera

## D. Zmiana silnika baz danych

Zmiana zmiennej systemu baz danych:

**SET storage\_engine= Nazwa\_Silnika**

lub dla tabeli:

```
ALTER TABLE table_name ENGINE = Nazwa_Silnika
```

lub podczas tworzenia tabeli

```
CREATE TABLE Nazwa_Tabeli  
( ... )  
ENGINE = Nazwa_Silnika;
```

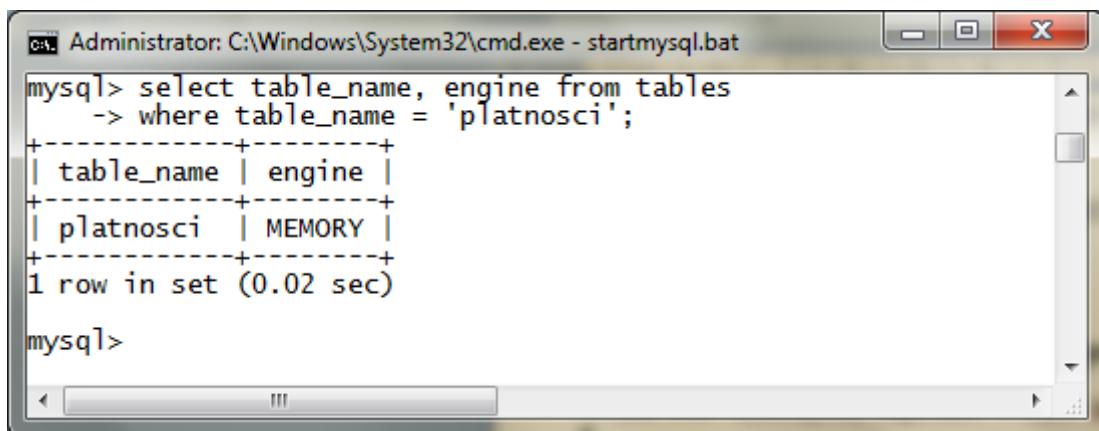
Przykład:

```
CREATE TABLE platnosci  
(  
    NumerKlienta int(11) not null,  
    Numer varchar(50) not null,  
    DataPlatnosci datetime not null,  
    kwota double not null,  
    PRIMARY KEY (NumerKlienta, Numer)  
)  
ENGINE = MEMORY;
```

#### E. Sprawdzenie czy dana tabela posiada określony silnik.

W tym celu wykonujemy zapytanie w bazie information\_schema;

```
select table_name, engine from tables  
where table_name = 'Nazwa_Tabeli';
```



The screenshot shows a Windows Command Prompt window titled "Administrator: C:\Windows\System32\cmd.exe - startmysql.bat". Inside, a MySQL session is running. The user has entered the following SQL query:

```
mysql> select table_name, engine from tables  
-> where table_name = 'platnosci';
```

The results of the query are displayed in a table:

table_name	engine
platnosci	MEMORY

Below the table, the message "1 row in set (0.02 sec)" is shown. The MySQL prompt "mysql>" appears at the bottom of the window.

lub wywołanie polecenia:

show table status\G

dla wybranej bazy

The screenshot shows a Windows Command Prompt window titled "Administrator: C:\Windows\System32\cmd.exe - startmysql.bat". The command entered is "mysql> show table status\G". The output displays two rows of table status information for tables "x" and "y". Both tables are InnoDB, version 10, with dynamic row formats, 0 rows, and 16384 data lengths. They were created on 2015-12-09 at 17:34:30 and have utf8\_general\_ci collation.

```
mysql> show table status\G
***** 1. row *****
      Name: x
      Engine: InnoDB
      Version: 10
    Row_format: Dynamic
      Rows: 0
Avg_row_length: 0
  Data_length: 16384
Max_data_length: 0
  Index_length: 0
  Data_free: 0
Auto_increment: NULL
Create_time: 2015-12-09 17:34:30
Update_time: NULL
Check_time: NULL
  Collation: utf8_general_ci
  Checksum: NULL
Create_options:
Comment:
***** 2. row *****
      Name: y
      Engine: InnoDB
      Version: 10
    Row_format: Dynamic
      Rows: 0
Avg_row_length: 0
  Data_length: 16384
Max_data_length: 0
  Index_length: 16384
  Data_free: 0
Auto_increment: NULL
Create_time: 2015-12-09 17:34:37
Update_time: NULL
Check_time: NULL
  Collation: utf8_general_ci
  Checksum: NULL
Create_options:
Comment:
2 rows in set (0.00 sec)

mysql>
```

## 11. Automatyzacja pracy a MySQL, czyli skrypty

Skrypt MySQL jest zwykłym plikiem tekstowym, który posiada rozszerzenie SQL.

Przykład: pliku skryptu

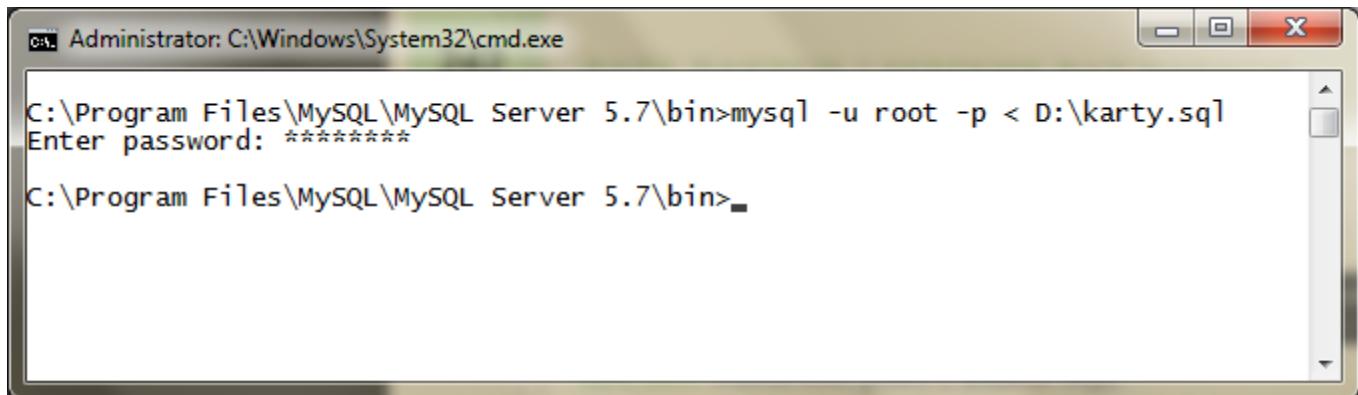
```
CREATE DATABASE IF NOT EXISTS Karty;
Use Karty
CREATE TABLE IF NOT EXISTS platnosci
(
  NumerKlienta int(11) not null,
  Numer varchar(50) not null,
  DataPlatnosci datetime not null,
```

```
kwota double not null,  
PRIMARY KEY (NumerKlienta, Numer)  
)  
ENGINE = MEMORY;
```

### A. Uruchamianie skryptów

I sposób: wykorzystanie polecenia mysql:

```
mysql -u Root -p < Scieżka_Nazw_Skryptu.sql
```



II sposób: wykorzystanie konsoli mysql

```
mysql> \. Scieżka_NazwaSkryptu.sql
```

Przykład:

```
mysql> \. D:\karty.sql
```

Lub

```
mysql> source Scieżka_NazwaSkryptu.sql
```

Przykład:

```
mysql> source D:\karty.sql
```

## **12. WSTAWIANIE, AKTUALIZACJA I USUWANIE DANYCH z BAZY MySQL**

### **A. Wstawianie danych do tabeli**

Wstawienie rekordów do tabeli można wykonać na cztery sposoby:

#### **I. Polecenie INSERT**

Składnia polecenia jest następująca:

```
INSERT INTO nazwa_tabeli (kolumny) VALUES (wartosci);
```

Uwaga: Kolumny w poleceniu insert można pominąć o ile wstawiamy dane w porządku określonym w strukturze tabeli.

#### **II. Polecenie INSERT i SELECT**

W tym przypadku dane do nowej tabeli pochodzą z innej tabeli.

Składnia polecenia jest następująca:

```
INSERT INTO Nazwa_Tabeli (Kolumna1, Kolumna2, Kolumna3)
```

```
SELECT Kolumna1, Kolumna2, Kolumna3 FROM Nazwa_TABELI_Danych;
```

#### **III. Załadowanie danych z pliku przy pomocy polecenia load data local infile**

Składnia polecenia:

```
load data local infile 'Ścieżka + Nazwa_Pliku'  
into table Nazwa_Tabeli  
fields terminated by 'znak'  
lines terminated by 'znak';
```

#### **IV. Załadowanie danych z pliku z pliku przy pomocy polecenia mysqlimport**

Składnia polecenia:

```
mysqlimport [options] Nazwa_Bazy Nazwy_Plikow
```

Najczęściej stosowane przełączniki options:

- --fields-terminated-by= - rozdzielenie pól
- --fields-enclosed-by= - zamknięcie pól
- --lines-terminated-by= - zakończenie linii
- -u - użytkownik
- -p - hasło

- -h - host

#### V. Przykłady dla polecenia INSERT:

W przykładzie będziemy wykorzystywać następującą tabelę:

```
CREATE TABLE Osoba2  
(  
ID INT,  
Imie VARCHAR(200),  
Nazwisko VARCHAR(200)  
);
```

```
C:\Windows\system32\cmd.exe - startmysql.bat

mysql> desc Osoba;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| ID    | int(11) | YES  |     | NULL    |       |
| Imie  | varchar(200)| YES |     | NULL    |       |
| Nazwisko | varchar(200) | YES |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.11 sec)

mysql> _
```

#### A) Wstawienie danych na wszystkie pola:

```
INSERT INTO Osoba (ID, Imie, Nazwisko) VALUES ('1', 'Jan', 'Kowalski');
```

**B) Wstawianie danych na wybrane pola:**

```
INSERT INTO Osoba (ID, Nazwisko) VALUES ('2', 'Nowicki');
```

C) Istnieje możliwość wstawienia jawniej wartości NULL, o ile pole na to pozwala

```
INSERT INTO Osoba (ID, Imie, Nazwisko)  
VALUES ('3', 'Tomasz', NULL);
```

D) Istnieje możliwość wstawiania wartości domyślnej

W tym celu zmodyfikujemy tabelę poleceniem:

```
ALTER TABLE Osoba  
ALTER COLUMN imie SET DEFAULT 'anonim';
```

A teraz dodajemy dane:

```
INSERT INTO Osoba (ID, Nazwisko) VALUES ('4', 'Kowalski');
```

**E) Istnieje możliwość wstawiania wielu wartości na raz**

```
INSERT INTO Osoba (ID, Imie, Nazwisko) VALUES  
('5', 'Michał', 'Jakubiak'),  
('6', 'Mikołaj', 'Roznerski');
```

Wykonując polecenie:

```
SELECT * FROM OSOBA;
```

Otrzymujemy:

```
C:\Windows\system32\cmd.exe - startmysql.bat  
mysql> select * from Osoba;  
+----+-----+-----+  
| ID | Imie   | Nazwisko |  
+----+-----+-----+  
| 1  | Jan    | Kowalski |  
| 2  | NULL   | Nowicki  |  
| 3  | Tomasz | NULL     |  
| 4  | anonym | Kowalski |  
| 5  | Michał | Jakubiak |  
| 6  | Mikołaj | Roznerski |  
+----+-----+-----+  
6 rows in set (0.00 sec)  
mysql> _
```

Co jest dowodem prawidłowego dodania wartości do tabeli.

## VI. Przykład dla polecenia mysqlimport

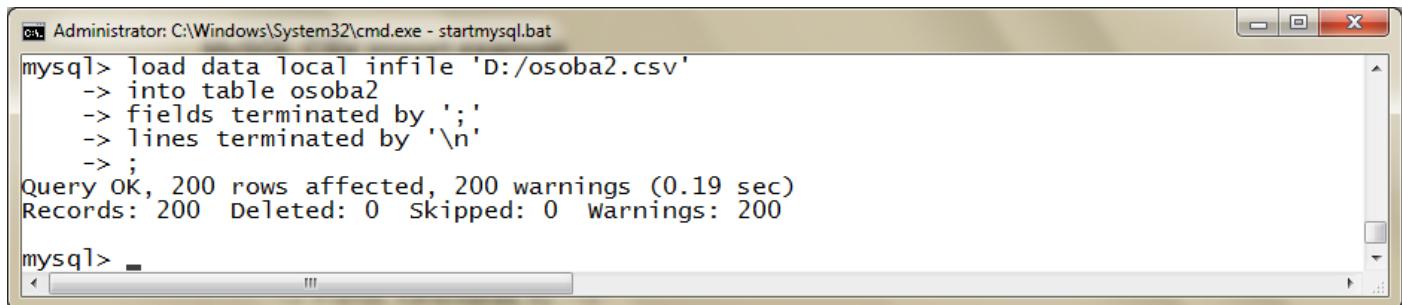
W przykładzie będziemy wykorzystywać następującą tabelę:

```
CREATE TABLE Osoba3  
(  
ID int,  
Imie VARCHAR(200),  
Nazwisko VARCHAR(200),  
Pesel VARCHAR(9)  
);
```

```
mysqlimport -u root -p --fields-terminated- by=;
```

```
--lines-terminated-by=\n mojaxy D:/osoba2.csv
```

## VII. Przykład dla polecenia load data local infule



```
Administrator: C:\Windows\System32\cmd.exe - startmysql.bat
mysql> load data local infile 'D:/osoba2.csv'
-> into table osoba2
-> fields terminated by ';'
-> lines terminated by '\n';
->
Query OK, 200 rows affected, 200 warnings (0.19 sec)
Records: 200 Deleted: 0 Skipped: 0 Warnings: 200
mysql>
```

### B. Aktualizacja danych w bazie

Do aktualizacji danych wykorzystujemy polecenie **update**.

Składnia polecenia:

```
UPDATE nazwa_tabeli
      SET kolumny do zmiany
          [WHERE które rekordy zmienić] ;
```

Przykłady:

#### I. Aktualizacja jednej kolumny:

```
UPDATE Produkt SET Cena = Cena * 1.20;
```

Powyższe polecenie aktualizuje kolumnę Cena w tabeli Produkt, podnosząc aktualną cenę o 20%

#### II. Aktualizacja wielu kolumn kolumny:

```
UPDATE Produkt SET StaraCena = Cena, Cena = Cena * 1.20;
```

Powyższe polecenie aktualizuje kolumnę Cena oraz StaraCena w tabeli Produkt, podnosząc aktualną cenę o 20%, a także zapamiętując cenę z przed podwyżki w kolumnie StaraCena.

#### III. Aktualizacja kolumn z warunkiem:

```
UPDATE Produkt
      SET Cena = Cena * 1,20
      WHERE Cena < 21.50;
```

Powyższe polecenie aktualizuje kolumnę Cena w tabeli Produkt, podnosząc aktualną cenę o 20% ale tylko produktów, których cena jest niższa od 21,50

### C. Usuwanie danych z bazy

Składnia polecenia:

```
DELETE FROM nazwatablei
[WHERE warunki] [ORDER BY ...] [LIMIT Liczba wierszy];
```

Przykłady:

#### I. Usuwanie pojedynczego rekordu

```
DELETE FROM Pracownicy WHERE ID = 4;
```

Powyższe polecenie usuwa z tabeli Pracownicy, dane które są identyfikowane przez pole ID = 4.

#### II. Usuwanie wielu rekordów

```
DELETE FROM Pracownicy WHERE ID > 5;
```

Powyższe polecenie usuwa z tabeli Pracownicy, dane który pola ID są większe od 5.

#### III. Usuwanie wielu rekordów z limitem ilości wierszy

```
DELETE FROM Pracownicy WHERE ID > 10 LIMIT 100;
```

Powyższe polecenie usuwa z tabeli Pracownicy, dane który pola ID są większe od 10, ale nie więcej niż 100 wierszy.

## 13. KOPIA ZAPASOWA w MySQL

### A. Wykonywanie kopii zapasowej

Składnia polecenia:

```
mysqldump -u user -p[haslo] nazwa_bazy > Nazwa_Pliku.sql
```

#### I. Możemy również wykonać kopię wszystkich baz:

```
mysqldump -u user -phaslo --all-databases > kopia_wszystkich_baz.sql
```

#### II. Możemy również wykonać kopię kilku baz jednocześnie:

```
mysqldump -u user -phaslo --databases baza1 baza2 > kopia_baz.sql
```

### III. Możemy również wykonać kopię wybranej tabeli bazy:

```
mysqldump -u user -phaslo baza tabela > kopia_baz.sql
```

Przykład:

The screenshot shows a Windows Command Prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The command entered is "C:\Program Files\MySQL\MySQL Server 5.7\bin>mysqldump -u root -p mojaxy osoba2 > D:/osoba2.sql". The password "mojaxy" is masked with asterisks. The command prompt then displays "Enter password: \*\*\*\*\*". Below the command, the path "C:\Program Files\MySQL\MySQL Server 5.7\bin>" is shown.

### B. Przywracanie kopii zapasowej

Składnia polecenia:

```
mysqldump -u user -p[haslo] nazwa_bazy < Nazwa_Pliku.sql
```

## 14. Dzienniki zdarzeń serwera MySQL

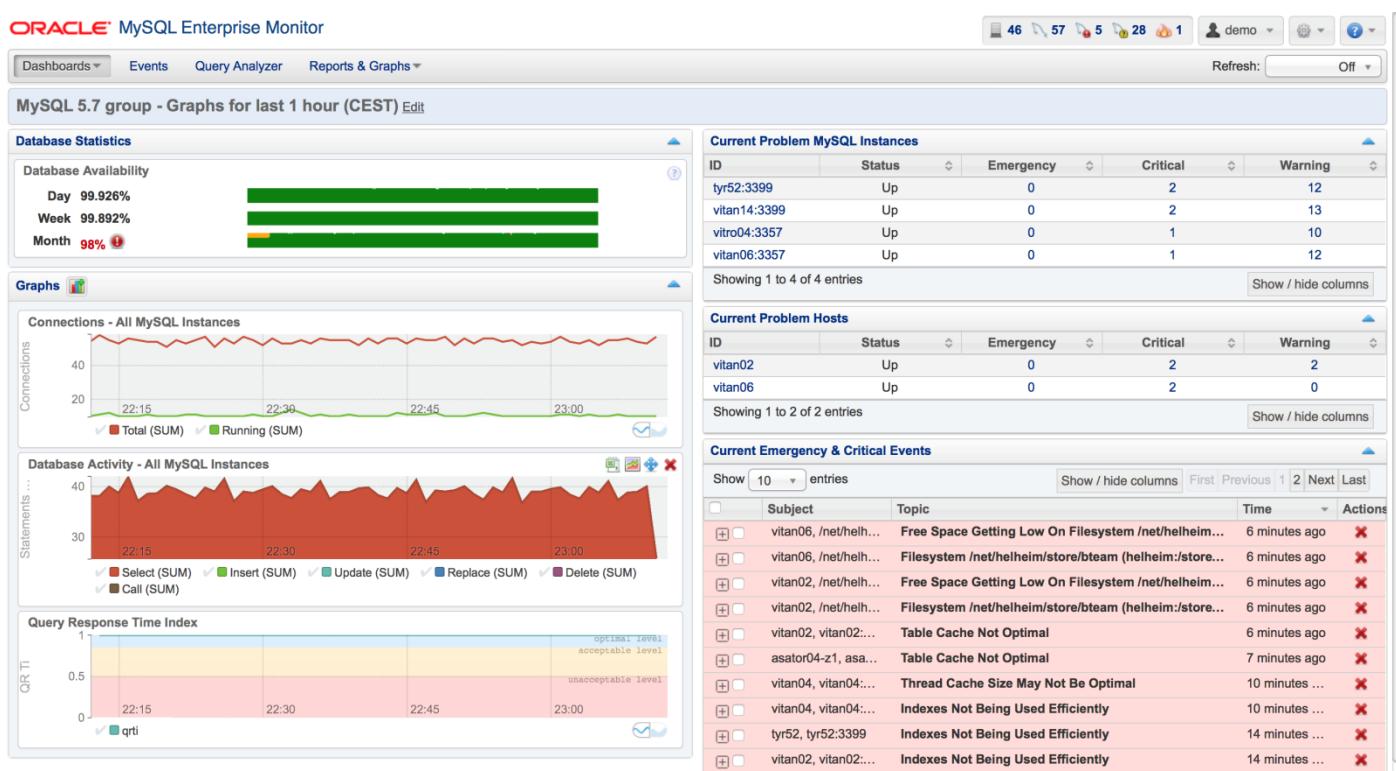
MySQL posiada kilka rodzajów logów:

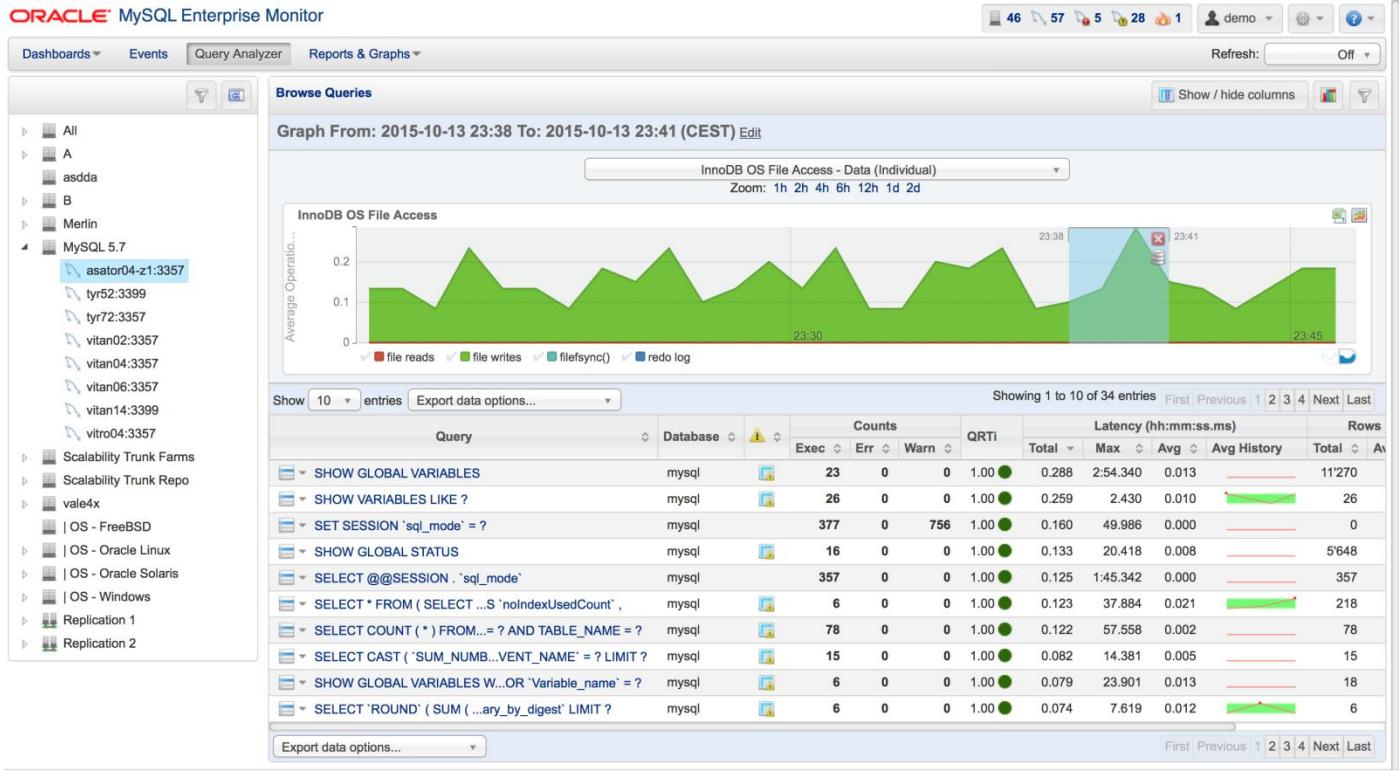
1. **error log** – log, do którego logowane są informacje na temat startu i zatrzymania się serwera MySQL, a także wszelkie informacje dotyczące krytycznych błędów, które wystąpiły w trakcie działania serwera. W szczególności logowane są tu informacje o tym, które tabele są uszkodzone i powinny zostać naprawione, a także zrzut stasu, przydatny do debugowania problemów z segfaultami. Jeśli włączona jest opcja automatycznej naprawy tabel MyISAM, to także w tym logu znajdą się informacje o przebiegu tego procesu. Plik ten nosi nazwę **HOSTNAME.err**. Do ustawiania tego dziennika służy zmienna `log_error` w pliku `my.ini`.
2. **general log** – log, który obejmuje informacje na temat bieżącej pracy MySQL. Logowane są w nim informacje o połączeniach – jaki użytkownik, z jakiego hosta, czy się poprawnie autoryzował. Trafiają tu także informacje o wszystkich zapytaniach jakie zostały wysłane do serwera MySQL. Ze względu na dużą ilość danych zapisywanych w tym logu, co negatywnie wpływa na wydajność serwera, domyślnie jest on wyłączony. Plik tego loga domyślnie nosi nazwę **HOSTNAME.log**. Do włączenia tego dziennika należy użyć zmiennej **general\_log**, a do ustawienia pliku użyć zmiennej **general\_log\_file**.
3. **binary log** – log ten to postać binarna wszelkich zmian, jakie zostały wprowadzone w bazach danego serwera MySQL na skutek wykonania zapytań INSERT, DELETE,, UPDATE. Nie są zapisywane zapytania SELECT. Dzięki binlogowi możliwe jest odtworzenie zawartości bazy w dowolnym punkcie czasu – wystarczy że posiadamy kopię zapasową danych np. na godzinę 00:00 a także binlogi od tego momentu do chwili obecnej. Binlog jest też podstawą działania replikacji MySQL. Domyślana nazwa bazowa pliku to **HOSTNAME-bin**. Do nazwy bazowej dodawane są kolejne numery, np. **HOSTNAME-bin.000001**. Do włączenia tego dziennika służy zmienna **log-bin**, do włączenia pliku indeksu binarnego dziennika zdarzeń służy zmienna **log-bin-index**. Można również określić maksymalną wielkość pliku w zmiennej **max\_binlog\_size**.

4. **relay log** – jest to binlog przegraný z serwera master na serwer slave, wykorzystywany przez slave do odtworzenia modyfikacji danych wykonanych na serwerze master – log ten występuje tylko gdy skonfigurowana jest replikacja. Nazywany często dziennikiem przekazywania. . Domyślona nazwa bazowa pliku to **HOSTNAME-relay-bin**. Do nazwy bazowej dodawane są kolejne numery, np. **HOSTNAME-relay-bin.000001**. Do włączenia tego dziennika służy zmienna **relay-log**, do włączenia pliku indeksu binarnego dziennika zdarzeń służy zmienna **relay-log-index**. Można również określić maksymalną wielkość pliku w zmiennej **max\_relay\_log\_size**.
5. **slow query log** – log do którego trafiają wszelkie zapytania, których czas wykonania przekroczył ustawiony w konfiguracji czas. Czas ten zdefiniowany jest w zmiennej **long\_query\_time**. Wolno wykonywane zapytania powodują zwiększenie zmiennej stanu **Slow\_queries**. Dane z tego logu ułatwiają wykonanie analizy obciążenia serwera MySQL. . Domyślona nazwa bazowa pliku to **HOSTNAME-slow**. Do włączenia tego dziennika służy zmienna **slow\_query\_log**, do włączenia pliku służy zmienna **slow\_query\_log\_file**. Do odczytu tego dziennika można posłużyć się programem **mysqldumpslow.pl** .

## 15. TESTY WYDAJNOŚCI w MySQL

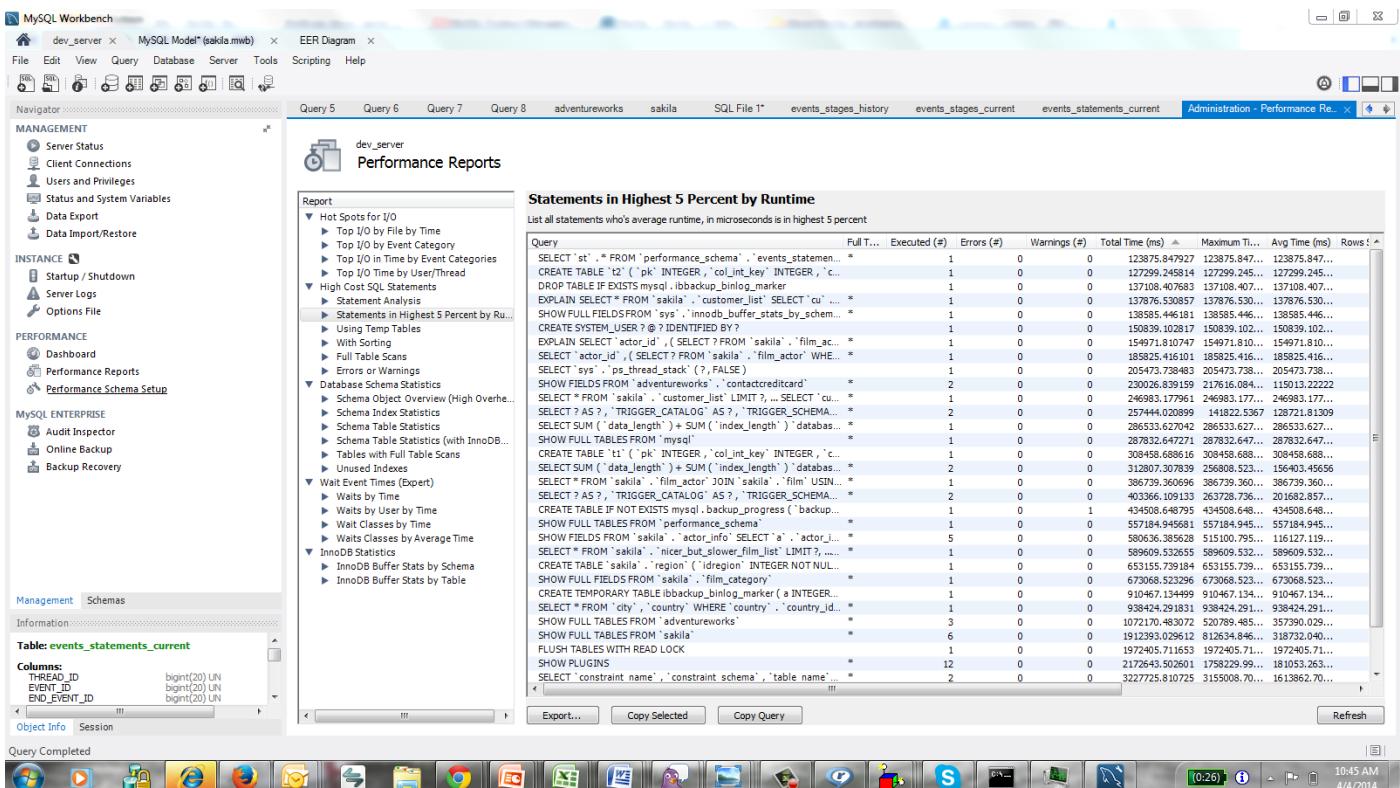
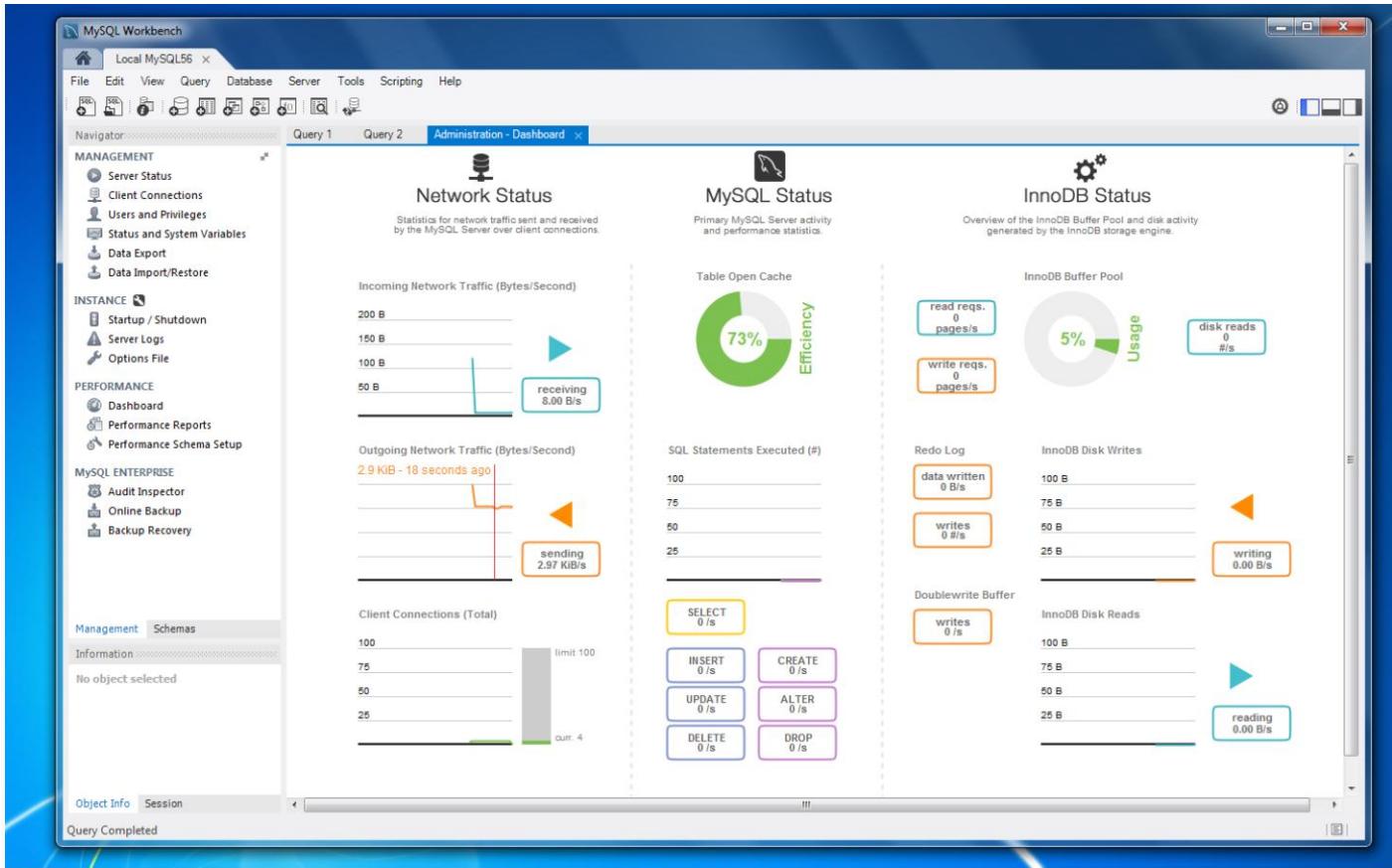
### A. MySQL Enterprise Monitor

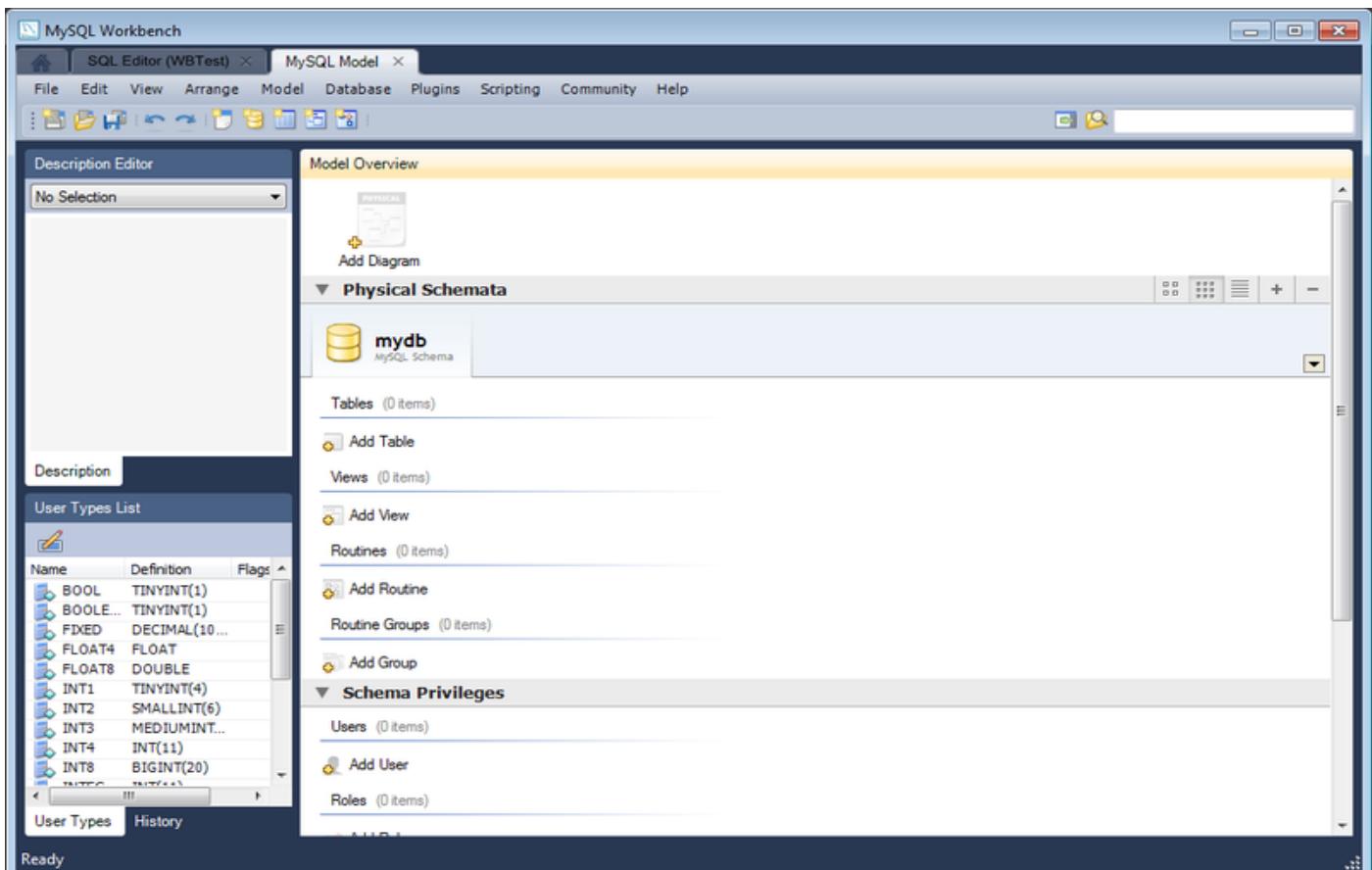
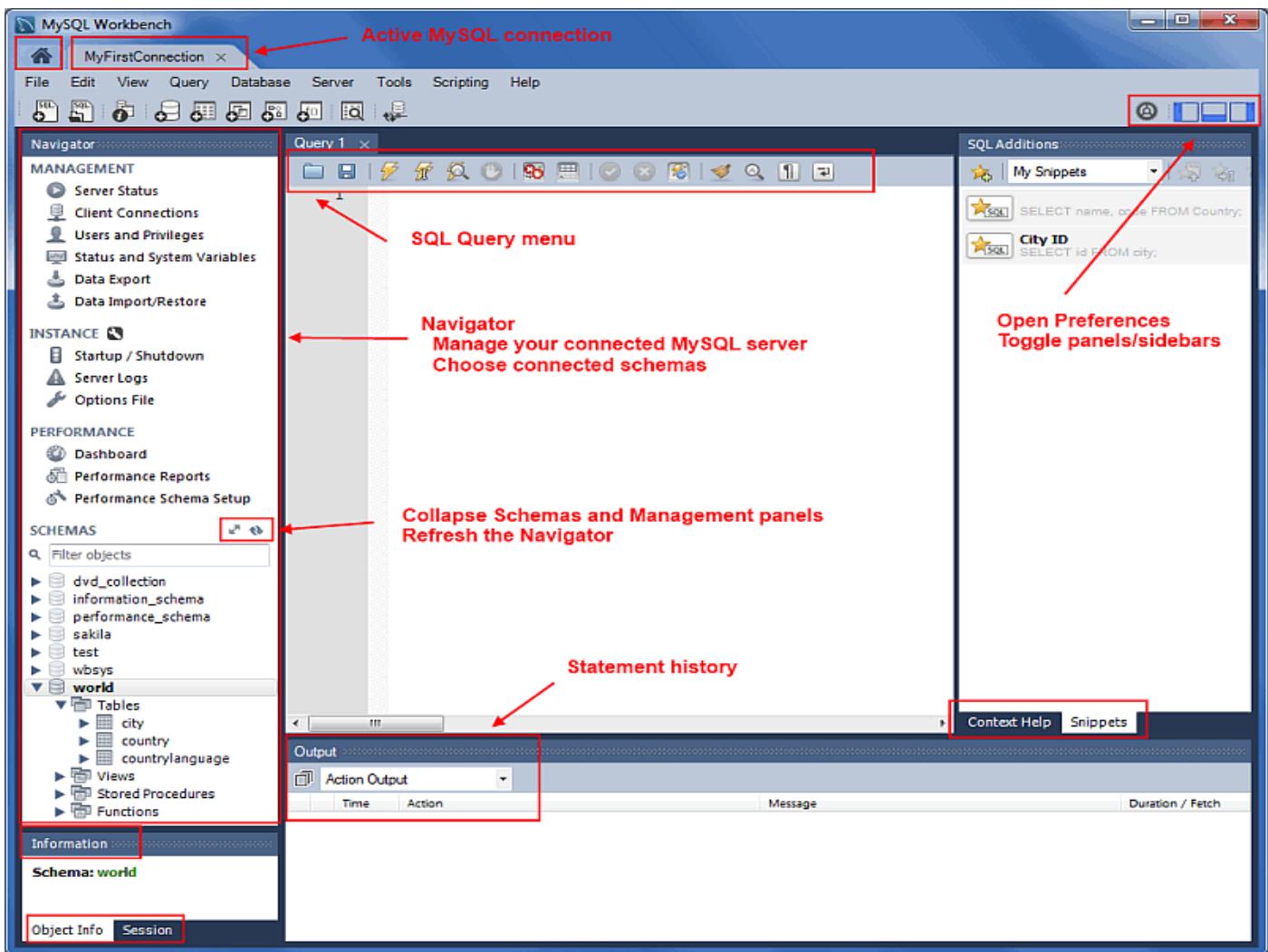




## B. MySQL Workbench

MySQL Workbench to narzędzie do zarządzania i modelowania baz danych MySQL. Za jego pomocą można edytować konfigurację serwera i jego komponentów, a także zaprojektować i stworzyć schematy (wizualne reprezentacje tabel, widoków itp.) nowych baz danych, wykonać dokumentację istniejących oraz zapewnić wsparcie przy procesach migracji do MySQL.





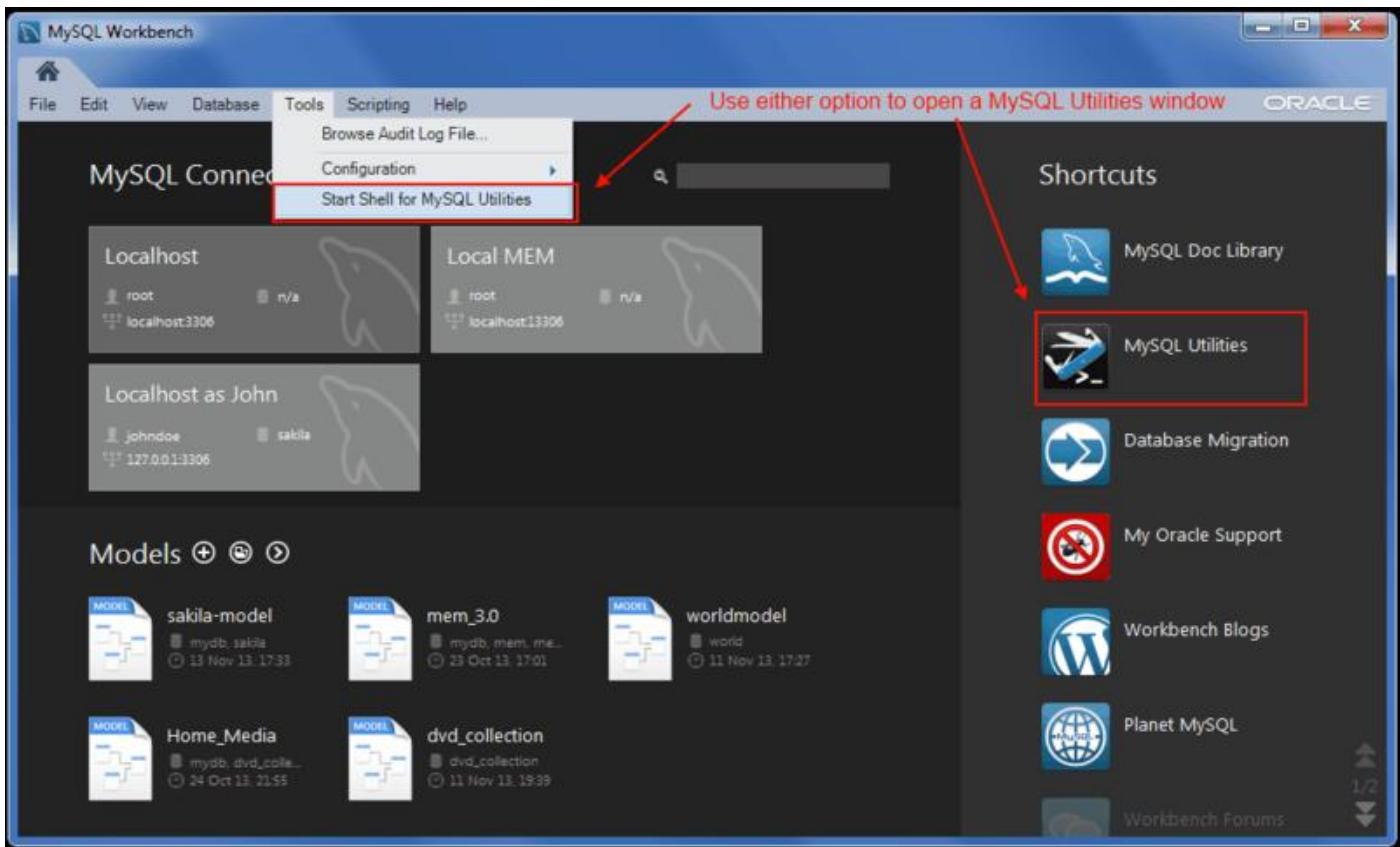
Jest to aplikacja dostępna w instalatorze bazy MySQL

## C. MySQL Utilities

Lista dostępnych narzędzi:

- **mysqlauditadmin** — Allows users to perform maintenance action on the audit log
- **mysqlauditgrep** — Allows users to search the current or an archived audit log
- **mysqldbcompare** — Compare Two Databases and Identify Differences
- **mysqldbcopy** — Copy Database Objects Between Servers
- **mysqldbexport** — Export Object Definitions or Data from a Database
- **mysqldbimport** — Import Object Definitions or Data into a Database
- **mysqldiff** — Identify Differences Among Database Objects
- **mysqldiskusage** — Show Database Disk Usage
- **mysqlfailover** — Automatic replication health monitoring and failover
- **mysqlfrm** — File reader for .frm files.
- **mysqlindexcheck** — Identify Potentially Redundant Table Indexes
- **mysqlmetagrep** — Search Database Object Definitions
- **mysqlprocgrep** — Search Server Process Lists
- **mysqlreplicate** — Set Up and Start Replication Between Two Servers
- **mysqlrplms** — Set Up and Start Replication Among a Slave and Multiple Masters
- **mysqlrpladmin** — Administration utility for MySQL replication
- **mysqlrplcheck** — Check Replication Prerequisites
- **mysqlrplshow** — Show Slaves for Master Server
- **mysqlrplsync** — Replication synchronization checker
- **mysqlserverclone** — Clone Existing Server to Create New Server
- **mysqlserverinfo** — Display Common Diagnostic Information from a Server
- **mysqluc** — Command line client for running MySQL Utilities
- **mysqluserclone** — Clone Existing User to Create New User

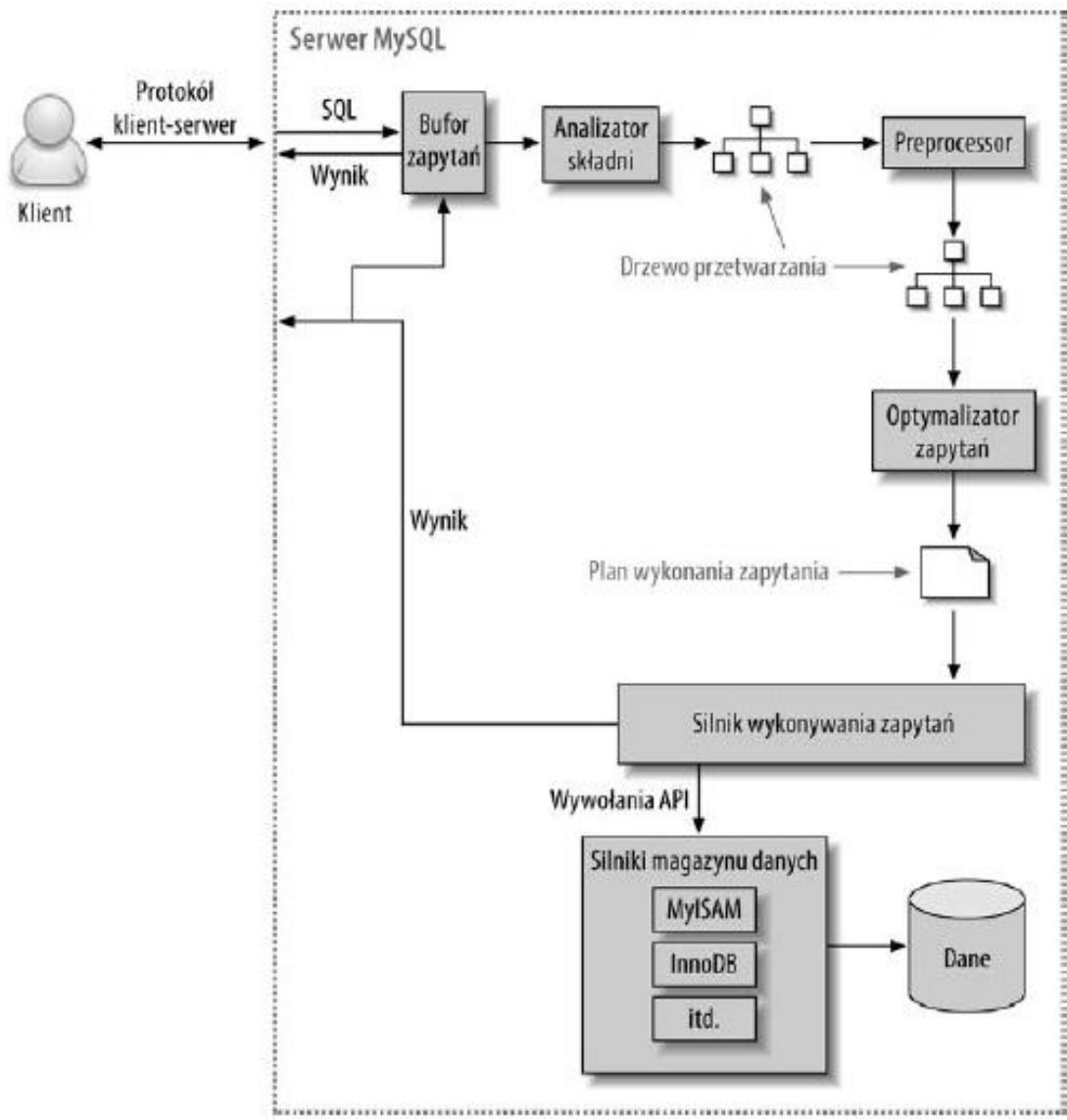
Dostęp do narzędzi jest z linii komend lub Workbencha



Szczegółowy opis każdego narzędzia można znaleźć na stronie: <https://dev.mysql.com/doc/mysql-utilities/1.5/en/utils-manuals.html>. To narzędzie mozna pobrać ze strony producenta.

## D. Explain

Zanim zaczniemy omawiać zastosowanie przełącznika explain, przedstawimy ogólny schemat wykonywania zapytań:



Tym razem interesuje nas bardziej polecenie: EXPLAIN SELECT... i nim się teraz zajmiemy. Ta wersja zapytania informuje nas o tym jak zachowa się serwer przy wykonywaniu danego zapytania. Zaczniemy od prostego zapytania:

```
mysql> EXPLAIN SELECT * FROM items;
*****1. row *****
id: 1
select_type: SIMPLE
table: items
type: ALL
possible_keys: NULL
key: NULL
key_len: NULL
ref: NULL
rows: 4
Extra: Using where
```

Co nam mówi ten rezultat? Ano, wiele.

1. **id** - identyfikator zapytania. Jest to numer sekwencyjny widać co w jakiej kolejności się wykonuje. Z działania wynika iż wyższy numer oznacza wcześniejsze wykonanie komendy.
2. **select\_type** - Informuje nas jakiego typu jest zapytanie.
  - o SIMPLE (proste zapytanie, brak unii i podzapytań),
  - o PRIMARY (główne zapytanie - w przypadku gdy mamy podzapytanie),
  - o UNION (gdy korzystamy z unii),
  - o DEPENDENT UNION (gdy podzapytanie jest unią zależną od głównego zapytania),
  - o UNION RESULT (wynik unii),
  - o SUBQUERY (podzapytanie),
  - o DEPENDENT SUBQUERY (podzapytanie zależne od głównego zapytania),
  - o DERIVED (pobieramy dane nie z tabeli ale z podzapytania)
3. **table** - nazwa tabeli na jakiej działa zapytanie.
4. **type** - informuje w jaki sposób wyszukiwane są wyniki.
  - o system: tabela ma tylko jedną kolumnę. Specjalny przypadek typu 'const';
  - o const: Tabela ma co najwyżej jeden pasujący wynik (np., gdy szukamy po unikalnym kluczu głównym);
  - o eq\_ref: Dla każdego wiersza z tabeli pierwszej (t1), wybrana kolumna (k1) jest porównywana z każdym wierszem tabeli drugiej po wskazanej kolumnie (k2).
  - o ref: wyświetlane wtedy gdy jest działanie na zakresach wyników lub gdy możliwy jest wynik większy niż 1 wiersz (klucz po którym odbywa się wyszukiwanie nie jest unikalny).
  - o ref\_or\_null: podobnie jak wyżej ale dodatkowo null też wchodzi w zbiór wyszukiwań.
  - o index\_mege: użyty został klucz wielokolumnowy.
  - o unique\_subquery: powiązany z podzapytaniem w którym używany jest klucz główny (primary key). Oznacza iż podzapytanie zwraca wynik do operatora 'IN'.
  - o index\_subquery: podobnie jak wyżej z tą różnicą iż klucz po którym następuje wyszukiwanie nie jest unikalny.
  - o range: działanie na zakresach wyników.
  - o index: podobnie jak zjawisko opisane poniżej, z tą różnicą iż wszystkie dane w tym wypadku pobierane są z klucza.
  - o ALL: występuje wtedy gdy w celu wyszukania wyniku należy odwiedzić wszystkie wiersze tabeli. Oczywiście najmniej optymalne i zazwyczaj tego wyniku będziecie się chcieli wystrzegać.
5. **possible\_keys**: indeksy które mogą zostać wykorzystane w celu znalezienia wyniku.
6. **key**: klucz, który został wybrany do wyszukania wyniku. Pragnę tutaj nadmienić iż nie zawsze MySQL wybiera najlepszą możliwość. Ta informacja pozwala nam jednak dowiedzieć się z jakich innych indeksów możemy próbować korzystać.
7. **key\_len**: długość klucza który został wybrany do wykorzystania w drodze wyszukiwania. Pozwala wywnioskować z ilu części klucza złożonego MySQL korzysta podczas tego zapytania.
8. **ref**: mówi nam które kolumny lub stałe posłużyły do wybrania wierszy z tabeli.
9. **rows**: ważna informacja, mówi ile minimalnie wierszy musi zostać przeszukane aby znaleźć wynik.
10. **extra**: dodatkowe informacje. Tutaj może się pojawić:
  - o distinct: informacja iż mysql pominie podobne wyniki;
  - o not exists: mysql mógł przeprowadzić operację left join, ale nie był zdolny wyciągnąć informacji z drugiej tabeli.
  - o range checked for each record (index map: #): mysql nie znalazł odpowiednich indeksów, możliwe jest jednak skorzystanie z innych indeksów. MySQL sprawdza zatem przy każdym wierszu czy może skorzystać z typu range lub index\_merge w celu zoptymalizowania wyszukiwania.
  - o using index: wyniki zostały wyszukane poprzez indeks, dane również zostały pobrane z indeksu. Nie było potrzeby wykonywać odczytu wszystkich informacji z wiersza.

- using temporary: w celu znalezienia wyniku utworzona została tabela tymczasowa.
  - using where: informuje iż składnia WHERE została użyta to ograniczenia danych wysyłanych klientowi.
  - Using sort\_union(...), Using union(...), Using intersect(...): informuje w jaki sposób wykorzystane zostały indeksy wiązane.
  - using index for group-by: informacja iż group by wykorzystał indeks. Niezwykle rzadki widok jednak niesamowicie piękny.

## E. mysqldumpslow.pl

Przykład użycia polecenia mysqldump.pl

```
mysqldumpslow.pl -a -s c -t 5 "C:\ProgramData\MySQL\MySQL Server  
5.7\Data\ZBIGNIEWACER-slow.log "
```

## F. MySQL Tuner

Program można pobrać ze strony: <https://mysqltuner.codeplex.com/>

Status	Notice
	MySQL Tuner 0.7 - Peter Chapman <peter@conglomerate.co.nz>
	Currently running supported MySQL version 5.7.9-log
	Switch to 64-bit OS - MySQL cannot currently use all of your RAM
	Archive Engine Installed
	Berkeley DB Engine Not Installed
	Federated Engine Not Installed
	InnoDB Engine Installed
	ISAM Engine Not Installed
	NDBCLUSTER Engine Not Installed
	Data in InnoDB tables: 432K (Tables: 27)
	Total fragmented tables: 0
	MySql.Data.MySqlClient.MySqlException (0x80004005): Unknown error 1054 w MySql.Data.MySqlClient.MySqlStream.Re...

## G. Polecenie mytop

To polecenie mytop dostępne jest tylko w systemie Linux, a znaczenie jego jest podobne do polecenia top tylko w odniesieniu do bazy MySQL

<b>Id</b>	<b>User</b>	<b>Host/IP</b>	<b>DB</b>	<b>Time</b>	<b>Cmd</b>	<b>Query or State</b>
1279446	root	localhost	test	0	Query	show full process
1279043	salmar	localhost	prodigmcrm	15	Sleep	
1260488	salmar	localhost:58157	openfire	17	Sleep	
1279435	salmar	localhost:60274	syscp	24	Sleep	
1279436	salmar	localhost:60275	syscp	24	Sleep	
1260489	salmar	localhost:58158	openfire	27	Sleep	
1260490	salmar	localhost:58159	openfire	27	Sleep	
1260491	salmar	localhost:58160	openfire	27	Sleep	
1260492	salmar	localhost:58161	openfire	27	Sleep	
1278668	salmar	localhost	prodigmcrm	96	Sleep	
1279377	salmar	localhost	prodigmcrm	136	Sleep	
1279344	salmar	localhost	prodigmcrm	176	Sleep	
1279060	salmar	localhost	prodigmcrm	338	Sleep	
1279255	salmar	localhost	prodigmcrm	378	Sleep	
1279173	salmar	localhost	prodigmcrm	499	Sleep	
1278992	salmar	localhost	prodigmcrm	660	Sleep	

## H. Indeksy

### 16. PERSPEKTYWY

W SQL tabela, która jest utworzono za pomocą zapytania CREATE TABLE, nazywa się tabelą *podstawową (base table)*. Jej struktura i dane przechowywane są przez system operacyjny.

Wynik każdego zapytania – SELECT też, formalnie, jest tabelą. Tabelę taką – nazywa się tabelą *pochodną (derived table)*.

**Widokiem (lub perspektywą)** nazywam trwałą definicję tabeli pochodnej, która to definicja przechowywana jest w bazie danych.

#### A. Tworzenie widoków

Składnia polecenia:

```
CREATE VIEW nazwa
AS
SELECT treść zapytania select;
```

Przykłady:

A.

```
CREATE VIEW Widok1
AS
SELECT imie, nazwisko, pesel
FROM OSOBA
WHERE wiek > 30;
```

B.

```
CREATE VIEW lewy
AS
SELECT arabic.i AS i, b, r
FROM arabic LEFT JOIN roman ON arabic.i=roman.i;
```

## B. UWAGI:

Widoki reagują na zmianę danych w tabelach, których użyto do ich stworzenia.

Widoki nie reagują na zmianę definicji tabeli, o ile nie narusza ona integralności widoku.

Widoki zapisywane są w widoku tabeli.

Jeżeli usuniemy tabelę, nie usuniemy zaś opartych na niej widoków, lub też tak zmodyfikujemy tabelę, że definicja widoku straci sens, dostaniemy widoki „*dysydujące*” (*dangling views*)

## C. Widoki niemodyfikowalne

Niektóre widoki są z założenia niemodyfikowalne, są te, które w liście SELECT zawierają UNION, UNION ALL, DISTINCT, DISTINCTROW, inny widok niemodyfikowalny lub podzapytanie.

*W MySQL widoki w ogóle nie mogą zawierać podzapytań.*

## D. SPRAWDZENIE PORAWNOŚCI WIDOKU

Składnia polecenia:

```
CHECK TABLE nazwawidoku
```

## E. Usuwanie widoków

Składnia polecenia:

```
DROP VIEW NazwaWidoku
```

## **17. INDEKSY**

### **A. Rodzaje indeksów**

#### **• UNIQUE**

Po zdefiniowaniu unikalnego indeksu nie będzie można wprowadzić do kolumny powtarzającej się wartości, jak już wiemy indeksów nie opłaca się stosować tam gdzie często wartości powtarzają się, w tym właśnie celu można utworzyć indeks unikalny który sam zadba o to, a w przypadku próby wprowadzenia wartości która już jest wyrzuci błąd.

#### **• PRIMARY KEY**

Klucz Podstawowy (PRIMARY KEY) nazywany też kluczem głównym, dzięki temu ograniczeniu będziemy mogli w łatwy sposób odwoływać się do rekordów, służy on więc do identyfikacji poszczególnych wierszy. W jednej tabeli można utworzyć najwyżej jeden klucz podstawowy. Dla pola z kluczem głównym można dodatkowo określić parametr AUTO\_INCREMENT. Na ten klucz automatycznie nakładana jest właściwość UNIQUE INDEX dzięki czemu do kolumny nie można wprowadzić powtarzającej się wartości. • Klucz główny nie jest tym samym co unikalny indeks, przybiera on dodatkowe właściwości m.in można na niego nałożyć wspomniane AUTO\_INCREMENT, a także odwoływać się przez klucz obcy

#### **• FULLTEXT**

FULLTEXT - Wsparcie MySQL do wyszukiwania pełnotekstowego. Zakładamy dla pól, gdzie chcemy wyszukiwać dane za pomocą składni MATCH. Silnik bazy MyISAM posiada możliwość założenia indeksu pełnotekstowego (fulltext), jednak niestety najpopularniejszy silnik InnoDB nie posiada wsparcia dla wyszukiwania pełnotekstowego .

#### **• INDEX**

Indeks jest specjalnym plikiem utworzonym na serwerze, dzięki któremu możemy poprawić wydajność przy pobieraniu danych w większych tabelach. W indeksie zawarte są uporządkowane informacje gdzie znajduje się dana wartość dzięki czemu MySQL zamiast przeszukiwać wszystkie rekordy w poszukiwaniu tego konkretnego pola zatrzyma się na indeksie, gdzie znajdzie się dana wartość co przyspieszy wyszukiwanie. Nałożenie indeksów należy dokładnie przemyśleć, w innym wypadku przyniosą odwrotny rezultat od zamierzonego, tym bardziej nie należy ich zakładać na każdą kolumnę. Indeksy mogą być nałożone na każdy typ kolumny, co najwyżej dla 16 kolumn w tabeli.

### **B. Widoki wykorzystywane są, aby:**

- Ułatwić odczytywanie danych użytkownikom. Jeżeli widok pobiera dane na przykład z siedmiu tabel bazowych, odczytanie danych poprzez widok będzie wymagało wykonania prostej instrukcji **SELECT**.
- Ułatwić odczytywanie użytkownikom danych obliczonych na podstawie informacji zapisanych w tabelach. Na przykład odczytanie widoku obliczającego średnią liczbę sprzedanych towarów i grupującego wyniki według nazw kategorii produktów będzie wymagało wykonania prostej instrukcji **SELECT**.

- Ograniczyć, ze względów bezpieczeństwa, dostęp użytkowników do poufnych danych. Jeżeli widok pobiera dane na przykład jedynie z wybranych kolumn tabeli, przez ten widok możliwe będzie odczytanie jedynie wybranych danych (np. imienia i nazwiska, ale nie pensji pracowników).
- Ukryć przed użytkownikami strukturę tabel bazy danych. Informacja ta nie jest niezbędna użytkownikom do pracy z bazą, a przez niektóre osoby może zostać wykorzystana w niewłaściwy sposób.
- Ułatwić zarządzanie uprawnieniami użytkowników.
- Oddzielić i uniezależnić aplikację kliencką od zmian tabel bazy danych.

### **C. Gdzie stosować:**

- Gdy często w zapytaniach w klauzuli WHERE korzystamy z danego pola lub zestawu pól
- Gdy pola często wykorzystujemy do łączenia tabel poprzez JOIN-y
- Gdy pola są często wykorzystywane do sortowania wyników tabeli
- Gdy pola są często wykorzystywane w funkcjach MIN() i MAX()
- Gdy rekordy w tabeli w danym polu mają bardzo różne wartości – wyszukiwanie przy pomocy indeksu jest tym skuteczniejsze, im mniej rekordów ma szukaną przez nas wartość (a więc gdy np. 5 spośród miliona rekordów mają szukaną przez nas wartość to indeks będzie korzystny, podczas gdy tę wartość będzie miała np. połowa to indeks tylko zajmuje niepotrzebnie miejsce, zapytania nam nie przyspieszy)

### **D. Gdzie nie należy stosować:**

- przede wszystkim nie należy ich stosować tam gdzie popadnie, indeks jest plikiem fizycznym co za tym idzie zajmuje on też miejsce na dysku
- indeksy przy poleceniach modyfikujących (INSERT, REPLACE, DELETE, UPDATE) muszą być także aktualizowane co za tym idzie przy tego typu poleceniach indeksy zamiast przyśpieszyć przynoszą odwrotny rezultat czyli zwolnią modyfikacje, w związku z czym nie należy ich stosować tam gdzie często jest coś modyfikowane, a rzadko pobierane
- indeksów nie opłaca się stosować tam gdzie wartość często się powtarza, tak więc nie opłaca się ich definiować tam gdzie do pola wpisywane są tylko wartości typu 0, 1 lub Kobieta, Mężczyzna etc.
- Nie powinno się zakładać indeksu wielokrotnie na tę samą kolumnę

## **E. TWORZENIE INDEKSÓW**

Składnia polecenia:

```
CREATE [UNIQUE|FULLTEXT|SPATIAL] INDEX nazwa_indeksu
  USING [BTREE | HASH | RTREE]
  ON nazwa_tabeli (nazwa_kolumny [(długość)] [ASC | DESC],...)
```

Najpierw trzeba określić indeks na podstawie typów lub silnika bazy danych:

- UNIQUE oznacza dla MySQL, że ma utworzyć ograniczenie, że wszystkie wartości w indeksie muszą być różne.
- FULLTEXT indeks jest obsługiwany tylko przez silnik MyISAM i jest akceptowany tylko dla kolumn, które mają typ danych CHAR, VARCHAR lub TEXT
- SPATITAL to indeks przestrzenny dla kolumny i dostępny w silniku MyISAM. Ponadto wartość kolumny nie może być NULL.

Przykład:

```
CREATE INDEX Stanowisko  
ON pracownicy(Stanowisko)
```

## F. USUWANIE INDEKSÓW

Składnia polecenia:

```
DROP INDEX nazwa_indeksu  
ON nazwa_tabeli
```

Przykład:

Na przykład, jeśli chcesz usunąć indeksu **Stanowisko**, który dodaliśmy do tabeli **pracownicy** , po prostu wykonaj następujące zapytanie:

```
DROP INDEX Stanowisko ON pracownicy
```

## 18. REPLIKACJA

