# *MySQL*

## *Essential Training*

***Shavit Ilan – 31/1/2010***

# תוכן העניינים

# *MYSQL - Essential Training*

## *Chapter 3: Configuring MySQL*

### *0301 Using MySQL Command Line Interface on Windows*

- Mysql CLI contain all Mysql feature (GUI tools don't include all of them)
- Create shortcut to 'cmd.exe' and change 'start in' (property field) to 'd:\SomePlace'
- Change the layout of the cmd.exe command (on property field):
  Windows size and screen buffer equal to width=130, height=60

### *0303 Setting up the root user*

- root user is a powerful user and it comes by default with empty password
- MySQL comes with two root users and we need to change their passwords:
  - *UPDATE mysql.user SET Password = PASSWORD('root1234') WHERE User = 'root'; PASSWORD is a function that encrypt the password*
  - *FLUSH PRIVILEGES; Tell mysql to re-read users and passwords tables*
- exit and re-enter to mysql with this command:
  - *mysql -u root -p*

### *0304 Setting up a regular user*

- Mysql user management is very complex and allows us to configure any user with it's own privileges. Give any user the exact privileges that he need (and not more)!
  - *CREATE USER web@localhost;*
    *This will create user for web application purpose. Access only from localhost*
  - *CREATE USER admin@localhost;*
    *This will create user for admin tasks. Access only from localhost*
  - *GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, FILE, INDEX, ALTER, CREATE TEMPORARY TABLES, CREATE VIEW, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, EXECUTE ON *.* TO web@localhost;*
    *This will add privileges to web@localhost*
  - *GRANT ALL on *.* TO admin@localhost WITH GRANT OPTION;*
    *This command allows the user to GRANT PRIVILEGES to other users.*

  - *FLUSH PRIVILEGES; Tell mysql to re-read the users and passwords tables*
  - *UPDATE mysql.user SET Password = PASSWORD('web1234') WHERE User = 'web';*
  - *UPDATE mysql.user SET Password = PASSWORD('admin1234') WHERE User = 'admin';*
  - *FLUSH PRIVILEGES;*

## *0305 Importing databases on Windows*

- *mysql -u admin -p < album-mysql.sql*
  *This command will import album-mysql database to mysql*
- *mysql -u admin -p*
  - *USE album;*
  - *SHOW TABLES;*
  - *SELECT * FROM album;*
  - *SELECT COUNT(*) FROM album;*

## *0501 Creating a database*

  - *CREATE DATABASE sales;*
  - *SHOW DATABASES;*
  - *DROP DATABASE sales;*

## *0502 Creating a table*

  - *CREATE TABLE test_customer (*
    *id  INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,*
    *name VARCHAR(255),*
    *address VARCHAR(255),*
    *city VARCHAR(255),*
    *state CHAR(2),*
    *zip CHAR(10)*
    *);*
  - *SHOW TABLES;*
  - *DESCRIBE test_customer;*
  - *DROP TABLE test_customer;*

# *Chapter 6: MySQL Data Types*

## *0601 What are data types*

Mysql data types:

- Numeric – for numbers
- String – for words and text
- Large storage – for files and documents
- Dates and Times
- Bit values – for flags or logical values
- Enumeration – for mnemonic values

## *0602 Numeric types (integer, Floating points)*

- Integer – for whole numbers
  - TINYINT -128 - +127 (or 0-255)
  - SMALLINT -32,768 - |32,767 (or 0-65536)
  - MEDIUMINT -8,388,608 - 8,388,607 (or 0-16,777,215)
  - INT -2,147,483,648 - -2,147,483,647 (or 0-4,294,967,295)
  - BIGINT - -9,223,372,036,854,775,808 9,223,372,036,854,775,807 (or 0 18,446,744,073,709,551,615)
- Floating points – for real numbers
- Fixed point – numbers with fixed precision (10.02 ILS)
  SELECT 1+2, 2+5, 2=2.0 ==> Will return 3,7,1

## *0603 String types (fix, variable)*

- Fixed – Always the same size
  - CHAR(LENGTH) is the fixed type
  - BINARY(LENGTH) – for fixed binary data
- Variable – Different values and size
  - VARCHAR(LENGTH) is the variable type
  - VARBINARY(LENGTH) – for variable binary data

## *0604 Large storage types*

- BLOB (store large **binary** object like pictures):
  - TINYBLOB – up to 256 bytes
  - BLOB - up to 64K bytes
  - MEDIUMBLOB - up to 16M bytes
  - LONGBLOB - up to 4G bytes
- TEXT (for storing **text** data like article, web pages etc...):
  - TINYTEXT – up to 256 bytes
  - TEXT - up to 64K bytes
  - MEDIUMTEXT - up to 16M bytes
  - LONGTEXT - up to 4G bytes

## *0605 Date and time types*

- DATE – range 1000-9999
- TIME
- DATETIME – date and time combination
- TIMESTAMP – for event logging
  Example (in 'test' database):
  - *CREATE TABLE datetest (date DATETIME, STAMP timestamp);*
  - *INSERT INTO datetest (date) VALUES ('2009-05-04 15:31:32');*
    *Notice that the timestamp field will insert in automatic way*
  - *SELECT date, stamp, DATEDIFF(date, stamp) FROM datetest;*
    *DATEDIFF return the different between two dates (in days)*

## *0606 Bit type*

Bit types use as a flag field. Generally used by programmers.

- *CREATE TABLE bittest (b1 BIT(8), b2 BIT(10));*
- *INSERT INTO bittest (b1,b2) VALUES (b'11110000', b'01001');*
- *SELECT * FROM bittest; This command will show GIBRISH values.*
  *We can **show binary values** like that:*
- *SELECT b1+0, b2+0 FROM bittest; **OR:***
- *SELECT BIN(b1), BIN(b2) FROM bittest; **OR:***
- *SELECT OCT(b1), HEX(b2) FROM bittest; **OCT** for octal base and **HEX** for hex base*

## *0607 Boolean values*

- No Boolean types in Mysql
- BOOL is an alias to TINYINT
- '1' is represented by 1 (one) and it means **TRUE**
- '0' is represented by 0 (zero) and it means **FALSE**
- BIT type is excellent for storing Boolean values
    - *SELECT 5=5;  ====> 1*
    - *SELECT 5>6;  ====> 0*

## *0608 Enumeration types*

- There are two types of enumeration: **ENUM** and **SET** and it works differently
- **ENUM**: is stored as an integer so in this example red=1, blue=2, green=3
    - *CREATE TABLE enumtest (color ENUM('red','blue','green'));*
    - *INSERT INTO enumtest(color) VALUES('red');*
    - *SELECT * FROM enumtest;*
    - *INSERT INTO enumtest(color) VALUES('orange'); Will insert blank value (not Null) and will not return any error to the user!!!*
- **SET**: is stored as an integer so in this example red=1, blue=2, green=3
    - *CREATE TABLE settest (attrib SET('red','blue','green'));*
      *any combination is allow: 'red','red,blue','green,blue'. Mysql store SET type as a BIT field*
    - *INSERT INTO settest(attrib) VALUES('red,green');*
      *Mysql will show the values in the same order that they had been created (and not stored!!!)*

# *Chapter 7: MySQL Functions*

## *0701 MySQL functions*

- In mysql functions used to derived values from other values\data:
  - *SELECT COUNT(*) FROM album;*
    *COUNT is a function, * is the argument*

## *0702 String functions*

- *SELECT 'Hello World';*
  *The result is a table with title 'Hello World' and result 'Hello World'*
- *SELECT CONCAT ('Hello', 'World', '!!!');*
  *The result is: 'HelloWorld!!!'*
- *SELECT CONCAT_WS (':', 'Hello', 'World', '!!!');*
  *WS means with separator, ':' is the separator. The result: Hello:World:!!!*
- *SELECT LPAD(title, 30, ' ') FROM album;*
  *will left pad with up to 30 space characters (30 is the max field length)*
- *SELECT RPAD(title, 30, ' ') FROM album;*
  *same as LPAD but pad from the right*
- *SELECT SEC_TO_TIME( 320 )*
  *Convert 320 (320 is an integer) to 00:05:20 (5 minute and 20 seconds)*
- *SELECT CONCAT_WS (':', 301 DIV 60, LPAD(301 MOD 60,2,'0'));*
  *The result 05:01 Remark: Use real DB field instead of 301...*

## *0703 Numeric functions*

- *SELECT 320/60; ==> 5.333*
- *SELECT 320 DIV 60; ==> 5*
- *SELECT 320 MOD 60; ==> 20*
- *SELECT CONV(320,10,16); ==> Convert 320 from base 10 to base 16*
- *SELECT CONV(0101,2,10); ==> 5*
- *SELECT HEX(10); ==> A*
- *SELECT BIN(10); ==> 1010*
- *SELECT OCT(10); ==> 12*
- *SELECT CRC32('Hello World'); ==> 1243066710 CRC32 is a hash function*
- *SELECT HEX(CRC32('Hello World')); ==> 4A17B156*
- *SELECT FORMAT(1000000000,2); ==> 1,000,000,000.00*
- *SELECT POW(16,2); ==> 256 (16*16)*
- *SELECT RAND(); ==> get random numbers less than 1 (example: 0.233423434)*

## *0704 Date and time functions*

- *SELECT NOW() ==> 2009-11-28 13:58:24*
- *SELECT CURDATE() ==> 2009-11-28*
- *SELECT CURTIME() ==> 13:58:24*
- *SELECT UTC_TIMESTAMP() ==> Date and Time in UTC: 2009-11-28 12:02:28*
- *SELECT NOW() - UTC_TIMESTAMP() ==> 20000.000000 If we want the real time we do:*
- *SELECT TIME(NOW() - UTC_TIMESTAMP()) ==> 02:00:00*
- *SELECT DATEDIFF(NOW(),'2009-03-21') ==> Days between two dates*
- *SELECT DATE_FORMAT (NOW(), '%W, %D %M %Y, %T')*
  *==> Saturday, 28th November 2009, 14:14:53*
  *DATE_FORMAT enable the user to format it's Date and Time result:*
  *%W – Day of the week. Example: Saturday*
  *%T – Time: 14:14:53*
  *%D ==> 28th*
  *%d ==> 28*
- *SELECT DATE_FORMAT (NOW(), '%d/%M/%Y, %T')*
  *==> 28/November/2009, 14:22:15*

## *0705 Time zones in MySQL*

- *Mysql can handle several time zones (each application may use different time zone)*
- *SELECT @@time_zone ==> SYSTEM*
  *@@ means that this is a SYSTEM (computer) variable*
  *Remark:*
  *MYSQL store every record in UTC, but in order to see it in correct time it use the time_zone variable*
- *We can change Mysql time zone:*
  *SET time_zone = 'US /Eastern';*
  *If we will get an error: #1298 - Unknown or incorrect time zone: 'US /Eastern'*
  *This means that we don't have the time zone in the mysql installation*
- *Chapter '0706 Installing time zone support in MySQL' explain how to install the TZ support in MySQL*
- *SET time_zone = 'SYSTEM';*
  *Will return the TZ to default*

## *0708 Aggregate functions*

- *SELECT COUNT(*) FROM Country*
  *COUNT is an aggregate function ==>239*
- *SELECT COUNT(DISTINCT region) FROM Country*
  *Count only the distinct values in region field ==> 25*
- *SELECT region, COUNT(region) FROM Country GROUP BY region;*
  *Show all regions and tell how many rows (country) in each region.*
  *This function group all regions area and count the number of raws in each group*

| region | COUNT(region) |
|---|---|
| Antarctica | 5 |
| Australia and New Zealand | 5 |
| Baltic Countries | 3 |
| British Islands | 2 |

- *SELECT region, GROUP_CONCAT(name) FROM Country GROUP BY region;*
  *This will show all countries in each region (separate by comma). This is the detail of the previous*

*command.*
*region GROUP_CONCAT(name)*
*Antarctica French Southern territories,Heard Island and McDonald Islands*
*Australia and New Zealand Australia,New Zealand,Norfolk Island*
*Baltic Countries Latvia,Estonia,Lithuania*

- *SELECT region, GROUP_CONCAT(name ORDER BY name SEPARATOR ' / ') FROM Country GROUP BY region;*
  *This will show all countries in each region (first ordered and than separated by comma).*
  *region GROUP_CONCAT(name)*
  *Antarctica French Southern territories / Heard Island and McDonald Islands*
  *Australia and New Zealand Australia / New Zealand / Norfolk Island*
  *Baltic Countries Estonia \ Latvia \ Lithuania*

### *0709 Full-text search*

- *SELECT * FROM table_name WHERE MATCH (field1, field2) AGAINST ('some_string');*
  *Will show all table column that field1 or field2 have the value 'some_string'*
- *SELECT * FROM table_name WHERE MATCH (field1, field2) AGAINST ('string1 string2');*
  *Will show all table column that field1 or field2 have the value string1 or string2*
  *Remark:*
  *This search called: 'natural language search': it will ignore strings that exist in more than 50% of the rows.*
- *SELECT * FROM table_name WHERE MATCH (field1, field2) AGAINST ('+string1 +string2' IN BOOLEAN MODE);*
  *Will show all table column that field1 or field2 have the value string1 AND string2 (refers to +string...)*
- *SELECT * FROM table_name WHERE MATCH (field1, field2) AGAINST ('+string1 -string2' IN BOOLEAN MODE);*
  *Will show all table column that field1 or field2 have the value string1 AND NOT string2*

## *Chapter 08 - PHPs MySQLi Interface*

### *0801 PHP programming interfaces*

- PHP provides several interfaces to work with mysql:
  - Mysql – The simplest and old interface
  - Mysqli – improves Mysql interface, Object Oriented and currently maintained
  - PDO – Modern, object oriented and multi-platform (support other DB too)

# *Appendix*

## *More operations on tables*

### *01. Delete table*

```
DROP  TABLE table_name
```

### *02. Deleting several rows tables*

```
DELETE FROM  table_name WHERE condition
```

### *03. Empting tables (with transaction logs)*

```
DROP  from table name
```

### *04. Empting tables (without transaction logs)*

```
TRANCATE TABLE table name
```

### *05. Update content in a table*

```
UPDATE  table_name SET field_name='xxxx' WHERE raw_condition
```

### *06. Update the contrust of a table*

```
ALTER TABLE table_name CHANGE 'field_name' 'field_name' CHAR(3)
```

## *Views*

View create  a new dynamic table (in the database) that all the time updates itself

### *Creating a view:*

```
CREATE VIEW view_name AS SELECT * from table_name;
```

# *Procedure*

## *01 - Define a new procedure (without parameters)*

Remark: // change the delimiter of the mysql command

```
DELIMITER //
mysql> CREATE PROCEDURE procedure_name()
→ BEGIN
→ SELECT * FROM table_name;
->END //
DELIMITER ;
```

## *02 - Calling a procedure*

```
CALL procedure_name()
```

## *03 - See all installed procedure*

```
SHOW PROCEDURE STATUS
```

## *04 - Running procedures from cron*

```
mysql -h hostname -u username -ppassword database_name -e “call procedure_name” -
```

## *05 - Define a new procedure (with parameters)*

```
use test;
delimiter /
CREATE PROCEDURE procedure_name() (OUT params INT)
begin
SELECT COUNT(*) INTO params FROM test.check;
end;
/
delimiter ;
```

This simple procedure counts the numbers of entries in the table "check" in "test" database. To check if the procedure has been created type:

```
show procedure status;
```

To see if the procedure is working type:

```
call procedure_name()(@samp);
select @samp;
```

This should display the number of rows in the "check" table

# *Subqueries in MySQL*

- Example No 1:

```
SELECT name, headofstate, population
FROM Country
WHERE population=(SELECT MAX(population) FROM Country);
```

- Example No 2:

```
SELECT MAX(tbl.nr) AS nr
FROM
    (SELECT countrycode, COUNT(*) AS nr
     FROM CountryLanguage
     WHERE isofficial='T'
     GROUP BY countrycode) AS tbl;
```

# *Variables in MySQL*

- You can store a value in a user-defined variable in one statement and use it later in another statement (This enables you passing values from one statement to another)
- User-defined variables are connection-specific (variable defined by one client can not be seeing by other clients
- All variables for a given client connection are automatically freed when that client exits
- User variables are written as @*var_name*
- You can define user-defined variable by using the SET statement

## *Define a new variable*

```
SET @var_name = expr [, @var_name = expr]   For SET, either = or := can be used
mysql> SET @t1=1, @t2=2, @t3:=4;
mysql> SELECT @t1, @t2, @t3, @t4 := @t1+@t2+@t3;
```

| @t1 | @t2 | @t3 | @t4 |
|---|---|---|---|
| 1 | 2 | 3 | 4 |

# Events

## The "hello world" of MySQL events

```
mysql> use test;
mysql> create table test.t (s1 timestamp);
Query OK, 0 rows affected (0.11 sec)

mysql> create event e on schedule every 1 second do
    insert into test.t values (current_timestamp);
Query OK, 1 row affected (0.00 sec)

/* 3-second delay */

mysql> select * from test.t;
+---------------------+
| s1                  |
+---------------------+
| 2006-04-05 15:44:26 |
| 2006-04-05 15:44:27 |
| 2006-04-05 15:44:28 |
+---------------------+
```

## Creating a new Event

```
CREATE EVENT
 [IF NOT EXISTS]
 event_name
 ON SCHEDULE schedule
 [ON COMPLETION [NOT] PRESERVE]
 [ENABLE | DISABLE]
 [COMMENT 'comment']
 DO sql_statement
```

There are two kinds of schedule:

- AT timestamp  /* "one-time schedule" */
- EVERY number-of-time-units time-unit [STARTS timestamp] [ENDS timestamp] /* "recurring schedule" */

**Examples:**

```
CREATE EVENT MONITOR_PROCESSLIST
    ON SCHEDULE EVERY '1' SECOND
    DO INSERT INTO db1.process_counter
       SELECT CURRENT_TIMESTAMP,COUNT(*)
       FROM INFORMATION_SCHEMA.PROCESSLIST;
```

This event will check every second "how many jobs are running" and will record the count in a table, along with the time that the count occurred. The process_counter will fill up, with 24 * 60 * 60 = 86,400 new rows every day, possibly.

```
CREATE EVENT DROP_TEST_T
   AT TIMESTAMP(CURRENT_DATE,'23:59:59')
   DO DROP TABLE test.t;
```

This event will happen once, at one minute to midnight, tonight. It will drop table t in db test.

### *Drop Event*

```
DROP EVENT
[IF EXISTS]
event_name
;
```