
Introduction to MySQL

MySQL

- Relational Database Management System
- Main characteristics:
 - ▶ Transactional support
 - ▶ Record-level constraints
 - ▶ Foreign keys (referential integrity constraints)
 - ▶ Views and nested queries
 - ▶ Stored procedure
 - ▶ ...
- Available in both commercial and free version
 - ▶ www.mysql.org/

Setup

- MySQL is constituted by a client and a server.
- MySQL Server
 - ▶ Start → Programs → MySQL → Start MySQL Server
- MySQL Client (text-base consolle)
 - ▶ Start → Programs → MySQL → MySQL Client

```
jd@lap-jd:~$ mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 5.0.45-Debian_lubuntu3-log Debian etch distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

Example schema

BOOK (ISBN, title, publishingYear, editor, genre, locationID)

AUTHOR (firstName, lastName, birthdate, shortBiography)

BOOKAUTHOR (ISBN, authorFirstName, authorLastName)

LOCATION (locationID, room, shelf, level)

LOAN (ISBN, userID, loanStartDate, loanEndDate)

USER (userID, firstName, lastName, address, phoneContact)

COMMENT (userID, ISBN, comment, judgement)

DB and Table Creation

DB creation

- Connecting to the DB Server:

```
create mysql -hhostname -uusername -p
```

- Creating the database:

```
mysql> create database database_name;
```

- Showing all the databases available on the system:

```
mysql> show databases;
```

- Selecting the working database:

```
mysql> use database_name;
```

Database creation

```
jd@lap-jd:--$ mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 5.0.45-Debian_lubuntu3-log Debian etch distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database library;
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| library |
| mysql |
+-----+
3 rows in set (0.00 sec)

mysql> use library;
Database changed
mysql> █
```

Tables creation

- Create Syntax:

```
mysql> create [temporary] table [if not exist]  
table_name [(definitions)] [options] [select]
```

- Definitions syntax:

```
fieldname TYPE [NOT NULL|NULL] [DEFAULT value]  
[auto_increment] [primary key] [foreign_keys  
(references)]
```

- Opzions syntax:

```
type = table_type;
```

Table types

- **MyISAM**: standard tables, they support compression but no transactions.
- **InnoDB**: transaction-safe (ACID properties), they support locking at tuple level.
- **BDB** (Berkeley DB): engine that supports transactions and locking at page-level.
- **HEAP**: tables are stored on main memory. and never on the persistent storage. Very fast engine without rollback.

MySQL Datatypes

• Standard Datatypes:

- ▶ NUMERIC o DECIMAL (DEC) : floating point values.
- ▶ INTEGER (INT) : standard integer value (4 bytes).
- ▶ FLOAT : signed single precision floating point value.
 - min 1.18×10^{-38} max 3.40×10^{38}
- ▶ DOUBLE : signed double precision floating point value.
 - min 2.23×10^{-308} max 1.80×10^{308}

MySQL Datatypes

• Strings and text:

- ▶ CHAR : fixed length string values:
 - e.g., char(20)
- ▶ VARCHAR : variable length string values with maximum size.
 - e.g., varchar(60)
- ▶ TEXT or BLOB : used to store long pieces of text and binary objects like images with maximum size of 64KB.
- ▶ ENUM : enumeration of pre-defined values.
 - e.g., Gender enum ('M', 'F')
- ▶ SET : like ENUM but the attribute may assume multiple values belonging to the set.

MySQL Datatypes

date and time:

- ▶ DATE: yyyy-mm-dd
- ▶ TIME: hh:mm:ss
- ▶ DATETIME: yyyy-mm-dd hh:mm:ss
- ▶ TIMESTAMP
 - if null, it assumes as value, the instance of the tuple creation.
- ▶ YEAR: two versions (yy and yyyy): YEAR(2) or YEAR(4)

Example - BOOK Table

BOOK (ISBN, title, publishingYear, editor, genre, locationID)

```
mysql> create table BOOK(  
-> ISBN    char (13) NOT NULL,  
-> title  varchar (40) ,  
-> publishingYear Year(2) ,  
-> editor  varchar (30) ,  
-> genre   varchar(30) ,  
-> locationID varchar (30) NOT NULL references  
        LOCATION(locationID) ,  
-> PRIMARY KEY (ISBN)  
-> ) type = InnoDB;
```

Example

```
jd@lap-jd:~$ mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.0.45-Debian_lubuntu3-log Debian etch distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use library
Database changed
mysql> create table BOOK(
  -> ISBN char(13) NOT NULL,
  -> title varchar (40),
  -> publishingYear Year(4),
  -> editor varchar (30),
  -> genre varchar(30),
  -> locationID varchar (30) NOT NULL references LOCATION (locationID),
  -> PRIMARY KEY (ISBN)
  -> ) type = InnoDB;
Query OK, 0 rows affected, 1 warning (0.05 sec)

mysql>
```

Example - USER Table

USER (userID, firstName, lastName, address, phoneContact)

```
mysql> create table USER(  
→ userID int NOT NULL auto_increment primary key,  
→ firstName varchar (30),  
→ lastName varchar (30),  
→ address varchar (40),  
→ phoneContact varchar (15)  
→ ) type = InnoDB;
```

Example: Multiple Keys

BOOKAUTHOR (ISBN, authorFirstName, authorLastName)

```
mysql> create table BOOKAUTHOR(  
-> ISBN char (13) NOT NULL references BOOK(ISBN) ,  
-> authorFirstName varchar (30) ,  
-> authorLastName varchar (30) ,  
-> PRIMARY KEY (ISBN, authorFirstName, authorLastName) ,  
-> FOREIGN KEY (authorFirstName, authorLastName)  
      references AUTHOR(firstName, lastName)  
-> ) type = InnoDB;
```

Controlling the DB Creation

- Showing all the tables of a database:

```
mysql> show tables;
```

```
mysql> show tables;
+-----+
| Tables_in_library |
+-----+
| AUTHOR             |
| BOOK               |
| BOOKAUTHOR        |
+-----+
3 rows in set (0.00 sec)
```

- Showing a table's schema:

```
mysql> describe table_name;
```

```
mysql> describe BOOK;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| ISBN           | char(13)      | NO   | PRI |          |       |
| title          | varchar(40)   | YES  |     | NULL    |       |
| publishingYear | year(4)       | YES  |     | NULL    |       |
| editor         | varchar(30)   | YES  |     | NULL    |       |
| genre          | varchar(30)   | YES  |     | NULL    |       |
| locationID     | varchar(30)   | NO   |     |          |       |
+-----+-----+-----+-----+-----+-----+
```

DB and Tables Removal

- Destroying a DB:

```
mysql> drop database database_name;
```

- Removing a table:

```
mysql> drop [temporary] table [if exists]  
table_name[, table2_name,...];
```

Modifying Table Structures

- Renaming:

```
mysql> ALTER TABLE old_name RENAME new_name;
```

- Adding a new field

```
mysql> ALTER TABLE table_name  
-> ADD fieldname TYPE;
```

- Removing a field

```
mysql> ALTER TABLE table_name  
-> DROP nome;
```

Altering a table

- Modifying a field:

```
mysql> ALTER TABLE table_name  
    -> MODIFY old_field new_field TYPE;
```

- Adding a key

```
mysql> ALTER TABLE table_name  
    -> ADD PRIMARY KEY (field);
```

- *Creating a DB from a script:*

```
mysql> source nome_file;
```

DML

SQL Insert

- Adding a tuple to a table.

```
mysql> INSERT INTO table_name VALUES ('char_val_1', 'string_val_2', num_val, ...);
```

- e.g.,

```
mysql> INSERT INTO BOOK VALUES  
('88-386-6030-1', 'basi di dati', '2003', 'Mc-Graw  
Hill', 'Computer Science', 'IT31'),  
('88-386-0762-1', 'elementi di fisica', '1998',  
'Pearson', 'Physics', 'PHY33');
```

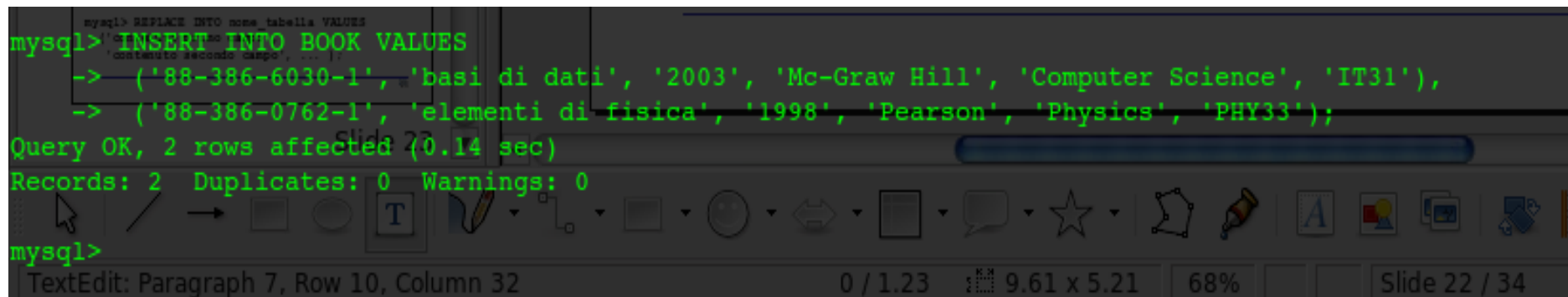
SQL Insert (2)

- To verify if the insert statement executed correctly it is possible to execute a scan query.

```
mysql> SELECT * FROM table_name;
```

- Replacing a tuple:

```
mysql> REPLACE INTO table_name VALUES  
('char_val', 'string_val', val );
```



```
mysql> REPLACE INTO nome_tabella VALUES  
mysql> INSERT INTO BOOK VALUES  
-> ('88-386-6030-1', 'basi di dati', '2003', 'Mc-Graw Hill', 'Computer Science', 'IT31'),  
-> ('88-386-0762-1', 'elementi di fisica', '1998', 'Pearson', 'Physics', 'PHY33');  
Query OK, 2 rows affected (0.14 sec)  
Records: 2 Duplicates: 0 Warnings: 0  
mysql>
```

SQL Delete

- Emptying a table:

```
mysql> DELETE FROM table_name;
```

alternative:

```
mysql> TRUNCATE TABLE table_name;
```

- Removing selected tuples:

```
mysql> DELETE FROM table_name;  
-> WHERE (cond);
```

SQL Update

- Update syntax:

```
mysql> UPDATE table_name
```

- SET *col1_name=expr1* [, *col2_name=expr2*,...]
- [WHERE *cond*];

- Esempio:

```
mysql> UPDATE LOAN
```

- SET *loanEndDate='2005-11-29'*
- WHERE *genre='Computer Science' AND userID=15 and loanStartDate='2005-10-15'*;

Queries

Queries

- Scan query:

```
mysql> SELECT * from table_name;
```

- Select query:

```
mysql> SELECT field1, field2, ...  
-> from table_name;
```

- Select-From-Where:

```
mysql> SELECT field1, field2, ...  
-> from table_name;  
-> where cond;
```

Is the Interpreter Smart?

```
mysql> select ISBN  
-> from BOOK join BOOKAUTHOR on ISBN = ISBN;
```

ERROR 1052 (23000): Column 'ISBN' in field list is ambiguous

Solution (dot notation and aliases):

```
mysql> select D.ISBN  
-> from BOOK B join BOOKAUTHOR BA on  
-> B.ISBN = BA.ISBN;
```

Is the Interpreter Smart?

```
mysql> select title, firstName, count(*)  
-> from BOOK B join BOOKAUTHOR BA  
-> on B.ISBN = BA.ISBN  
-> where editor = 'Mc-Graw Hill';
```

ERROR 1140 (42000): Mixing of GROUP columns (MIN(),MAX(),COUNT(),...) with no GROUP columns is illegal if there is no GROUP BY clause

Queries

- COUNT: counts the tuples that satisfy the condition in the where clause.

```
mysql> SELECT count(*) from LOAN
-> where loanStartDate >= '2005/10/01'
-> and loanStartDate <= '2005/10/31';
```

- GROUP BY: groups the tuples on the basis of a given field.

```
mysql> SELECT count(*) as loanNumber, ISBN
-> from LOAN
-> Group by ISBN;
```

Interrogazioni semplici

```
mysql> select title, editor  
-> from BOOK  
-> where editor < "P";
```

```
mysql> select title, editor  
-> from BOOK  
-> where editor < 'P';  
  
+-----+-----+  
| title | editor |  
+-----+-----+  
| basi di dati | Mc-Graw Hill |  
+-----+-----+  
1 row in set (0.00 sec)  
  
mysql> █
```

Adding a bit of complexity

- Finding the books of Computer Science borrowed in June 2005

```
mysql> select title
-> from LOAN L join BOOK B
-> where L.ISBN = B.ISBN
-> and loanStartDate >= '2005/06/01'
-> and loanStartDate <= '2005/06/30';
```

- I libri currently under loan

```
mysql> select title
-> from LOAN L join BOOK B
-> where L.ISBN = B.ISBN
-> and loanEndDate is null;
```

Nested Queries

- Example: Finding the IDs and the names of users that never borrowed a book of Logic and Computation with editor Springer-Verlag.

```
select userID, firstName
from USER
where userID not in (select userID
                    from LOAN L join BOOK B on L.ISBN = B.ISBN
                    where B.editor = 'Springer-Verlag' and
                           B.genre = 'Logic and Computation');
```