THE UNIVERSITY OF CHICAGO

DEPARTMENT OF ECONOMICS

PRACTICAL COMPUTING FOR ECONOMISTS
WEEK 6 & 7: DATA MANIPULATION, DATA MANAGEMENT, AND DATA STORAGE

SPRING 2014

Time and location

Week 6 & 7, Harper Center

Instructor

Jörn Boehnke
jboehnke@uchicago.edu

Objectives

Week 6 & Week 7 of this workshop are aimed to give an introduction to efficient data manipulation, management, and storage. We will cover:

Data structuring: It is very important to know about data types. It is a classification identifying one of various types of data, such as real, integer or boolean, that determines the possible values for that type. Storing numbers as texts, for example, will reduce efficiency and impede value comparison. Every data type has its own effective way to store it.

RegEx: Regular expressions (RegEx) is a very powerful text editing tool and an important part of this workshop. Any modern programming language (e.g. Java, C++, VB), statistical software (e.g. R, Stata, MATLAB) and advanced plain text editor (e.g. Notepad++, TextWrangler, Sublime Text) are capable of RegEx. It provides a concise and flexible means to match (specify and recognize) strings of text, such as particular characters, words, or patterns of characters. With its functionality, RegEx offers a very efficient and easy way to edit / manipulate text-based data.

MySQL: MySQL is the world's second most widely used database. It is the most widely used open source database.[1] You will most likely receive an SQL dump if you apply for data from an internet company.
(My)SQL interfaces well with programming language and statistical software. It allows you to query / alter subsets of the data easy and fast. You can think of (My)SQL as free Microsoft Excel on speed, and without row limitations.

MongoDB & JSON: MongoDB is the world's most widely used NonSQL DB. You can store anything in it, and MongoDB will do the magic for you. It is the natural choice if you are having a hard time defining a column structure for your data. JSON stands for JavaScript Object Notation and is the data structuring language MongoDB operates in. It is, however, not limited to MongoDB. The Twitter firehose application, for example, required knowledge of JSON…

Software Requirements

Please get the following software running on your operating system:

[1] http://db-engines.com/en/ranking_trend

- RegEx compatible text editor
  - for Windows users: Notepad++ is very nice (available for free at http://notepad-plus-plus.org/)
  - for Mac users: TextWrangler is the way to go (available for free at http://download.cnet.com/TextWrangler/3000-2351_4-10220012.html)
  - for Linux users: Sublime Text (evaluation copy available at http://www.sublimetext.com/2)

- MySQL Community Server 5.6 and MySQL Workbench
  - http://dev.mysql.com/downloads/mysql/ (The MySQL Community Server 5.6 Installation will include MySQL Workbench)

- MongoDB
  - follow the instructions at https://www.mongodb.org/downloads

I am assuming that R and RStudio is installed and running on your machine.

Testing

Notepad++ / TextWrangler / Sublime Text

To know if your installation of this advanced plain text editor works, simply try to view any txt with it and see if it shows you the content correctly.

MySQL

Start the command-line tool `mysql` (probably located in "[path]\MySQL Server 5.6\bin\mysql.exe" or similar). Then create a database `mytest` and "use" it:

```
mysql> CREATE DATABASE mytest;
Query OK, 1 row affected (0.00 sec)

mysql> USE mytest;
Database changed
```

You can create and populate an example table shop with these statements:

```
mysql> CREATE TABLE shop (
    article INT(4) UNSIGNED ZEROFILL DEFAULT '0000' NOT NULL,
    dealer  CHAR(20)                 DEFAULT ''     NOT NULL,
    price   DOUBLE(16,2)             DEFAULT '0.00' NOT NULL,
    PRIMARY KEY(article, dealer));
mysql> INSERT INTO shop VALUES
    (1,'A',3.45),(1,'B',3.99),(2,'A',10.99),(3,'B',1.45),
    (3,'C',1.69),(3,'D',1.25),(4,'D',19.95);
```

Now query the newly created table:

```
mysql> SELECT * FROM shop;
+---------+--------+-------+
| article | dealer | price |
+---------+--------+-------+
|    0001 | A      |  3.45 |
|    0001 | B      |  3.99 |
|    0002 | A      | 10.99 |
|    0003 | B      |  1.45 |
|    0003 | C      |  1.69 |
|    0003 | D      |  1.25 |
|    0004 | D      | 19.95 |
+---------+--------+-------+
7 rows in set (0.00 sec)
```

MongoDB
Insert Multiple Documents Using a For Loop
You can add documents to a new or existing collection by using a JavaScript for loop run from the mongo shell.

1. From the mongo shell, insert new documents into a `testData` collection using the following for loop. If the `testData` collection does not exist, MongoDB creates the collection implicitly.

```
for (var i = 1; i <= 25; i++) db.testData.insert( { x : i } )
```

2. Use find() to query the collection:

```
db.testData.find()
```

The mongo shell displays the first 20 documents in the collection. Your `ObjectId` values will be different:

```
{ "_id" : ObjectId("51a7dc7b2cacf40b79990be6"), "x" : 1 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990be7"), "x" : 2 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990be8"), "x" : 3 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990be9"), "x" : 4 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bea"), "x" : 5 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990beb"), "x" : 6 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bec"), "x" : 7 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bed"), "x" : 8 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bee"), "x" : 9 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bef"), "x" : 10 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf0"), "x" : 11 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf1"), "x" : 12 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf2"), "x" : 13 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf3"), "x" : 14 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf4"), "x" : 15 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf5"), "x" : 16 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf6"), "x" : 17 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf7"), "x" : 18 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf8"), "x" : 19 }
{ "_id" : ObjectId("51a7dc7b2cacf40b79990bf9"), "x" : 20 }
```

3. The `find()` returns a cursor. To iterate the cursor and return more documents use the `it` operation in the mongo shell. The mongo shell will exhaust the cursor, and return the following documents:

```
{ "_id" : ObjectId("51a7dce92cacf40b79990bfc"), "x" : 21 }
{ "_id" : ObjectId("51a7dce92cacf40b79990bfd"), "x" : 22 }
{ "_id" : ObjectId("51a7dce92cacf40b79990bfe"), "x" : 23 }
{ "_id" : ObjectId("51a7dce92cacf40b79990bff"), "x" : 24 }
{ "_id" : ObjectId("51a7dce92cacf40b79990c00"), "x" : 25 }
```

Having problems?
If you encounter problems in setting up and testing your system in the described way, please consult Google first. Since it is highly unlikely that you are the first one with that problem, about 99% of all problems can be resolved by stating the problem / error message in Google in a well worded fashion.

If this does not help, I will be 15 minutes early at my first session. Please use that time. I expect you to have the database software and test code up and running.