A continuation of the MySQL tutorial

On our last MySQL tutorial, we left off at the stage where we downloaded the 2008 fire hydrant parking violation file.

From there, we ran queries that allowed us to summarize information in some of the same ways we did with the pivot tables.

However, working with merely one table containing a few thousand records is only a starting point to exploiting the power of MySQL. Now we want to build a master table containing data of all the years from 2008 to 2014. This will allow to perform queries that reveal patterns – and story ideas -- over the span of several years.

There are two ways to do this: create a master table into which we can import the individual tables; use a query to combine all the tables.

The advantage of the second method is that it takes up less space on your hard drive. So this is essentially how it works.

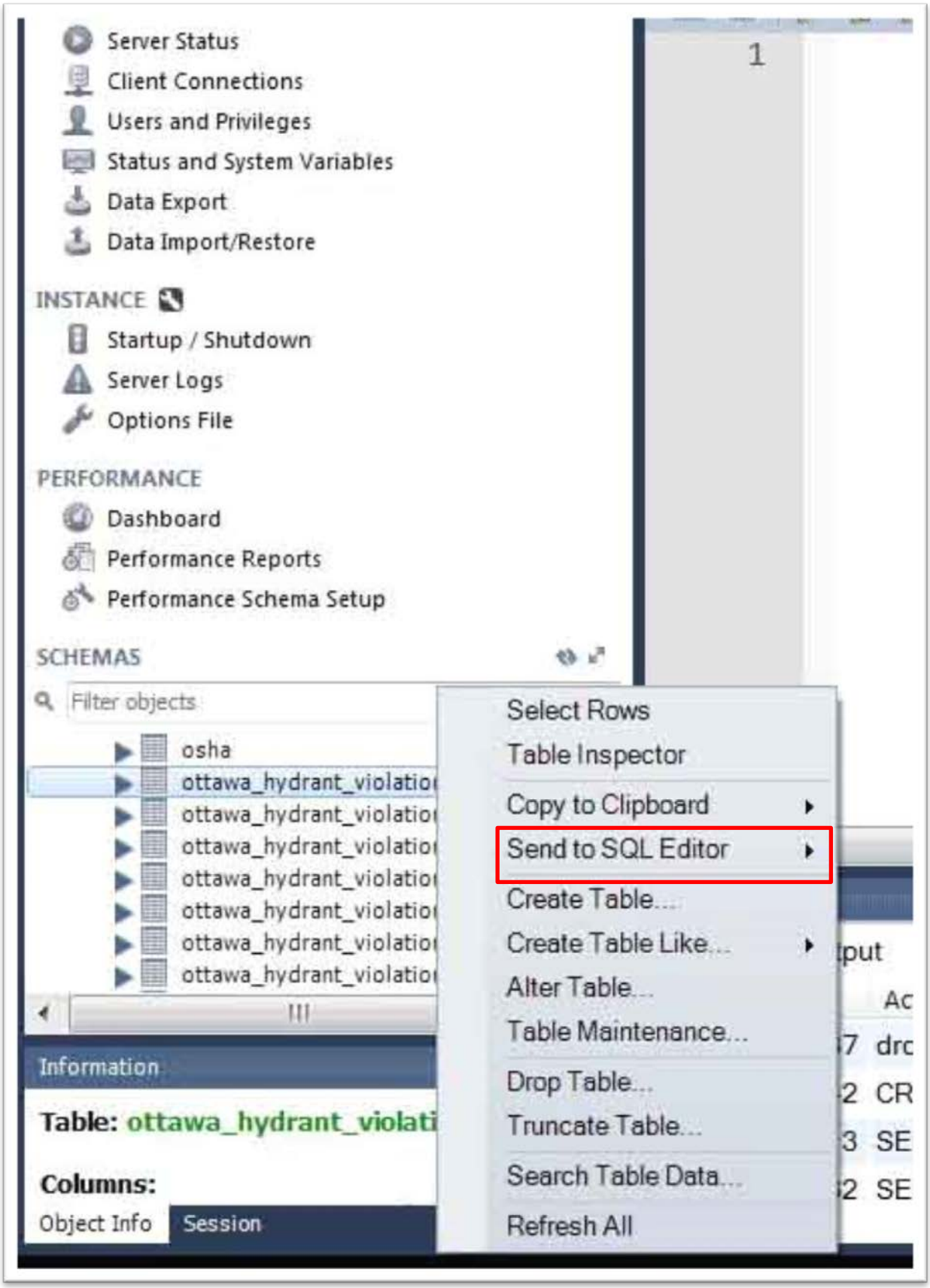
You write a query that allows you to pull records from each table. In MySQL-speak, that query is called a "View". Whenever a query is run against this view, it recovers the data from each table, and reassembles that data – fines for parking too close to fire hydrants, times of day, street names, etc. -- in a single result. So, essentially, treat the "view" as all the tables put

together. If you're still confused, it should be clearer after completing the tutorial, which will walk you through the paces of creating this strange thing we're calling a "view".

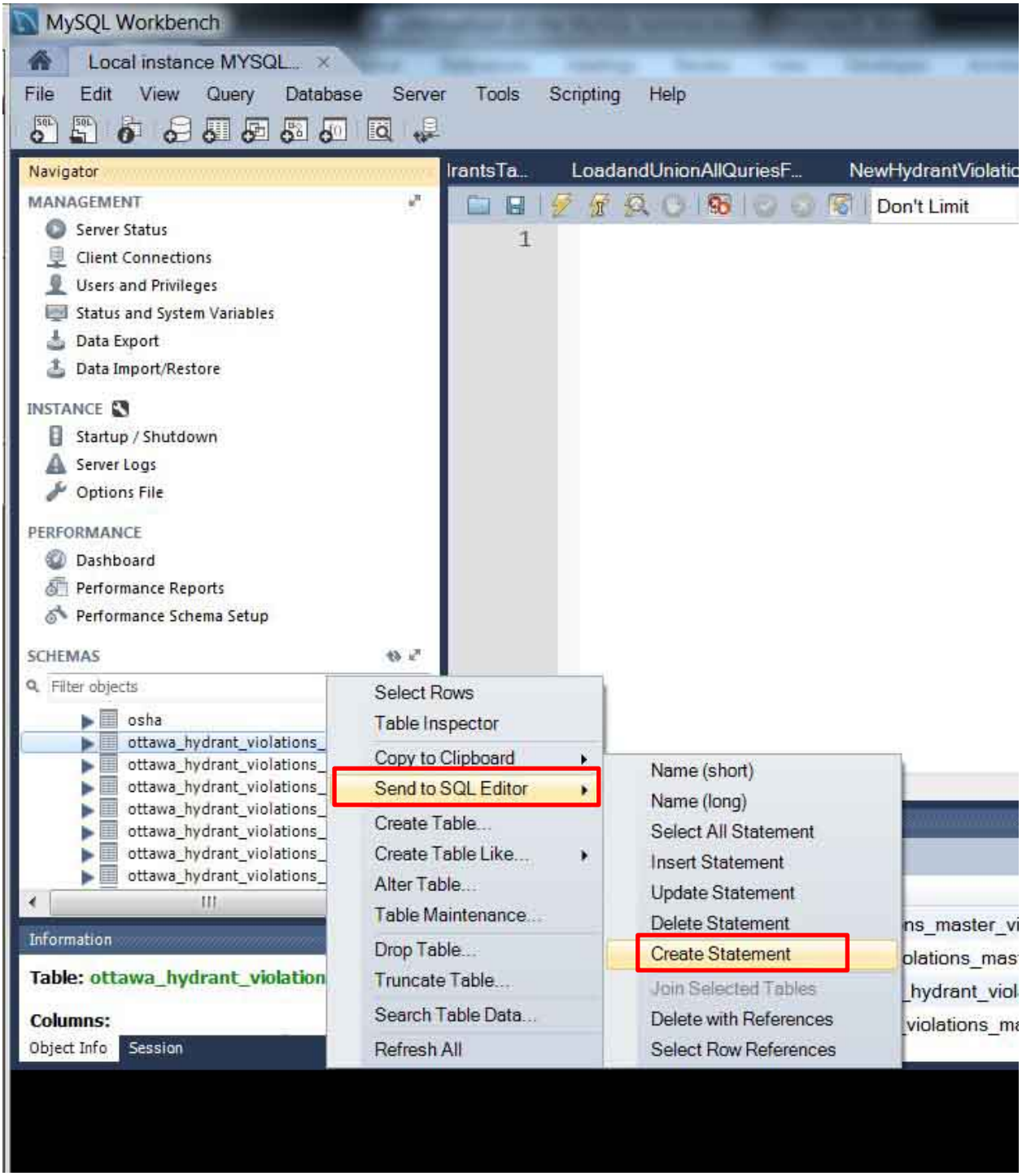
Let's get started.

1. Download the zip file containing the rest of the Ottawa fire hydrant tables and save them as individual tables in the same area on your hard drive that contains the first table.
2. We have to create a SEPARATE table for EACH of the subsequent parking violations tables for 2008 to 2014 in a new query browser or tab.
3. Select a new query tab, from the "File" section of the menu, or by clicking on the small "SQL" icon right underneath "File".
4. We will then copy the formula MySQL used to create the 2008 data into this new query tab.
5. To do this, right-click on the "ottawa_hydrant_violations_2008" table in your Parking schema to obtain a drop-down menu which you can see in

this screen shot.


Server Status
Client Connections
Users and Privileges
Status and System Variables
Data Export
Data Import/Restore
INSTANCE
Startup / Shutdown
Server Logs
Options File
PERFORMANCE
Dashboard
Performance Reports
Performance Schema Setup
SCHEMAS
Filter objects
osha
ottawa_hydrant_violatio
ottawa_hydrant_violatio
ottawa_hydrant_violatio
ottawa_hydrant_violatio
ottawa_hydrant_violatio
ottawa_hydrant_violatio
ottawa_hydrant_violatio
Information
Table: ottawa_hydrant_violati
Columns:
Object Info
Session
Select Rows
Table Inspector
Copy to Clipboard
Send to SQL Editor
Create Table...
Create Table Like...
Alter Table...
Table Maintenance...
Drop Table...
Truncate Table...
Search Table Data...
Refresh All

6. Select the “Send to SQL Editor” option, and then the “Create Statement” option.



7. The “Create Statement” option will populate the browser to the right with the “CREATE TABLE” query.

```
CREATE TABLE `ottawa_hydrant_violations_2008` (
  `DATE_NEW` date DEFAULT NULL,
  `Time_New` varchar(5) DEFAULT NULL,
  `STREET` varchar(50) DEFAULT NULL,
  `BETWEEN_` varchar(50) DEFAULT NULL,
  `AND_` varchar(50) DEFAULT NULL,
  `SIDE_OF_STREET` varchar(5) DEFAULT NULL,
  `TOTAL_FINES_AND_FEES` float DEFAULT NULL,
  `AMOUNTDUE` float DEFAULT NULL,
  `DUE_DATE_NEW` date DEFAULT NULL,
  `REC_STATUS_DATE_NEW` date DEFAULT NULL,
  `REVIEW_CODE` varchar(5) DEFAULT NULL,
  `TRIAL_CODE` varchar(5) DEFAULT NULL,
  `EMPTY` varchar(45) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

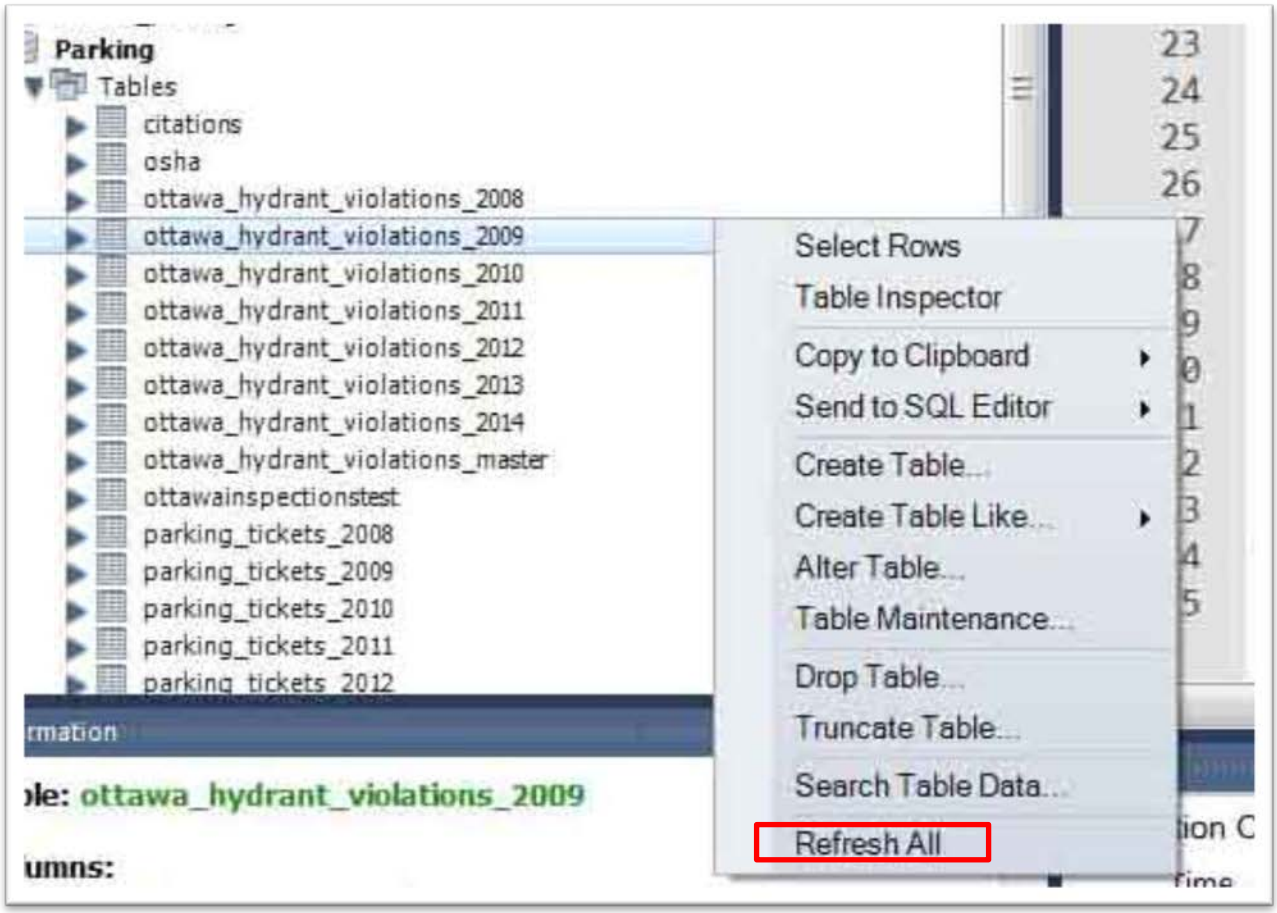
8. We have created a table for 2008. Now we have to create tables for the rest of the years. But instead of doing it manually like we did in the first tutorial, all we have to do is copy this query, paste it below, and then change the

year to 2009.

```
CREATE TABLE `ottawa_hydrant_violations_2009` (
  `DATE_NEW` date DEFAULT NULL,
  `Time_New` varchar(5) DEFAULT NULL,
  `STREET` varchar(50) DEFAULT NULL,
  `BETWEEN_` varchar(50) DEFAULT NULL,
  `AND_` varchar(50) DEFAULT NULL,
  `SIDE_OF_STREET` varchar(5) DEFAULT NULL,
  `TOTAL_FINES_AND_FEES` float DEFAULT NULL,
  `AMOUNTDUE` float DEFAULT NULL,
  `DUE_DATE_NEW` date DEFAULT NULL,
  `REC_STATUS_DATE_NEW` date DEFAULT NULL,
  `REVIEW_CODE` varchar(5) DEFAULT NULL,
  `TRIAL_CODE` varchar(5) DEFAULT NULL,
  `EMPTY` varchar(45) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

9. Note that the entire “CREATE TABLE” query is in brackets. The part outside the brackets, highlighted above, is added automatically and is the information MySQL needs to import the table. Once again, notice that we changed the year of the table we’ve just copied.
10. Repeat the steps for the remaining years, making sure to leave an empty row between each individual query, and ensuring that each query ends with a semi-colon.

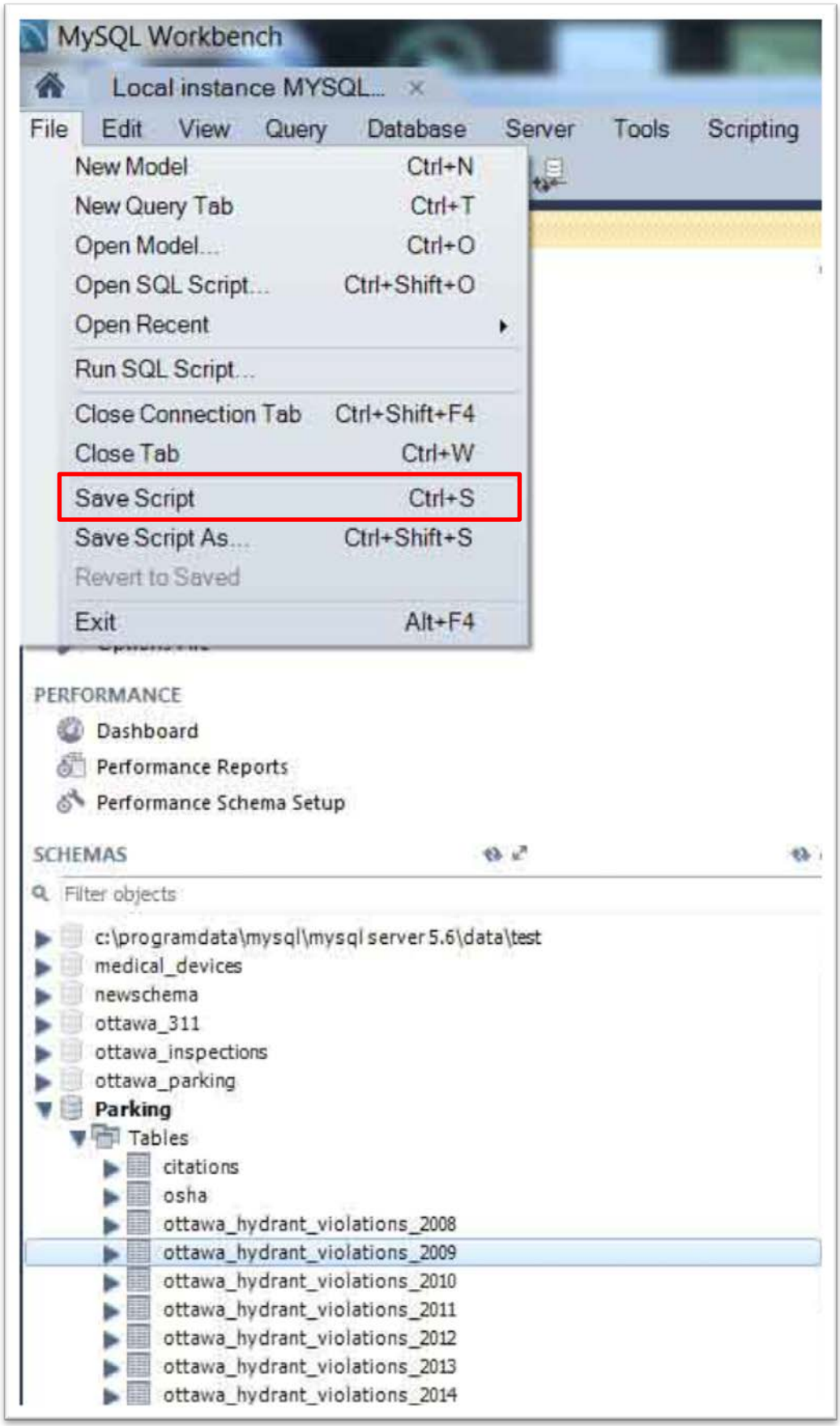
11. If you refresh your Parking Schema to the left, you'll see all the tables.



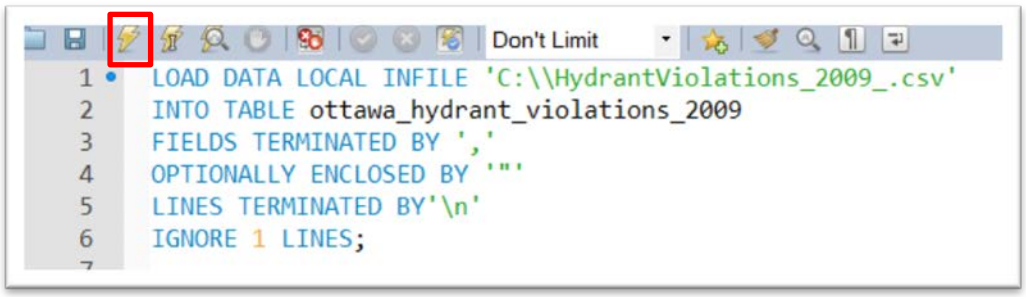
12. By selecting the "Refresh All" option, you'll see the new tables.

13. Now that we have all of our CREATE TABLE queries in one place, let's save the queries as one file by clicking on the "File" section of the menu at the top of the MySQL

Workbench browser.


MySQL Workbench
Local instance MYSQL...
File
Edit
View
Query
Database
Server
Tools
Scripting
New Model Ctrl+N
New Query Tab Ctrl+T
Open Model... Ctrl+O
Open SQL Script... Ctrl+Shift+O
Open Recent
Run SQL Script...
Close Connection Tab Ctrl+Shift+F4
Close Tab Ctrl+W
Save Script Ctrl+S
Save Script As... Ctrl+Shift+S
Revert to Saved
Exit Alt+F4
PERFORMANCE
Dashboard
Performance Reports
Performance Schema Setup
SCHEMAS
Filter objects
c:\programdata\mysql\mysql server 5.6\data\test
medical_devices
newschema
ottawa_311
ottawa_inspections
ottawa_parking
Parking
Tables
citations
osha
ottawa_hydrant_violations_2008
ottawa_hydrant_violations_2009
ottawa_hydrant_violations_2010
ottawa_hydrant_violations_2011
ottawa_hydrant_violations_2012
ottawa_hydrant_violations_2013
ottawa_hydrant_violations_2014

14. Save the script in the same section of your hard drive that contains the tables for this tutorial.
15. To download the MySQL script with the CREATE TABLE queries, please click here. Save this query and open it in a Workbench query browser to see all the queries in one tab.
16. Open a new query tab.
17. Make sure that you have established your schema, in the case "Parking", as your "default" schema. (NOTE: If you neglect to do this, MySQL won't know where to find the table. You can also accomplish the same goal by using this query at beginning of the entire exercise: "USE Parking". This query instructs MySQL to use the Parking place all the tables into the Parking schema.)
18. Now we can use the "LOAD DATA LOCAL INFILE" command to populate each table.



19. Run the query.
20. Hit enter, and in line number eight, let's use a SELECT query to count the number of records to ensure that we

got everything.

```
LOAD DATA LOCAL INFILE 'C:\\HydrantViolations_2009_.csv'
INTO TABLE ottawa_hydrant_violations_2009
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY'\n'
IGNORE 1 LINES;

select count(*)
from ottawa_hydrant_violations_2009;
```

21. To run the “SELECT COUNT” query, highlight query, and then run it.

```
LOAD DATA LOCAL INFILE 'C:\\HydrantViolations_2009_.csv'
INTO TABLE ottawa_hydrant_violations_2009
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY'\n'
IGNORE 1 LINES;

select count(*)
from ottawa_hydrant_violations_2009;
```

```
LOAD DATA LOCAL INFILE 'C:\\HydrantViolations_2009_.csv'
INTO TABLE ottawa_hydrant_violations_2009
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY'\n'
IGNORE 1 LINES;

select count(*)
from ottawa_hydrant_violations_2009;
```

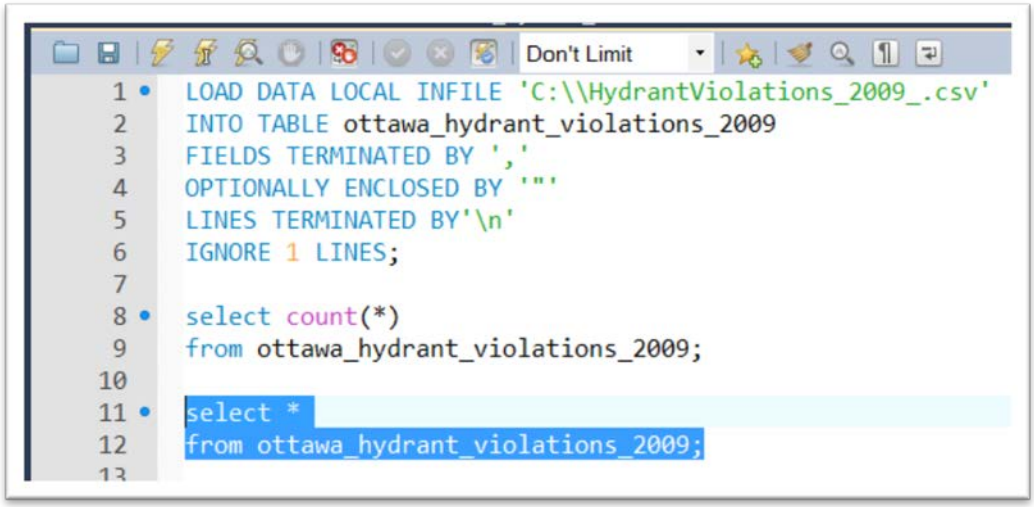
count(*)
5465

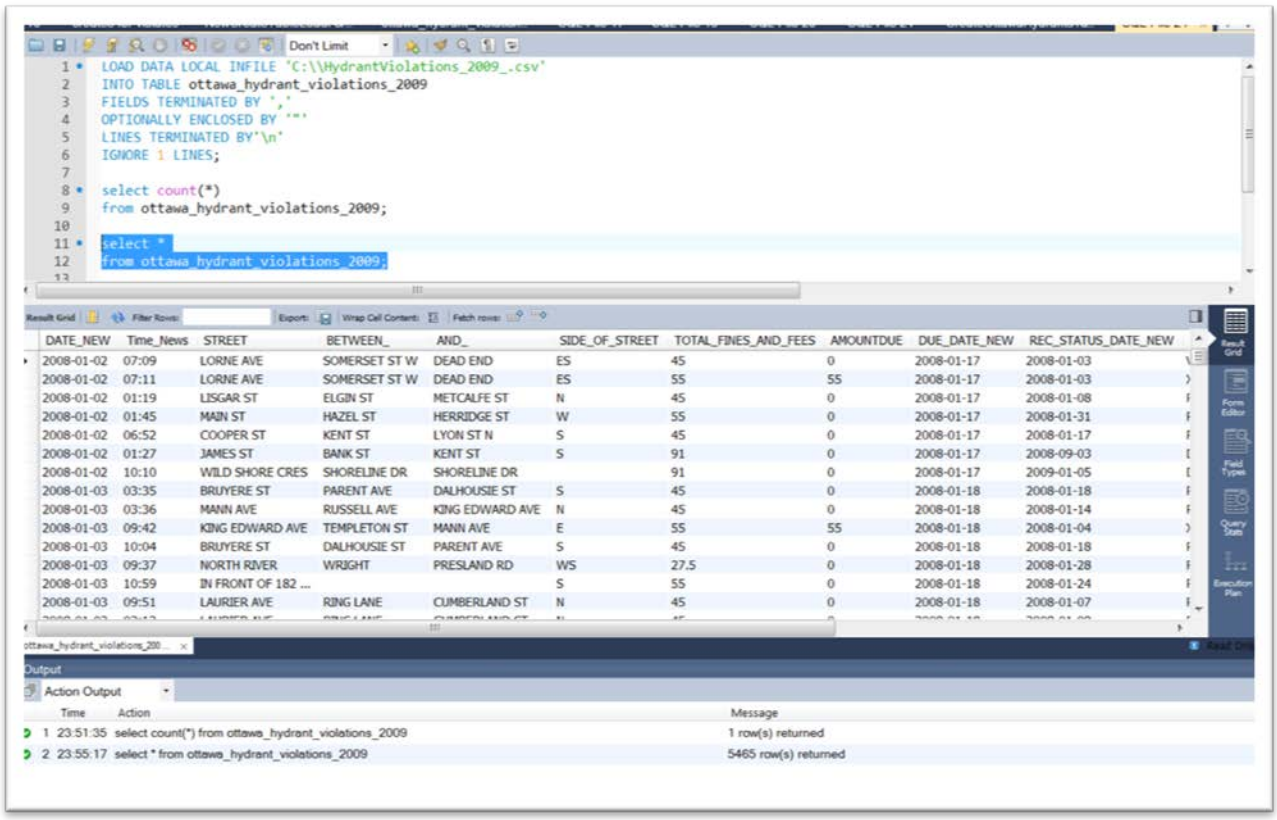
22. When pasting queries in the same tab – more convenient than creating a brand new tab for each query – be sure end the preceding query with a semi-colon, enter a blank row, and then create a new query, which you HIGHLIGHT, as you can see in the screen shot above, and then run. If you neglect to highlight the queries individually, MySQL will run them all, giving you skewed results. If you accidentally do this, you can simple empty

the table of all the data using the “TRUNCATE” query (“TRUNCATE ottawa_hydrant_violations_2009”), and then re-run the load query.

23. We have successfully loaded the 2009 data into the table we’ve created.

24. Hit enter, and use a SELECT query to see the table.





DATE_NEW	Time_News	STREET	BETWEEN_	AND_	SIDE_OF_STREET	TOTAL_FINES_AND_FEES	AMOUNTDUE	DUE_DATE_NEW	REC_STATUS_DATE_NEW
2008-01-02	07:09	LORNE AVE	SOMERSET ST W	DEAD END	ES	45	0	2008-01-17	2008-01-03
2008-01-02	07:11	LORNE AVE	SOMERSET ST W	DEAD END	ES	55	55	2008-01-17	2008-01-03
2008-01-02	01:19	LISGAR ST	ELGIN ST	METCALFE ST	N	45	0	2008-01-17	2008-01-08
2008-01-02	01:45	MAIN ST	HAZEL ST	HERRIDGE ST	W	55	0	2008-01-17	2008-01-31
2008-01-02	06:52	COOPER ST	KENT ST	LYON ST N	S	45	0	2008-01-17	2008-01-17
2008-01-02	01:27	JAMES ST	BANK ST	KENT ST	S	91	0	2008-01-17	2008-09-03
2008-01-02	10:10	WILD SHORE CRES	SHORELINE DR	SHORELINE DR		91	0	2008-01-17	2009-01-05
2008-01-03	03:35	BRUYERE ST	PARENT AVE	DALHOUSIE ST	S	45	0	2008-01-18	2008-01-18
2008-01-03	03:36	MANN AVE	RUSSELL AVE	KING EDWARD AVE	N	45	0	2008-01-18	2008-01-14
2008-01-03	09:42	KING EDWARD AVE	TEMPLETON ST	MANN AVE	E	55	55	2008-01-18	2008-01-04
2008-01-03	10:04	BRUYERE ST	DALHOUSIE ST	PARENT AVE	S	45	0	2008-01-18	2008-01-18
2008-01-03	09:37	NORTH RIVER	WRIGHT	PRESLAND RD	WS	27.5	0	2008-01-18	2008-01-28
2008-01-03	10:59	IN FRONT OF 182 ...			S	55	0	2008-01-18	2008-01-24
2008-01-03	09:51	LAURIER AVE	RING LANE	CUMBERLAND ST	N	45	0	2008-01-18	2008-01-07

25. We have successfully imported the 2009 data into the table!!

26. Now complete the same step – in the same query tab – for 2010, making sure to enter a blank row, and change

the year of the table in the LOAD statement, and the table name after the “INTO TABLE” portion of the script.

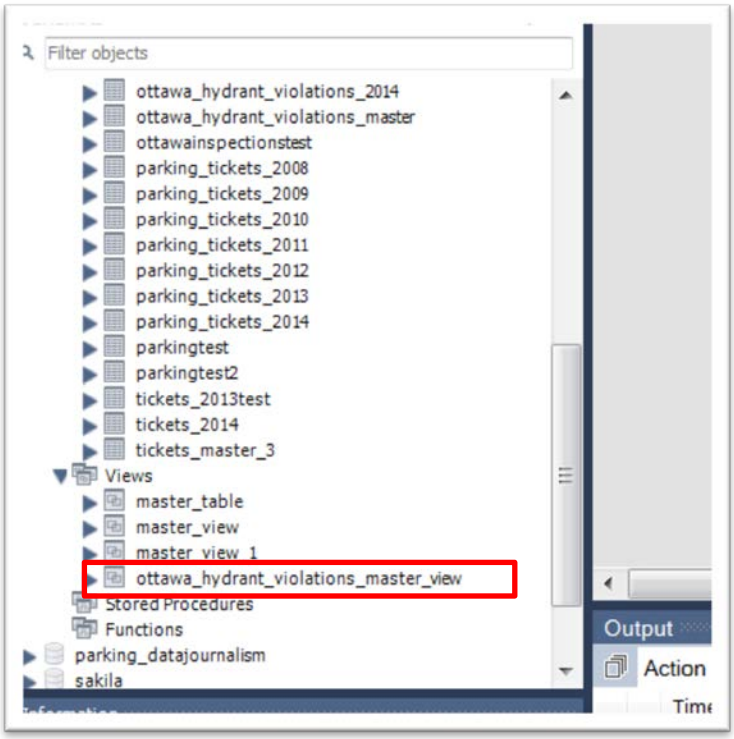
```
LOAD DATA LOCAL INFILE 'C:\\HydrantViolations_2009_.csv'
INTO TABLE ottawa_hydrant_violations_2009
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY'\n'
IGNORE 1 LINES;

select count(*)
from ottawa_hydrant_violations_2009;

select *
from ottawa_hydrant_violations_2009;

LOAD DATA LOCAL INFILE 'C:\\HydrantViolations_2010_.csv'
INTO TABLE ottawa_hydrant_violations_2010
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY'\n'
IGNORE 1 LINES;

select count(*)
from ottawa_hydrant_violations_2010;

select *
from ottawa_hydrant_violations_2010;
```

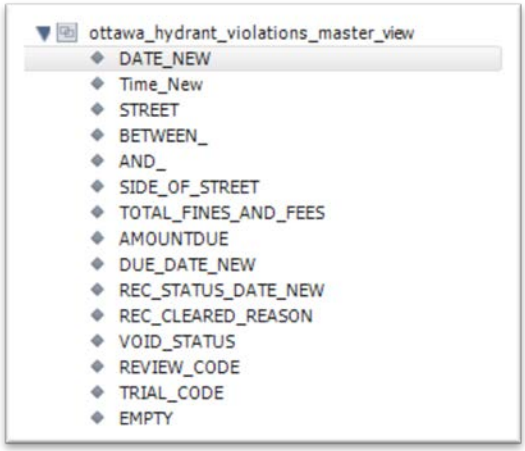
27. To repeat, be sure to highlight each section of the query that you want to run. In this case, we are loading the data into the 2010 table, then we counting the records to ensure we got everything, and then selecting the table to make sure all the data is in the right format.
28. Repeat the same steps for the rest of the years: 2011, 2012, 2013, 2014.
29. Now we are ready to use a query to create a master that will draw data from each individual table.
30. Save the select query tab (give it a meaningful title) and open a new one.

31. We will use a “UNION ALL” query to do this, which is covered on pages 197 to 198 of Computer-Assisted Reporting.
32. This is what the query looks like:

```
CREATE VIEW ottawa_hydrant_violations_master_view AS
    SELECT * FROM ottawa_hydrant_violations_2008 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2009 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2010 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2011 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2012 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2013 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2014;
```

33. This is a CREATE VIEW query. The “ottawa_hydrant_violations_master_view” name is what we are giving to the view. Run the query.
34. To find this table, go to your Schema on the left of the MySQL Workbench browser, scroll down to the “Views” section, and click on the arrow to the left of the “Views” icon to see the individual tables.


Filter objects
ottawa_hydrant_violations_2014
ottawa_hydrant_violations_master
ottawainspectionstest
parking_tickets_2008
parking_tickets_2009
parking_tickets_2010
parking_tickets_2011
parking_tickets_2012
parking_tickets_2013
parking_tickets_2014
parkingtest
parkingtest2
tickets_2013test
tickets_2014
tickets_master_3
Views
master_table
master_view
master_view_1
ottawa_hydrant_violations_master_view
Stored Procedures
Functions
parking_datajournalism
sakila
Output
Action


ottawa_hydrant_violations_master_view
DATE_NEW
Time_New
STREET
BETWEEN_
AND_
SIDE_OF_STREET
TOTAL_FINES_AND_FEES
AMOUNTDUE
DUE_DATE_NEW
REC_STATUS_DATE_NEW
REC_CLEARED_REASON
VOID_STATUS
REVIEW_CODE
TRIAL_CODE
EMPTY

35. Now let's use our "SELECT COUNT(*)" query to see the number of records.

```
CREATE VIEW ottawa_hydrant_violations_master_view AS
    SELECT * FROM ottawa_hydrant_violations_2008 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2009 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2010 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2011 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2012 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2013 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2014;

SELECT COUNT(*)
FROM ottawa_hydrant_violations_master_view;

SELECT *
FROM ottawa_hydrant_violations_master_view;

select YEAR(date_new) as Date, COUNT(amountdue) as Number
from ottawa_hydrant_violations_master_view
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

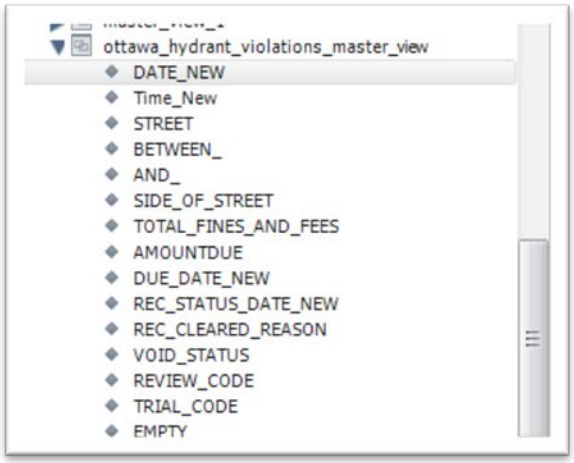
Date	Number
2013	7431
2012	6522
2011	6167
2008	5465
2009	5131
2014	3124

36. Running thE query on the master view table that we've created, pulls all the records from each of the tables above. As we mentioned at the beginning of this tutorial, this is a good method to use because the view you've created does not take up any additional hard-drive space. I would also draw your attention to the instruction we've given the browser, which is to place no limits ("Don't Limit') on the number of records we import, a drop-down menu provides options that limit the number of rows. This

comes in handy if you're dealing with a table that contains millions of records. Instead of loading the entire table, which could crash your hard drive, you just use Workbench's limit option to import the first few thousand rows, just to make sure that the dataset is intact.

37. Now we can begin running queries that pull data from all the tables.

38. Be sure to click the arrow beside your view table on the left to get the column headings.



39. This allows you to see which columns you want to import, and then do so by simply clicking on the column name to produce it in your query.

40. So if we wanted to select certain columns, we would click on each one we want to import, and separate it with

a comma.

```sql
CREATE VIEW ottawa_hydrant_violations_master_view AS
    SELECT * FROM ottawa_hydrant_violations_2008 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2009 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2010 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2011 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2012 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2013 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2014;

    SELECT COUNT(*)
    FROM ottawa_hydrant_violations_master_view;

SELECT DATE_NEW,STREET,AMOUNTDUE
FROM ottawa_hydrant_violations_master_view;
```

41. In this example, we want to select the DATE_NEW, STREET, and AMOUNTDUE columns from the tables.

NewCreateTableLoadFor... ottawa_hydrant_violation... SQL File 17 SQL File 18* SQL File 20* SQ

Don't Limit

```
CREATE VIEW ottawa_hydrant_violations_master_view AS
    SELECT * FROM ottawa_hydrant_violations_2008 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2009 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2010 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2011 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2012 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2013 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2014;

    SELECT COUNT(*)
    FROM ottawa_hydrant_violations_master_view;

SELECT DATE_NEW,STREET,AMOUNTDUE
FROM ottawa_hydrant_violations_master_view;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

DATE_NEW	STREET	AMOUNTDUE
2008-01-02	LORNE AVE	45
2008-01-02	LORNE AVE	55
2008-01-02	LISGAR ST	45
2008-01-02	MAIN ST	55
2008-01-02	COOPER ST	45
2008-01-02	JAMES ST	91
2008-01-02	WILD SHORE CRES	91
2008-01-03	BRUYERE ST	45
2008-01-03	MANN AVE	45
2008-01-03	KING EDWARD AVE	55
2008-01-03	BRUYERE ST	45
2008-01-03	NORTH RIVER	27.5
2008-01-03	IN FRONT OF 182	55

42. To sort the amounts in descending order, we need to add and “ORDER BY” phrase to our query.

```
CREATE VIEW ottawa_hydrant_violations_master_view AS
    SELECT * FROM ottawa_hydrant_violations_2008 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2009 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2010 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2011 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2012 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2013 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2014;

    SELECT COUNT(*)
    FROM ottawa_hydrant_violations_master_view;

SELECT DATE_NEW,STREET,AMOUNTDUE
FROM ottawa_hydrant_violations_master_view
ORDER BY AMOUNTDUE DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

DATE_NEW	STREET	AMOUNTDUE
2011-07-16	PRIMROSE AVE	141
2012-09-06	NEPEAN ST	141
2008-06-23	NICHOLAS ST	126
2008-06-11	CUMBERLAND ST	126
2008-03-25	PETER MORAND CRES	126
2008-06-11	CUMBERLAND ST	126
2013-11-29	RICHMOND RD	121
2013-09-12	MCLEOD ST	106
2013-09-12	MCLEOD ST	106
2013-09-12	MCLEOD ST	106
2013-09-13	REGENT ST	106
2013-09-13	BRUYERE ST	106
2013-09-13	MCLEOD ST	106
2013-09-14	GILMOUR ST	106
2013-09-14	IN FRONT OF 124 CE...	106

ottawa_hydrant_violations_mas...

43. If you’re happy with this table, you can export it as a csv file, by selecting the “Export” tab also highlighted above.

44. In addition to straight-forward select queries, we might want to group certain fields (as we did in pivot tables) and then COUNT the number of tickets. To count the number of fines for each year, and then sort them in descending order, we would use the query highlighted below.

```
CREATE VIEW ottawa_hydrant_violations_master_view AS
    SELECT * FROM ottawa_hydrant_violations_2008 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2009 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2010 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2011 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2012 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2013 UNION ALL
    SELECT * FROM ottawa_hydrant_violations_2014;

    SELECT COUNT(*)
    FROM ottawa_hydrant_violations_master_view;

SELECT DATE_NEW,STREET,AMOUNTDUE
FROM ottawa_hydrant_violations_master_view
ORDER BY AMOUNTDUE DESC;

SELECT YEAR(DATE_NEW) AS Date, COUNT(AMOUNTDUE) AS Number
FROM ottawa_hydrant_violations_master_view
GROUP BY date
ORDER BY Number DESC;
```

Date	Number
2008	10930
2013	7431
2012	6522
2011	6167
2010	4995
2014	3124

45. It seems as though 2008 was the banner year for fire hydrant tickets. Let’s unpack this statement. We’re using

the “Year” function that we saw in Excel to pull the year out of the date field. The “AS” part of the statement indicates that we want to use an “alias” or a new word for that column, as we can see in the grid above. Then we want to count the number of fines, using the “COUNT” function that we’ve also seen in Excel. We are pulling these files from our master table or view that we’ve created, grouping the data by the date (the alias) and ordering the Number (the alias) in descending order. Please use pages 183 to 200 of Computer-Assisted Reporting as a handy reference.

46. If you’re happy with this result, you can also export it, and then keep going with new queries in the same query tab, making sure to save each one as a csv file.

ADDITIONAL INFORMATION CONCERNING MYSQL

MYSQL queries can be complicated. However, they are always built upon the same basic groundwork that we’ve discussed in the text book, and which we can see in the sections to follow. The first section is based upon the language conventions in MYSQL.

LANGUAGE CONVENTIONS

In MYSQL, names of created objects and keywords, as well as a variety of other objects, each have distinct requirements in their names.

A query is a single bit of MYSQL code. A query is finished with a semicolon (;)

A Script is anywhere from one to many queries. Generally, a script has a specific purpose. Any of the complete files given with this could be called a script.

A Clause is a portion of a MYSQL script. For example, in SELECT * FROM table_name WHERE x = 1, there are 3 clauses: SELECT, FROM, and WHERE.

Created Objects and Column Names

Cannot Contain Spaces (generally, underscores are used “table_name”)

Strings

Whenever you input a string, or a sentence or words, etc., they must be surrounded by single quotes (‘person’s name’), or double quotes (“person’s name”)

Keywords

Keywords are any words recognized by MYSQL. This includes things such as SELECT, CREATE, UPDATE, DELETE

MYSQL does not require keywords to be in all caps, though they are for this document, to distinguish them