Module 4
Processing MOVES Output in MySQL

EPA

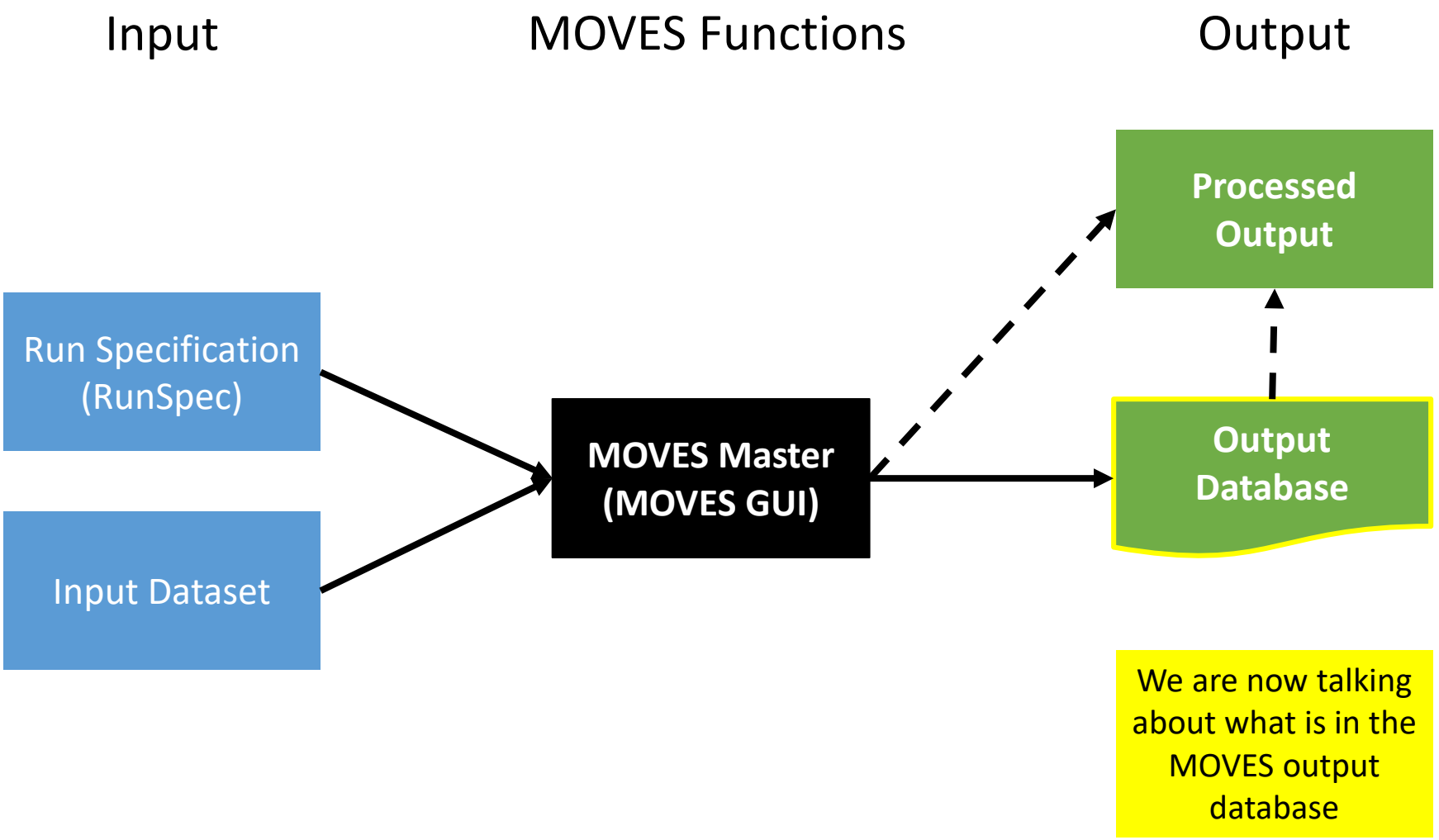
Module Overview

- Describe what is contained in the MOVES output tables
- Use the Post-Processing Menu
- Review MOVES inputs with MySQL Workbench
- View and manipulate MOVES output using MySQL Workbench
- Compare the results of the MOVES National Scale and County-scale inventory exercises using MySQL Workbench

MOVES
Motor Vehicle Emission Simulator

EPA

Using MOVES: Where Are We?



MOVES
Motor Vehicle Emission Simulator

EPA

MOVES Output: General

- All of the results of a MOVES run are stored in MySQL database tables

- These results can be accessed by
  - Using MOVES summary reporter
  - Using MySQL query commands and/or the MySQL Workbench
  - Using Microsoft Access with a MySQL Open Database Connectivity (ODBC)

- Any table may be exported to other applications (e.g, MS Excel) for further processing

MOVES
Motor Vehicle Emission Simulator

EPA

Output Database Tables

- The MOVES output database contains numerous output tables with results of the run, input data, and other information about the run

- MOVESOutput table
  - Contains the quantity of emissions (by sourcetype, pollutant/process, etc., based on output detail selections made in the RunSpec)

- MOVESActivityOutput table
  - Contains measurements of activity (e.g. VMT, population, etc., based on output detail selections made in the runspec)—useful to ensure no VMT was "lost"

- MOVESRun table
  - Information about the run (e.g., date/time of run, domain and scale, units selected)

MOVES
Motor Vehicle Emission Simulator

EPA

Output Database Tables

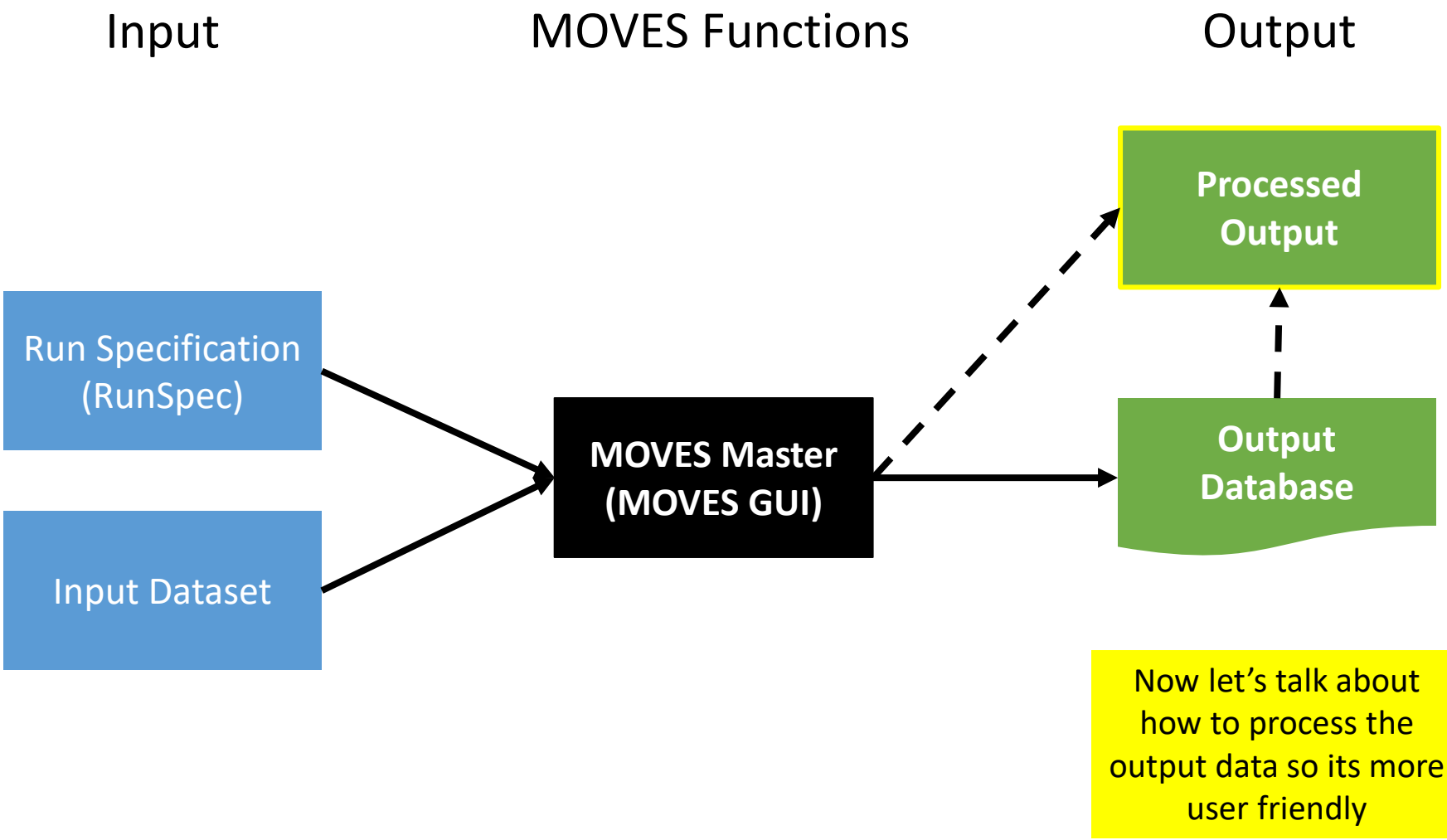
- Some tables are only populated when doing an Emission Rates run (not relevant to Inventory run)
  - RatePerDistance
  - RatePerVehicle
  - RatePerProfile
  - RatePerStart
  - RatePerHour
  - StartsPerVehicle

- Some tables are useful for diagnostic purposes
  - ActivityType
  - MOVESError
  - MOVESTablesUsed
  - MOVESWorkersUsed
  - MOVESEventLog

MOVES
Motor Vehicle Emission Simulator

EPA

Using MOVES: Where Are We?



MOVES
Motor Vehicle Emission Simulator

EPA

MOVES Post-Processing Menu

- Use this menu option only after you have completed a run

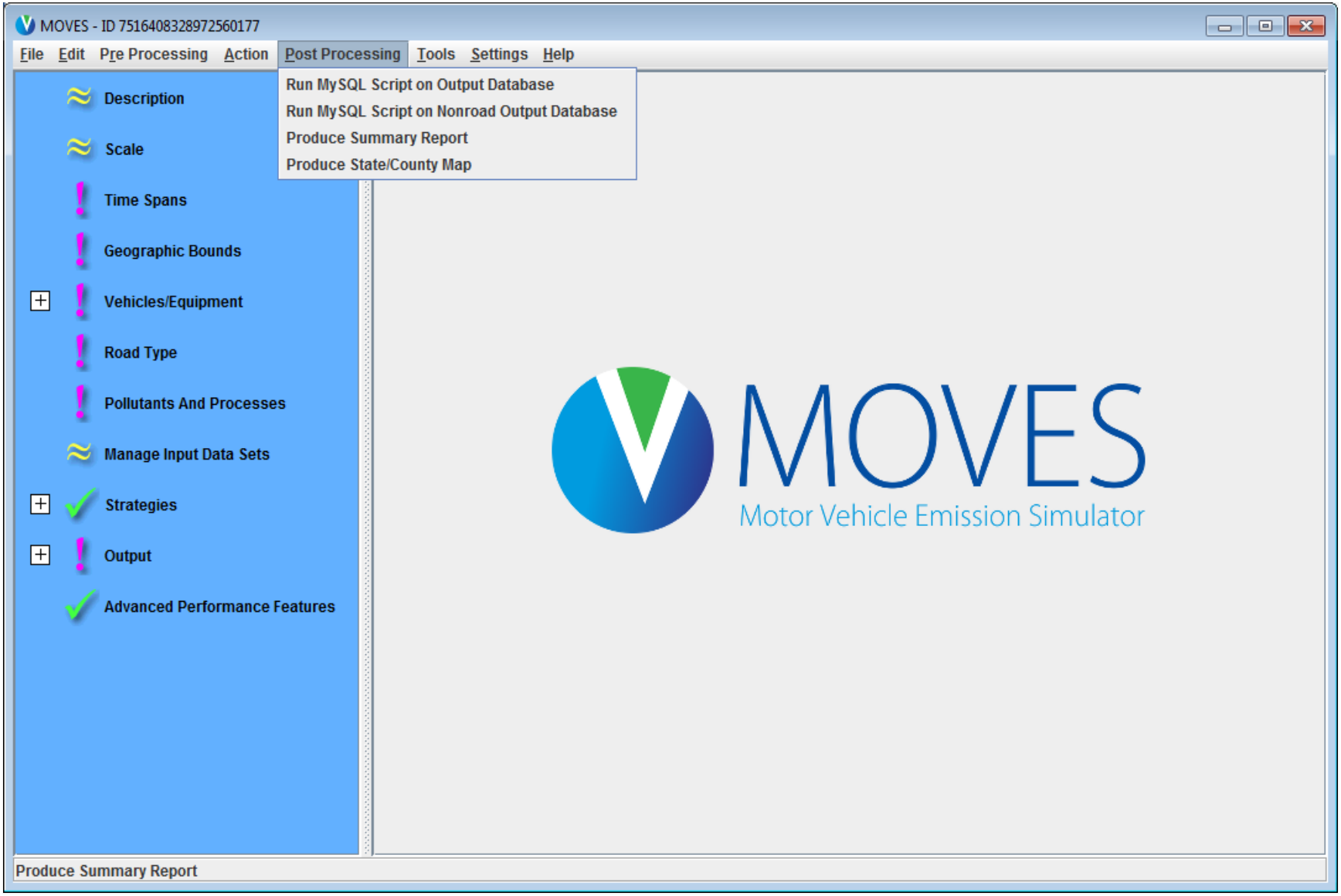
- Select an existing output database using the RunSpec used to generate the results
  - If you are interesting in doing any post-processing from the Post-Processing Menu, it's often easiest if you immediately do so upon conclusion of the run

- Options for processing output include
  - Execute any MySQL scripts that come embedded in MOVES
  - Summarize results into text files
  - Graphically represent results in a county map

MOVES
Motor Vehicle Emission Simulator

EPA

MOVES Post-Processing Menu



Post-Processing Scripts

- The scripts are applied to the current output database selected in the RunSpec
- You can view how many runs are stored in the output database using the MOVES Run Error Log window from the pull down Action menu
- There are several MySQL command scripts stored in the /database/OutputProcessingScripts folder of the MOVES application installation
- Users may write their own scripts and add them to the folder or add scripts obtained from other users

MOVES
Motor Vehicle Emission Simulator

EPA

Post-Processing Scripts

- Read the script documentation before running MOVES
  - Project-scale scripts may require that you run MOVES in a particular way
  - Scripts may require running with a specific calculation type or certain units in the output

- Send ideas for useful scripts to mobile@epa.gov

MOVES
Motor Vehicle Emission Simulator

EPA

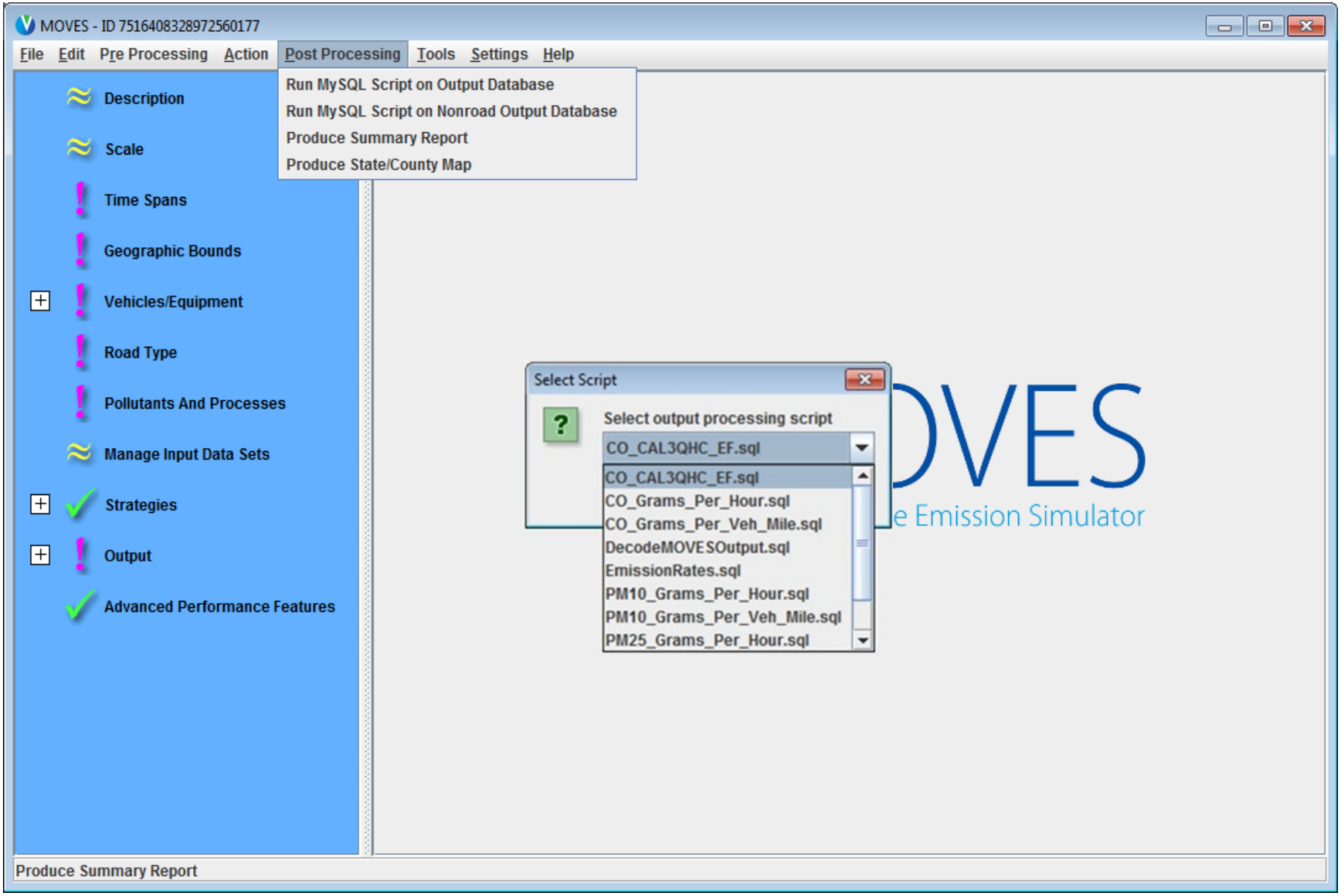
Post-Processing Scripts

Script Title	Description
DecodeMOVESOutput.sql	Decodes most key fields of MOVESOutput and MOVESActivityOutput tables
EmissionRates.sql	Produces an output table which reports the emission results in units of mass per distance
TabbedOutput.sql	Produces tab-delimited output suitable for reading into an EXCEL Spreadsheet from the MOVES MySQL database output tables

Post-Processing Scripts: Project Scale

Script Title	Description
CO_CAL3QHC_EF.sql	Produces CO emission rates for use in the CAL3QHC air quality model
CO_Grams_Per_Hour.sql	Produces CO emission rates as grams per hour for each link (project-scale runs)
CO_Grams_Per_Veh_Mile.sql	Produces CO emission rates as grams per vehicle-mile for each link (project-scale runs)
PM10_Grams_Per_Hour.sql	Produces PM10 emission rates as grams per hour for each link (project-scale runs)
PM10_Grams_Per_Veh_Mile.sql	Produces PM10 emission rates as grams per vehicle-mile for each link (project-scale runs)
PM25_Grams_Per_Hour.sql	Produces PM2.5 emission rates as grams per hour for each link (project-scale runs)
PM25_Grams_Per_Veh_Mile.sql	Produces PM2.5 emission rates as grams per vehicle-mile for each link (project-scale runs)

Selecting a MySQL Output Processing Script



Post-Processing Menu: Summary Report

- Uses the output tables in the database referenced in the current RunSpec
- Reports output emissions and activity in varying levels of detail, based on selections by the user
- Covered in detail in Module 2

MOVES
Motor Vehicle Emission Simulator

EPA

Other Post-Processing Options

- Post process using MySQL script(s)

- Export the data using MySQL Workbench
  - CSV
  - MS Excel
  - PLIST

- Use the data from MS Access / R / Python / etc. using the ODBC or other MySQL connectors (not covered here)

MOVES
Motor Vehicle Emission Simulator

EPA

MySQL Workbench

- Should have been installed with MySQL Server
  - If not, it can be added using the MySQL Community Installer
- It is a tool for viewing databases, executing queries, and editing tables
- Results can be exported as .csv
- Queries are written as scripts, so you can repeat them without retyping
- Tables can be edited directly, rather than using MySQL commands

Exploring MOVES Databases with MySQL

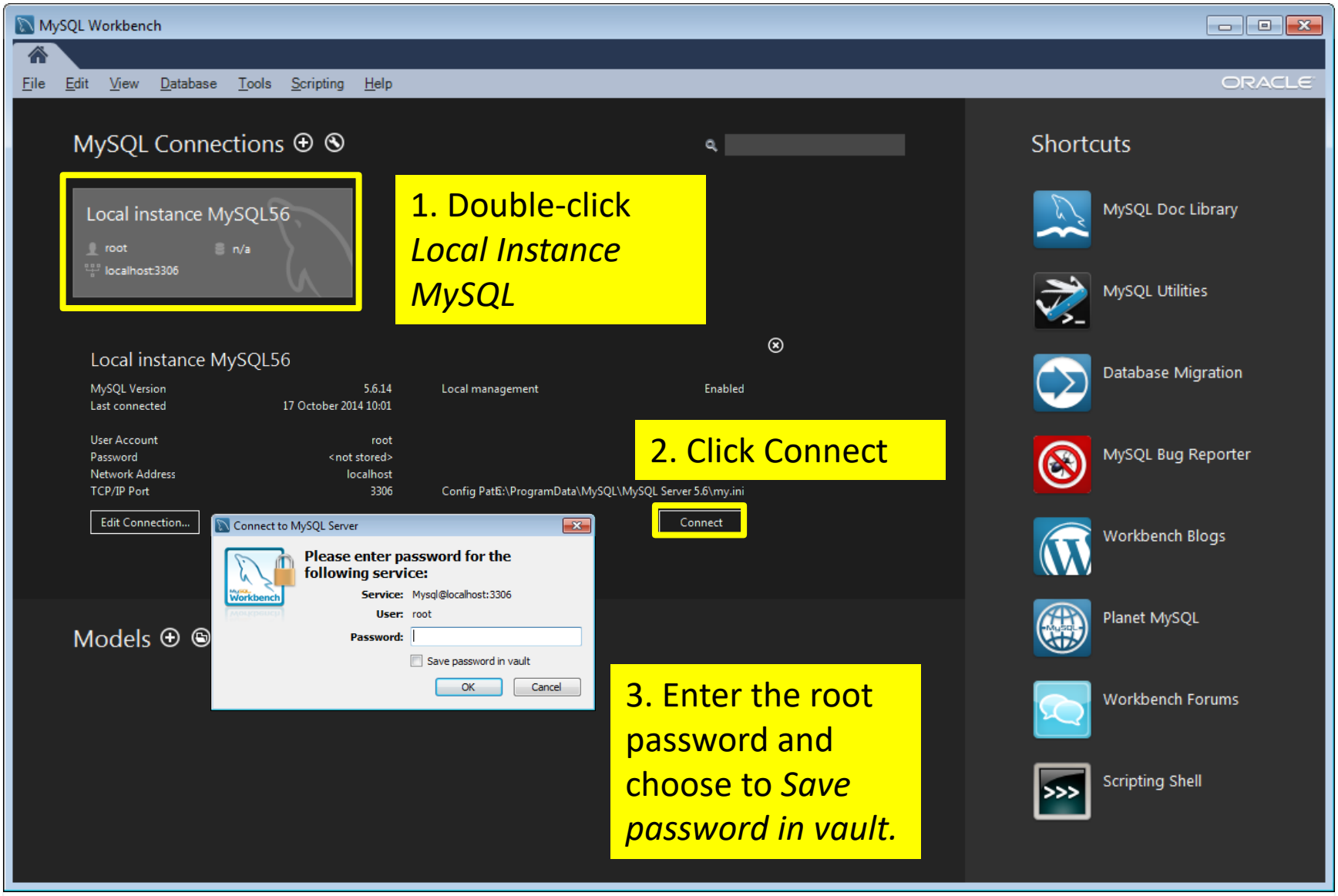
- Introduction to Workbench and hands-on practice using our Module 2 National-scale inventory database and Module 3 County-scale input and output databases

- Instructions for Exploring MOVES Databases Exercise:
  - Open MySQL Workbench
  - Start/Programs/MySQL/MySQL Workbench
  - Click *Local instance MySQL*
  - Click *Connect*
  - If this is your first time in Workbench, you may need to enter the root user password after clicking *Connect*
    - This password was set when MySQL was installed
    - If you do not know this password, right-click on *Local instance MySQL*, select *Edit Configuration…,* set Username to **`moves`**, and click *Close*. When prompted for a password, enter **`moves`**

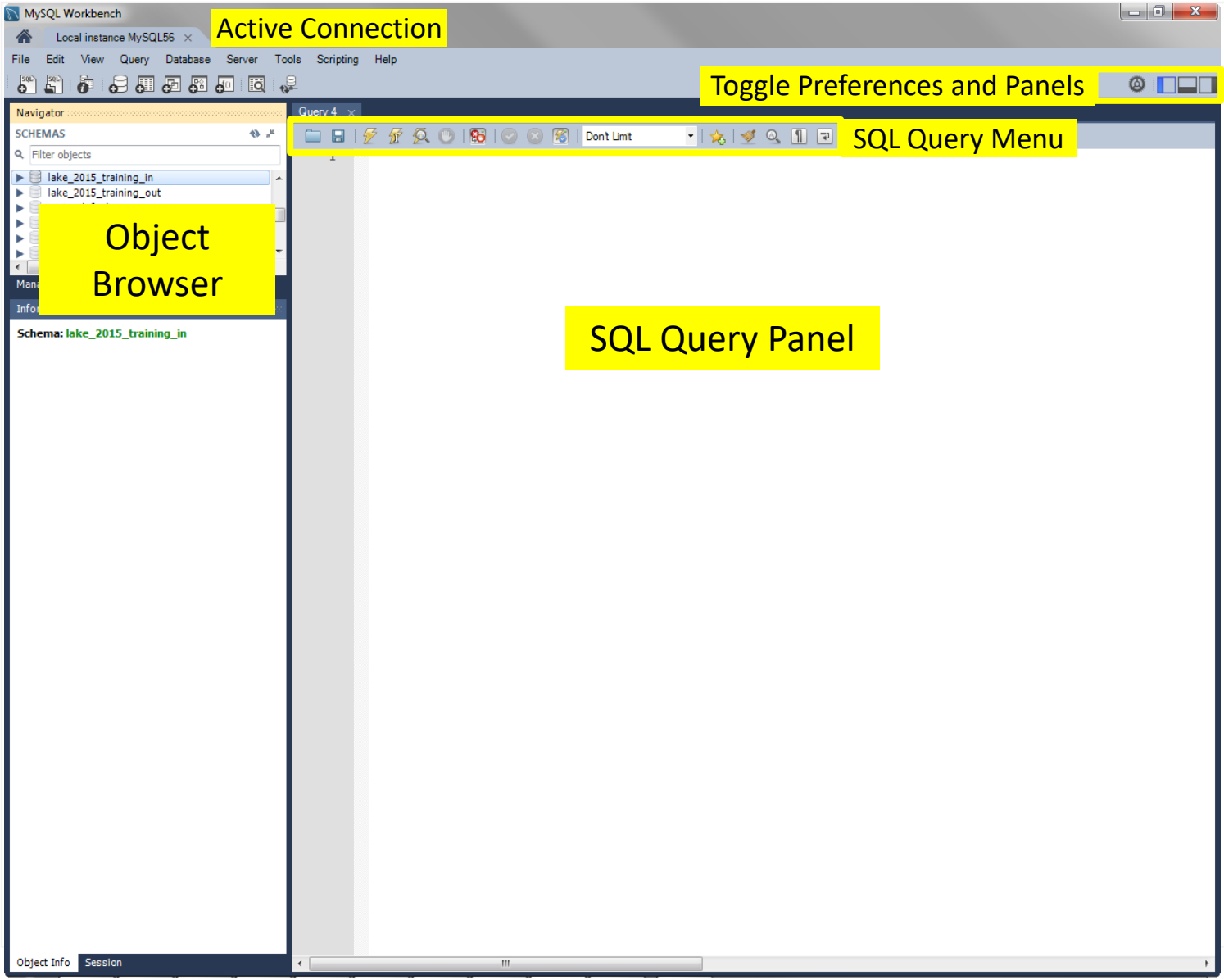
MOVES
Motor Vehicle Emission Simulator

EPA

MySQL Workbench: Getting Started



MySQL Workbench: Layout



MySQL Workbench: Basics

From left to right, these buttons are:

Open a SQL Script File: Loads a saved SQL script to be ready for execution.

Save SQL Script to File: Saves the current SQL script to a specified file.

Execute SQL Script: Executes the selected portion of the query, or the entire query if nothing is selected.

Execute Current SQL script: Execute the statement under the keyboard cursor.

Explain: Execute the **EXPLAIN** command on the query after the keyboard cursor.

Stop the query being executed: Halts execution of the currently executing SQL script.

Toggle whether execution of SQL script should continue after failed statements: If the red "breakpoint" circle is displayed, the script terminates on a statement that fails. If the button is depressed so that the green arrow is displayed, execution continues past the failed code, possibly generating additional result sets.

Commit: Commits the current transaction. Note: All query tabs in the same connection share the same transactions. To have independent transactions, a new connection must be opened.

Rollback: Rolls back the current transaction. Note: All query tabs in the same connection share the same transactions. To have independent transactions, a new connection must be opened.

Toggle Auto-Commit Mode: If selected, each statement will be committed independently. Note: All query tabs in the same connection share the same transactions. To have independent transactions, a new connection must be opened.

Set row limit (drop down): Allows the user to specify the return row limit on the result set

Snippets: Allows the user to select to save portions of queries for later use.

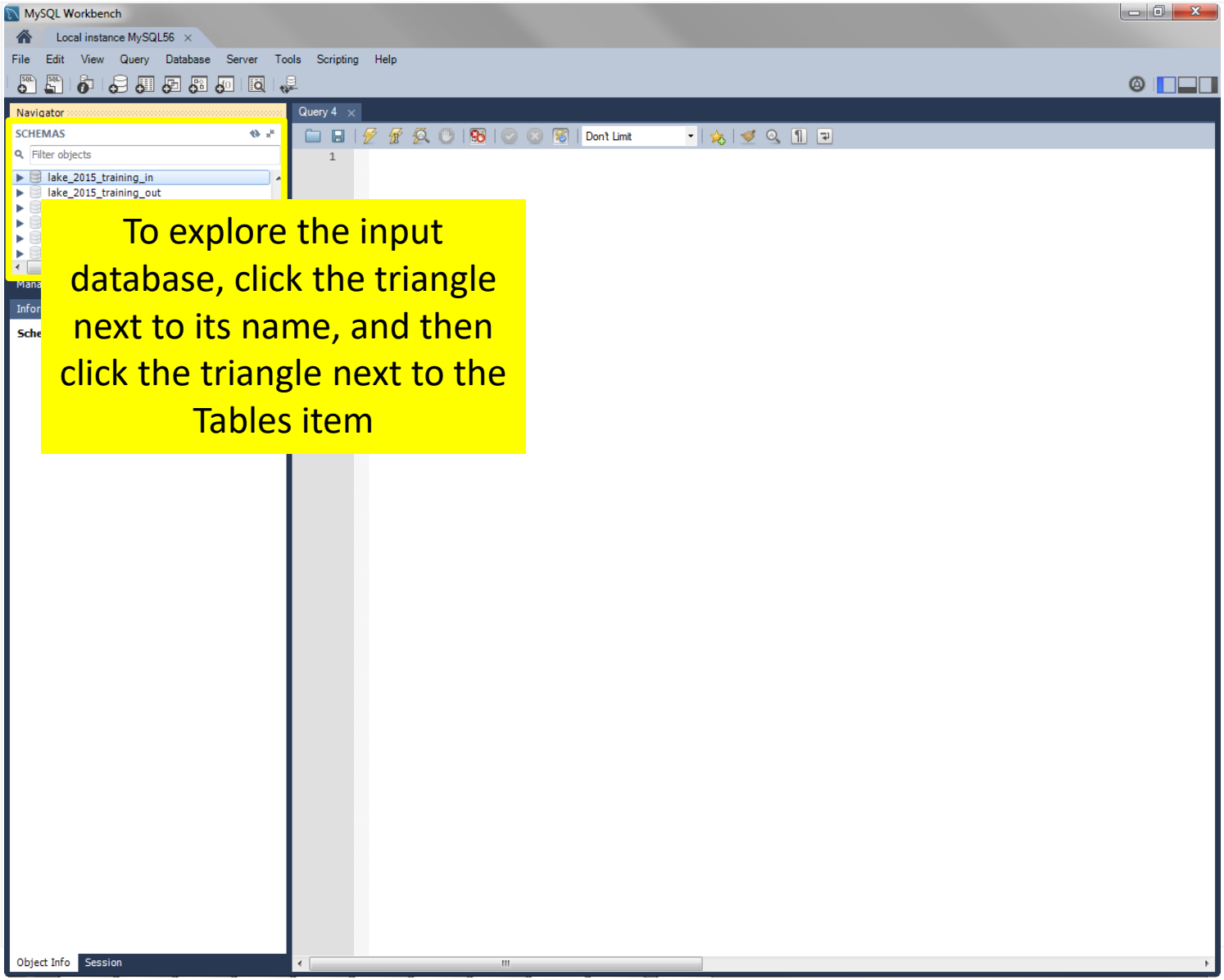
Beautify SQL: Beautify/reformat the SQL script.

Find panel: Show the Find panel for the editor.

Invisible characters: Toggle display of invisible characters, such as newlines, tabs or spaces.

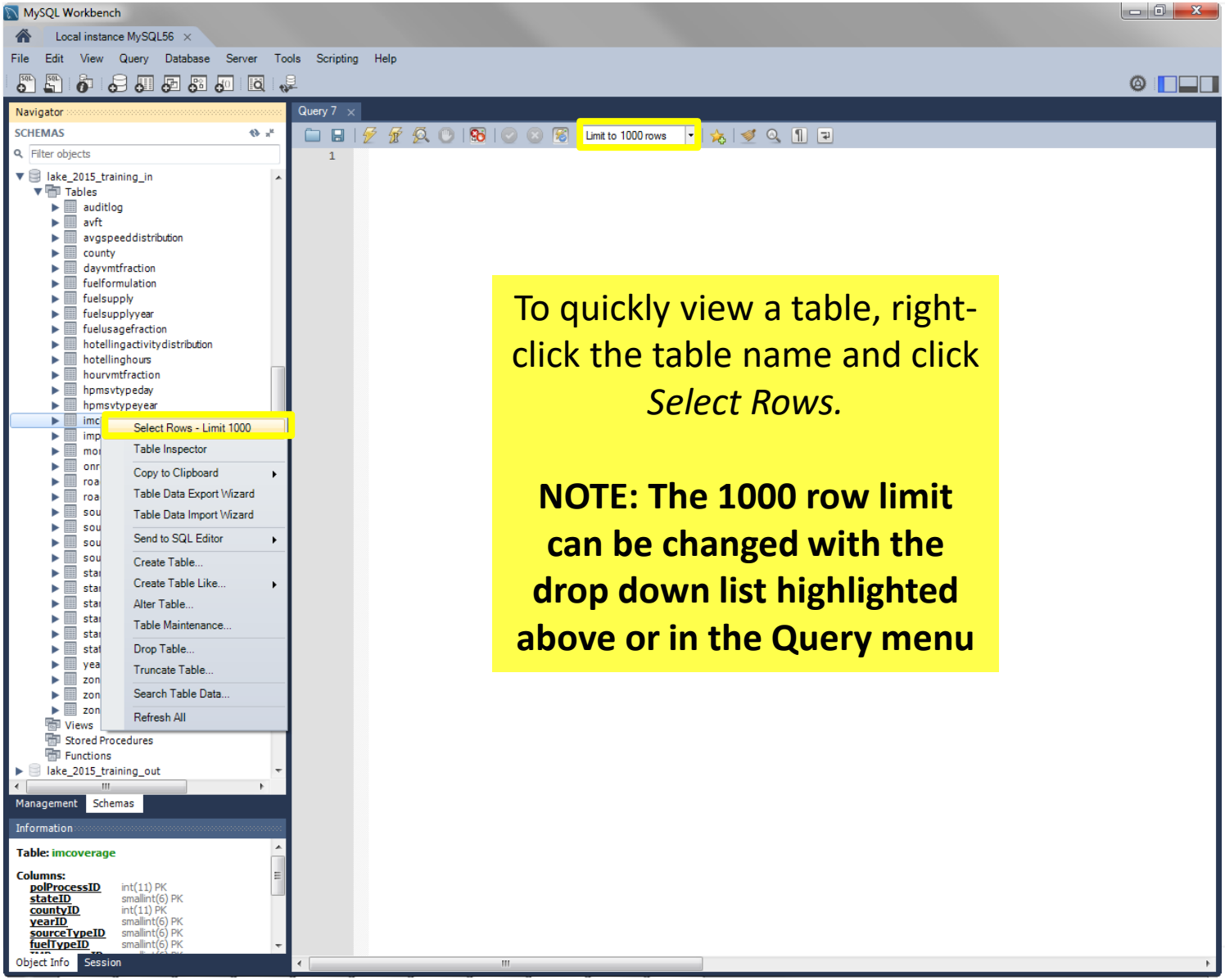
Wrapping: Toggles the wrapping of long lines in the SQL editor window.

MySQL Workbench: Basics



To explore the input database, click the triangle next to its name, and then click the triangle next to the Tables item

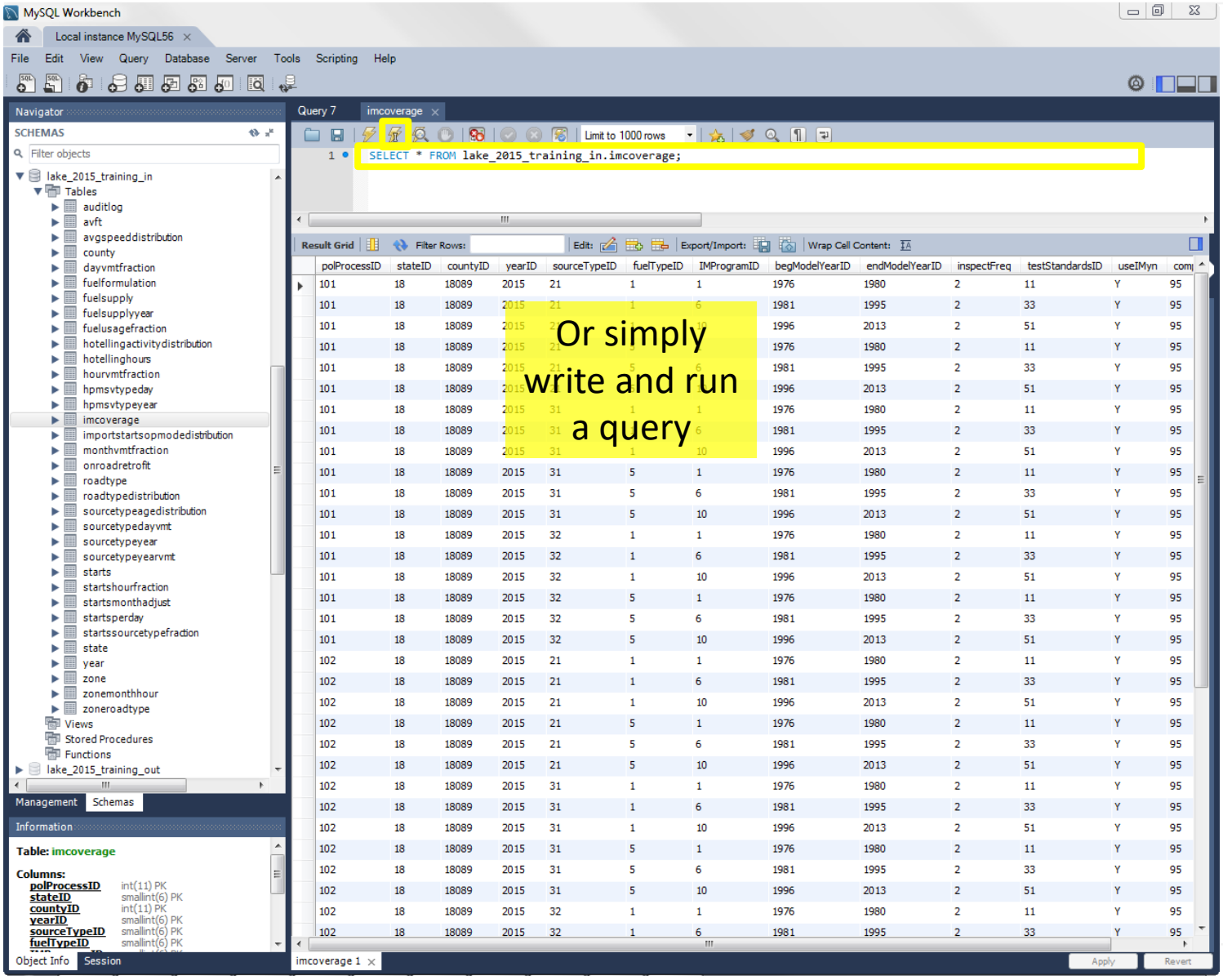
MySQL Workbench: Basics



To quickly view a table, right-click the table name and click *Select Rows.*

NOTE: The 1000 row limit can be changed with the drop down list highlighted above or in the Query menu

MySQL Workbench: Basics



MySQL Queries: Syntax

- Queries are used to display or aggregate output database data
- A query can be typed into the Query Area
- **The order and arrangement of query commands are important!**
  - Use the "**Introduction to MYSQL Workbench Syntax**" handout.
  - Syntax must be used in the order given on the handout/next slide.
  - Not all commands are needed to complete a query
  - To identify the table to be queried from, the syntax is database name followed by "." and the table name
  - Commas can be used to separate multiple fields following a command
  - All of this will be covered more on the next few slides
- Click the  button or hit CTL/ENTER to execute queries
- NOTE: All the queries in this module can be copied and pasted from the text file called *postprocessing.sql* in the County Inventory Exercise folder

MOVES
Motor Vehicle Emission Simulator

EPA

MySQL Queries: Commands

Syntax	Function	Example
SELECT	Selects one or more data fields, separated by commas. A "*" following the SELECT command indicates "all fields"	SELECT MOVESRunID, sourceTypeID
SUM	Adds up the data in the field indicated in parentheses (note required spacing)	SUM(activity)
AS	Used with the SUM command to name the results of the SUM command (optional)	AS vmt
FROM	Indicates the database and table the SELECT command is pulling from. Database and table must be separated by period	FROM lake_2015_training_out.movesactivityoutput
WHERE	Used to specify the value(s) of the field to be selected	WHERE activityTypeID = 1
AND	Used to specify more than one field when using the WHERE command	WHERE activityTypeID = 1 AND dayID = 2
GROUP BY	Groups data together by the field(s) indicated **(if using aggregation, all non-aggregated columns must be listed here!)**	GROUP BY MOVESRunID, sourceTypeID
ORDER BY	Specifies the order of data presented in the field(s) following the command	ORDER BY sourceTypeID, MOVESRunID

MOVES
Motor Vehicle Emission Simulator

EPA

Building MySQL Queries

- Start with a simple select all command:

```
SELECT *
FROM lake_2015_training_out.movesoutput;
```

- In natural language, this query means:
  - Select all columns in the *movesoutput* table of the *lake_2015_training_output* database

- Here are the results:

MOVESRunID	iterationID	yearID	monthID	dayID	hourID	stateID	countyID	zoneID	linkID	pollutantID	processID	sourceTypeID	regClassID	fuelTypeID
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	32	NULL	5
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	31	NULL	5
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	21	NULL	5
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	42	NULL	2
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	32	NULL	2
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	31	NULL	2
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	21	NULL	2
1	1	2015	7	5	24	18	18089	NULL	NULL	1	19	32	NULL	1

MOVES
Motor Vehicle Emission Simulator

EPA

Building MySQL Queries

- Select only the columns of interest:

```
SELECT MOVESRunID, hourID, pollutantID, processID,
       sourceTypeID, fuelTypeID, roadTypeID, emissionQuant
FROM lake_2015_training_out.movesoutput;
```

- In natural language, this query means:
  - Select the run, hour, pollutant, process, source type, fuel type, road type, and emissionQuant columns in the *movesoutput* table of the *lake_2015_training_output* database
- Here are the results:

MOVESRunID	hourID	pollutantID	processID	sourceTypeID	fuelTypeID	roadTypeID	emissionQuant
1	24	1	19	32	5	5	0.130985
1	24	1	19	31	5	5	0.551499
1	24	1	19	21	5	5	0.188705
1	24	1	19	42	2	5	2.41674
1	24	1	19	32	2	5	3.83791
1	24	1	19	31	2	5	5.26899
1	24	1	19	21	2	5	1.45448
1	24	1	19	42	1	5	0.0524875
1	24	1	19	32	1	5	56.8643
1	24	1	19	31	1	5	232.671

MOVES Motor Vehicle Emission Simulator

EPA

Building MySQL Queries

- Filter the data to only look at running emissions:

```
SELECT MOVESRunID, hourID, pollutantID, processID,
       sourceTypeID, fuelTypeID, roadTypeID, emissionQuant
FROM lake_2015_training_out.movesoutput
WHERE processID = 1;
```

- In natural language, this query means:
  - Select the run, hour, pollutant, process, source type, fuel type, road type, and emissionQuant columns in the *movesoutput* table of the *lake_2015_training_output* database
  - Filter the results where processID = 1 (i.e. where the rows contain running emissions)
- Here are the results:

MOVESRunID	hourID	pollutantID	processID	sourceTypeID	fuelTypeID	roadTypeID	emissionQuant
1	24	1	1	32	5	5	0.472404
1	24	1	1	31	5	5	2.12184
1	24	1	1	21	5	5	0.613188
1	24	1	1	42	3	5	154.043
1	24	1	1	42	2	5	76.6988
1	24	1	1	32	2	5	225.16
1	24	1	1	31	2	5	263.598

MOVES
Motor Vehicle Emission Simulator

EPA

Building MySQL Queries

- Aggregate over all hours of the day:

```
SELECT MOVESRunID, pollutantID, processID, sourceTypeID,
       fuelTypeID, roadTypeID, SUM(emissionQuant)
FROM lake_2015_training_out.movesoutput
WHERE processID = 1
GROUP BY MOVESRunID, pollutantID, processID, sourceTypeID,
         fuelTypeID, roadTypeID;
```

- In natural language, this query means:
  - Select the run, pollutant, process, source type, fuel type, and road type columns, as well as the sum total of emissionQuant column in the *movesoutput* table of the *lake_2015_training_output* database
  - Filter the results where processID = 1 (i.e., running emissions)
  - Group the results by run, pollutant, process, source type, fuel type, and road type (i.e. report the summed emissionQuant by these columns)

Note: When more than one pollutant was selected in the RunSpec, a*lways* SELECT and GROUP BY pollutantID so that you do not SUM different pollutants

MOVES Motor Vehicle Emission Simulator

EPA

Building MySQL Queries

- Here are the results from the previous query:

MOVESRunID	pollutantID	processID	sourceTypeID	fuelTypeID	roadTypeID	SUM(emissionQuant)
1	1	1	21	1	2	12618.516464233398
1	1	1	21	1	3	13714.356651306152
1	1	1	21	1	4	129490.98303222656
1	1	1	21	1	5	224125.78424072266
1	1	1	21	2	2	76.12213844060898
1	1	1	21	2	3	82.55443170666695
1	1	1	21	2	4	778.6520645618439
1	1	1	21	2	5	1341.031759262085
1	1	1	21	5	2	2.59324908349663
1	1	1	21	5	3	2.5133802900090814
1	1	1	21	5	4	25.571258798241615
1	1	1	21	5	5	37.29305297136307
1	1	1	21	1	2	18260.441764831543

MOVES
Motor Vehicle Emission Simulator

EPA

More MySQL Queries

- Aggregate over fuel type and road type, too:

```
SELECT MOVESRunID, pollutantID, sourceTypeID, SUM(emissionQuant)
FROM lake_2015_training_out.movesoutput
WHERE processID = 1
GROUP BY MOVESRunID, pollutantID, sourceTypeID;
```

- In natural language, this query means:
  - Select the run, pollutant, and source type columns, as well as the sum total of emissionQuant column in the *movesoutput* table of the *lake_2015_training_output* database
  - Filter the results where processID = 1 (i.e., running emissions)
  - Group the results by run, pollutant, and source type (i.e. report the summed emissionQuant by these columns)

More MySQL Queries

- Here are the results from the previous query:

MOVESRunID	pollutantID	sourceTypeID	SUM(emissionQuant)
1	1	21	382295.9717236031
1	1	31	513701.49523668736
1	1	32	141017.13435490057
1	1	42	23020.548599638045
2	1	21	42483.65460957378
2	1	31	50404.01205980808
2	1	32	61197.036043951375
2	1	42	736.0118490145542

MOVES
Motor Vehicle Emission Simulator

EPA

More MySQL Queries

- Order by source type:

```
SELECT MOVESRunID, pollutantID, sourceTypeID, SUM(emissionQuant)
FROM lake_2015_training_out.movesoutput
WHERE processID = 1
GROUP BY MOVESRunID, pollutantID, sourceTypeID
ORDER BY sourceTypeID, MOVESRunID, pollutantID;
```

- In natural language, this query means:
  - Select the run, pollutant, and source type columns, as well as the sum total of emissionQuant column in the *movesoutput* table of the *lake_2015_training_output* database
  - Filter the results where processID = 1 (i.e., running emissions)
  - Group the results by run, pollutant, and source type (i.e. report the summed emissionQuant by these columns)
  - Order the results first by source type, then run, and finally pollutant

More MySQL Queries

- Here are the results from the previous query:

MOVESRunID	pollutantID	sourceTypeID	SUM(emissionQuant)
1	1	21	382295.9717236031
2	1	21	42483.65460957378
1	1	31	513701.49523668736
2	1	31	50404.01205980808
1	1	32	141017.13435490057
2	1	32	61197.036043951375
1	1	42	23020.548599638045
2	1	42	736.0118490145542

MOVES
Motor Vehicle Emission Simulator

EPA

More MySQL Queries

- That's quite a difference between runs! Let's look at the total running inventory difference between the runs:

```
SELECT MOVESRunID, pollutantID, SUM(emissionQuant)
FROM lake_2015_training_out.movesoutput
WHERE processID = 1
GROUP BY MOVESRunID, pollutantID;
```

- In natural language, this query means:
  - Select the run, pollutant, and the sum total of emissionQuant column in the *movesoutput* table of the *lake_2015_training_output* database
  - Filter the results where processID = 1 (i.e., running emissions)
  - Group the results by run and pollutant (i.e. report the summed emissionQuant by these columns only): reports total running emissions from each run
- The results:

MOVESRunID	pollutantID	SUM(emissionQuant)
1	1	1060035.149914829
2	1	154820.71456234777

MOVES Motor Vehicle Emission Simulator

EPA

Advanced MySQL Queries

- Let's examine the activity:

```
SELECT MOVESRunID, sourceTypeID, sum(activity) as vmt
FROM lake_2015_training_out.movesactivityoutput
WHERE activityTypeID = 1
GROUP BY MOVESRunID, sourceTypeID
ORDER BY sourceTypeID, MOVESRunID;
```

- In natural language, this query means:
  - Select the run, source type, and sum total activity column (call those results "vmt") from the *movesactivityoutput* table of the *lake_2015_training_output* database
  - Filter the results where activity type id = 1 (distance traveled)
  - Group the results by run and source type (i.e. report the summed VMT by these columns only)
- The results:

MOVESRunID	sourceTypeID	vmt
1	21	7150793.038978636
2	21	1418522.2080183732
1	31	5150135.843889236
2	31	790738.805012794
1	32	1310486.1984015107
2	32	790738.7306143299
1	42	16015.189692489803
2	42	10000.00192767568

MOVES
Motor Vehicle Emission Simulator

EPA

Advanced MySQL Queries

- Make sure that the VMT output is the same as VMT input
- This will require a more complex query because VMT was input by HPMS class, and VMT is output by source type
- We will need to tell MySQL how to map source type results to HPMS, which can be found in the default database's *sourceusetype:*

```
SELECT *
FROM movesdb20161117.sourceusetype;
```

sourceTypeID	HPMSVtypeID	sourceTypeName
21	25	Passenger Car
31	25	Passenger Truck
32	25	Light Commercial Truck
51	50	Refuse Truck
52	50	Single Unit Short-haul Truck
53	50	Single Unit Long-haul Truck
54	50	Motor Home
43	40	School Bus
42	40	Transit Bus
41	40	Intercity Bus
61	60	Combination Short-haul Truck
62	60	Combination Long-haul Truck
11	10	Motorcycle

MOVES
Motor Vehicle Emission Simulator

EPA

Advanced MySQL Queries

- In MySQL, if you want to use information stored in two different tables in the same query, you use the `JOIN` keyword
- The syntax is:

```
FROM table1
JOIN table2 USING (col1, col2)
```

- The `USING` keyword specifies what column(s) should be used to merge on (i.e. rows with the same value in the `USING` columns will be joined together)
- Any column in either table can be used in the SELECT statement, or performing SELECT * will select all columns in both tables:

```
SELECT *
FROM lake_2015_training_out.movesoutput
JOIN movesdb20161117.sourceusetype using(sourceTypeID);
```

MOVES
Motor Vehicle Emission Simulator

EPA

Advanced MySQL Queries

- Now that we have access to HPMS class, sum the VMT by run and HPMS using the `JOIN` command:

```
SELECT MOVESRunID, HPMSVtypeID, sum(activity) as vmt
FROM lake_2015_training_out.movesactivityoutput
JOIN movesdb20161117.sourceusetype using (sourceTypeID)
WHERE activityTypeID = 1
GROUP BY MOVESRunID, HPMSVtypeID;
```

- In natural language, this query means:
  - Select the run and HPMS columns, as well as the sum total of the activity column (call those results "vmt") from…
  - The *lake_2015_training_output.movesactivityoutput* table merged with the *movesdb20161117.sourceusetype* table on the source type column
  - Filter the results where activityTypeID = 1 (i.e. VMT rows)
  - Group the results by run and HPMS class (i.e. report VMT by these columns)

MOVES Motor Vehicle Emission Simulator

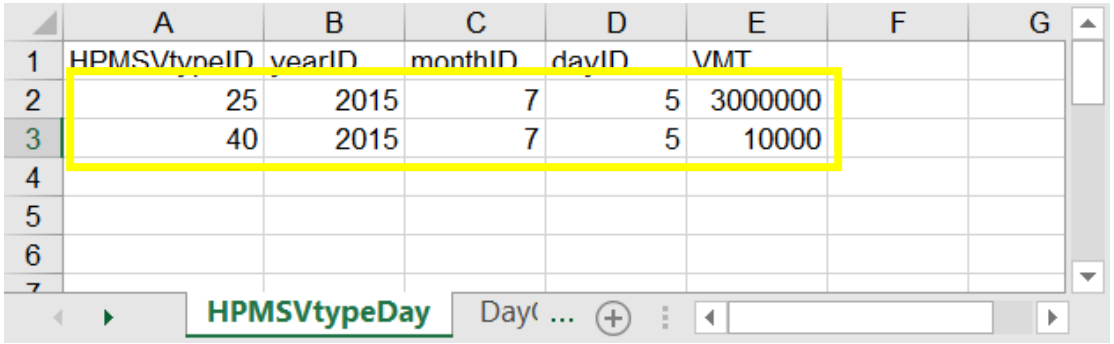
EPA

Advanced MySQL Queries

- Here are the results from the previous query:

MOVESRunID	HPMSVtypeID	vmt
1	25	13611415.081269383
1	40	16015.189692489803
2	25	2999999.7436454976
2	40	10000.00192767568

- Compare to the VMT input in the CDM (lake_vmt.xls from Module 3 shown below):



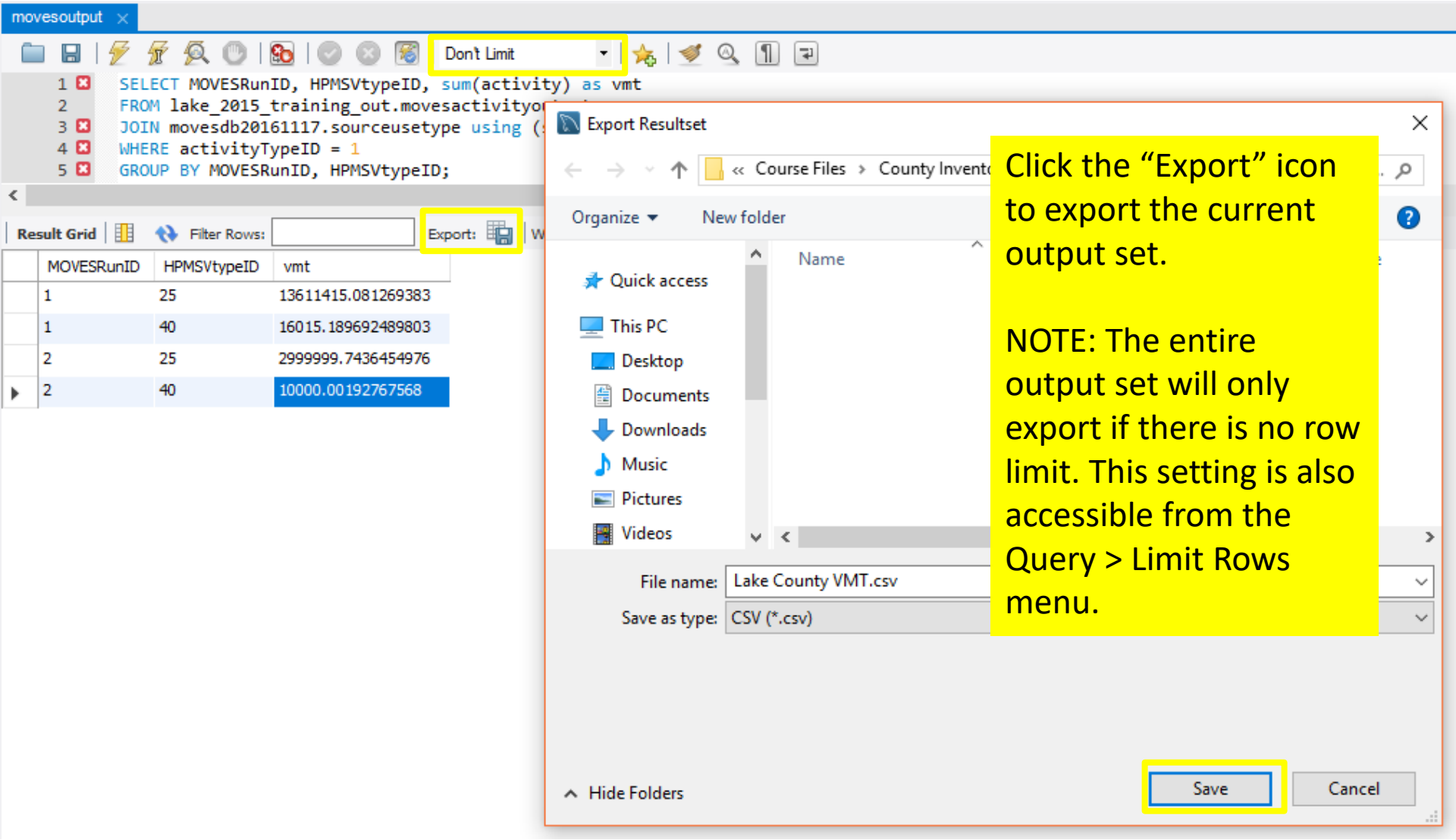
Other Workbench Actions

- Exporting result sets
- Saving scripts
- Sharing databases with others for review

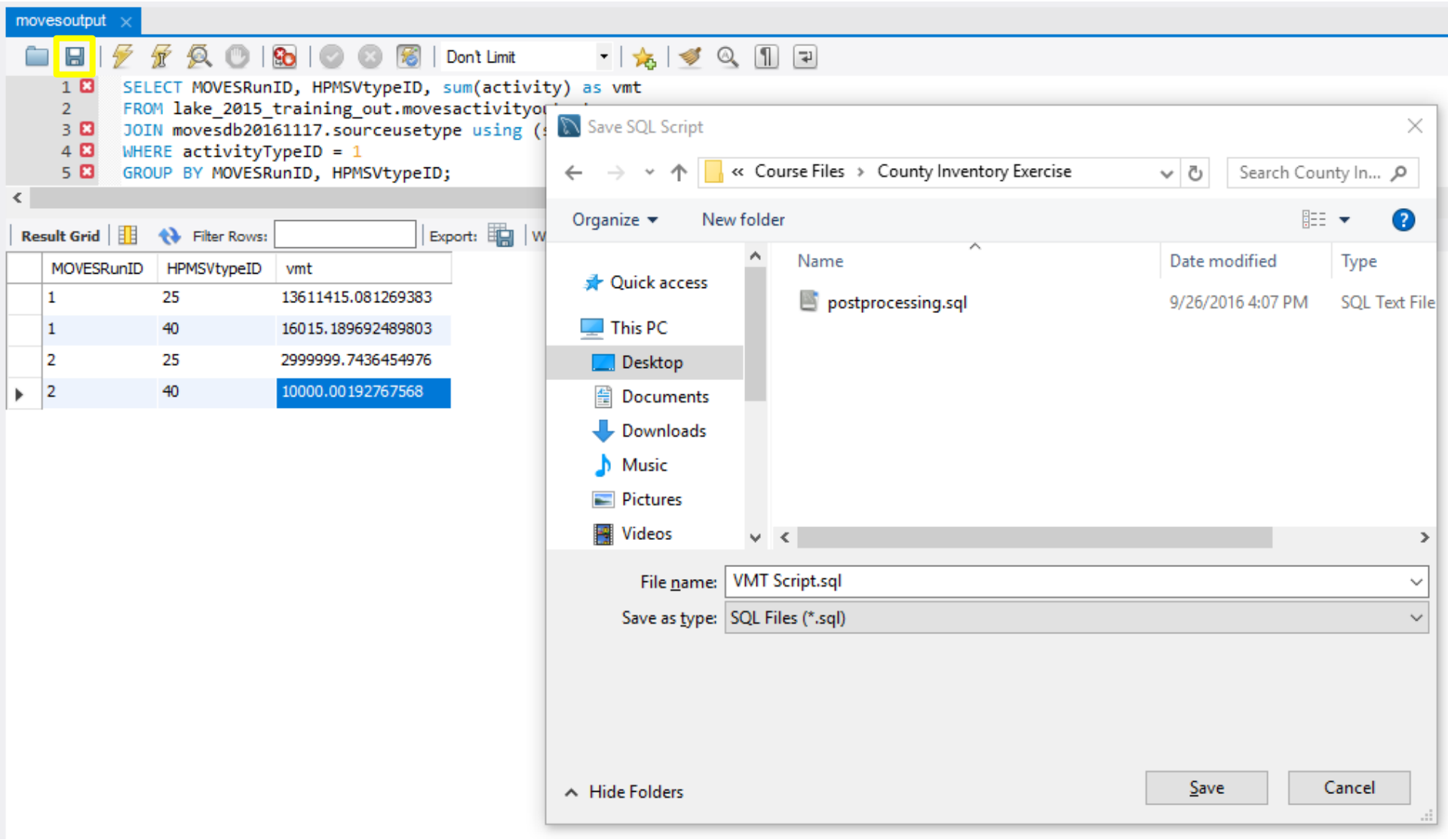
MOVES
Motor Vehicle Emission Simulator

EPA

Other Workbench Actions: Exporting

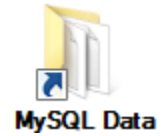


Other Workbench Actions: Saving Scripts



Copying and sending MySQL databases

- Input and output databases are stored in the data folder



- Databases may be copied and zipped for sharing and review

Questions?

EPA