

Introduction to MySQL

FHWA Resource Center
EPA Office of Transportation and Air Quality



EPA-420-B-09-023

Why Learn About MySQL?

- **Custom inputs**
 - MOVES has importers for most common types of inputs, but some MySQL expertise would be helpful for using specialized inputs
- **Working with output**
 - While MOVES can produce summary reports from the GUI, MySQL can be used to produce customized output

MySQL

- Relational Database Management System (RDBMS)
- What is a “relational database”?

Relational Databases 101

- A relational database provides a way of storing, querying, and manipulating large amounts of data
- A database contains multiple tables, in which data are stored. Each table contains records (rows) and fields (columns)
- These tables relate to each other based on values in a particular column

Relational Databases 101

- **Example: Excel spreadsheet**
 - Each “spreadsheet” is like a database
 - Each “worksheet” within the spreadsheet is like a table
 - However, the rules for database tables are more rigorous

Relational Database Rules

- Each table is a set of columns and rows that describe multiple instances of something
- Each field (column) must contain a unique type of data in a specific format
- Each row represents one unique record (tables cannot contain duplicate records)

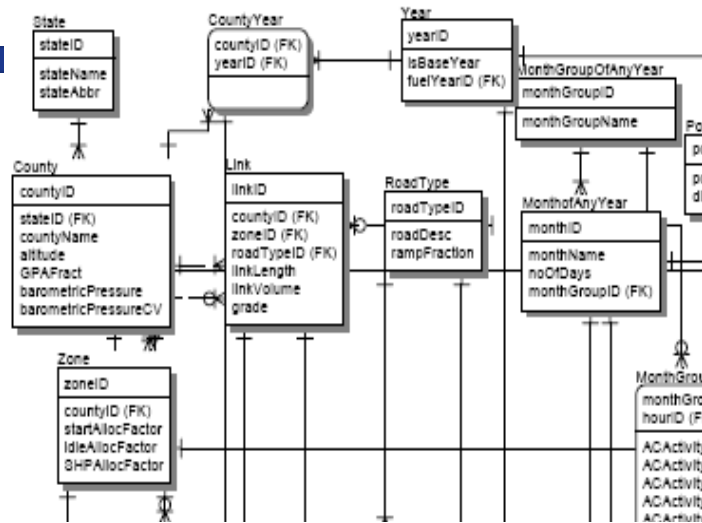
Relational Database Rules

- **Each table must contain at least one field with completely unique values to identify a record (row), and so that this record can be related to data in other tables (a key)**
 - Keys are usually numeric for flexibility and speed.
- **NULL values are allowed when data are not known or not relevant. Zeros are not the same as NULL.**
 - Keys cannot be NULL

MOVES Databases

- **The MOVES default database has over 100 different tables that store**
 - Lookup/reference information
 - Conversion/adjustment factors
 - Emissions data
 - Activity data
- **MOVES also uses databases to store user inputs, intermediate results and final output**

Entity-Relationship Diagrams



9

The symbols on the connecting lines indicate the type of relationship. The top fields (above the line) in the boxes are the key fields used in joining tables.

Types of Relationships

- **One to one**
 - Example: for each month there is only one month code, and for each month code there is only one month
- **One to many**
 - Example: one model year car can be sold in any of 12 months, but during any given month only one model year car is being sold
- **Many to many**
 - Example: any vehicle type can be of any age (any of the 30 age bins)

Structured Query Language (SQL)

- **SQL is the language used to work with modern relational databases**
 - It provides the programming syntax (language) to communicate with (query) the database and its tables
- **Like English, an SQL statement is made up of subjects, verbs, and predicates**
 - “SELECT * FROM state WHERE stateid = 32” means “Show me all the data from the “state” table where the state code in the table is 32.”

MOVES and MySQL

- **Draft MOVES2009 uses only MySQL version 5.0.27**
 - Newer or older versions will not work.
- **Support for MySQL is available in the help files, the reference manual (C:\MySQL\docs\manual.chm), and the MySQL web site (<http://www.mysql.com>)**
- **Entity/relationship diagrams for the MOVES default database are located at C:\MySQL\data\MOVES default database folder\readme**

MOVES and MySQL

- **Do you need expertise in relational databases and MYSQL?**
 - Not needed for simple runs
 - Some basic knowledge gives users flexibility to customize outputs and view input databases
 - In-house expert would be helpful for more advanced analysis
- **MySQL training is available from various commercial vendors.**

MySQL Query Browser

- Provided with the MOVES installation package.
- Windows tool for viewing databases, executing queries, and editing tables.
- Resultsets can be exported as csv or Excel files.
- Built-in Help files.
- Records query history, so you can repeat queries without retyping them.
- Tables can be edited directly, rather than using MySQL commands.
 - Table must have a key to be edited directly

MySQL Query Browser

The screenshot shows the MySQL Query Browser interface. The title bar indicates the connection is to @localhost:3306 / MOVESDB20070412. The query editor contains the text 'SELECT FROM avgSpeedBin;'. The 'Resultset 1' pane displays a table with three columns: 'avgSpeedBin...', 'avgBinSpeed', and 'avgSpeedBinDesc'. The table contains 16 rows of data. The 'Schemata' pane on the right shows a tree view of the database schema, including tables like 'agecategory', 'agegroup', 'averagetankgasoline', 'averagetanktemperature', 'avgSpeedBin', 'avgSpeedDistribution', 'coldsoakinitialfraction', and 'coldsoaktanktemperature'. The 'Syntax' pane shows a list of SQL keywords under the 'Data Manipulation' category, including DELETE, DO, HANDLER, INSERT, LOAD DATA INFILE, REPLACE, SELECT, Subquery, TRUNCATE, and UPDATE. The status bar at the bottom indicates '16 rows fetched in 0.0066s (0.0003s)' and provides navigation buttons like 'Edit', 'Apply Changes', 'Discard Changes', 'First', 'Last', and 'Search'.

avgSpeedBin...	avgBinSpeed	avgSpeedBinDesc
1	2.5	speed < 2.5mph
2	5	2.5mph <= speed < 7.5mph
3	10	7.5mph <= speed < 12.5mph
4	15	12.5mph <= speed < 17.5mph
5	20	17.5mph <= speed < 22.5mph
6	25	22.5mph <= speed < 27.5mph
7	30	27.5mph <= speed < 32.5mph
8	35	32.5mph <= speed < 37.5mph
9	40	37.5mph <= speed < 42.5mph
10	45	42.5mph <= speed < 47.5mph
11	50	47.5mph <= speed < 52.5mph
12	55	52.5mph <= speed < 57.5mph
13	60	57.5mph <= speed < 62.5mph
14	65	62.5mph <= speed < 67.5mph
15	70	67.5mph <= speed < 72.5mph
16	75	72.5mph <= speed

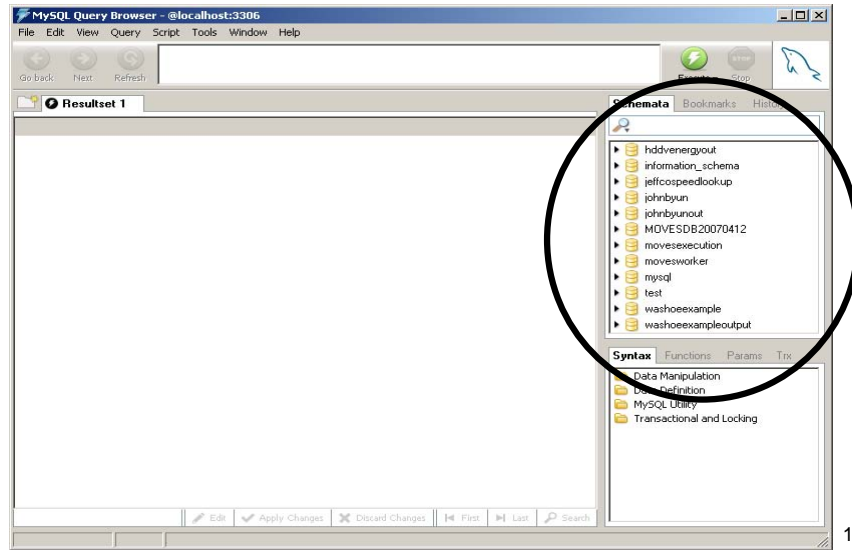
15

The Browser can be used to display the contents of the tables in a spreadsheet-like form.

Exploring MOVES Databases with Query Browser

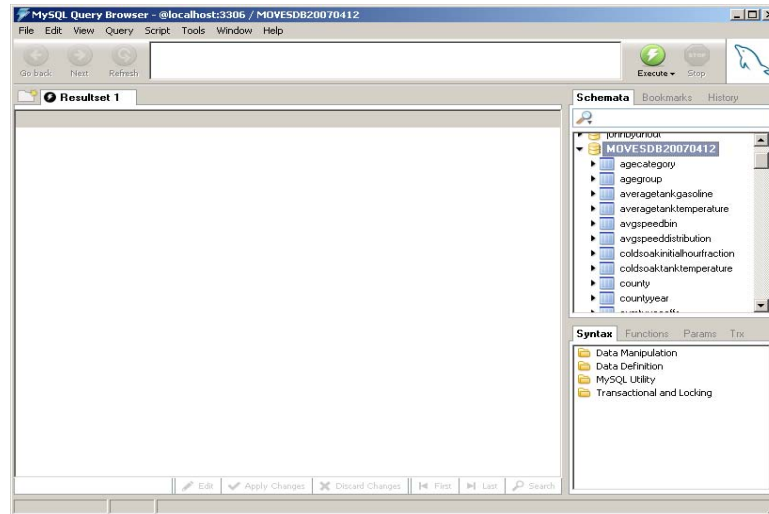
- **We'll demonstrate some simple MySQL commands in Query Browser.**
 - Commands are shown in the slide headings and screenshots.
 - Note that some screenshots show older versions of MOVES and/or output databases you won't have on your computer.
- **To Open Query Browser:**
 - Start/Programs/MySQL/MySQL Query Browser.
 - Make sure "localhost" is specified (might not be after initial installation) and click OK.
 - Click "ignore" on warning message about schema.
 - You can suppress this message for future use.

MySQL Query Browser opening view with available databases (example)



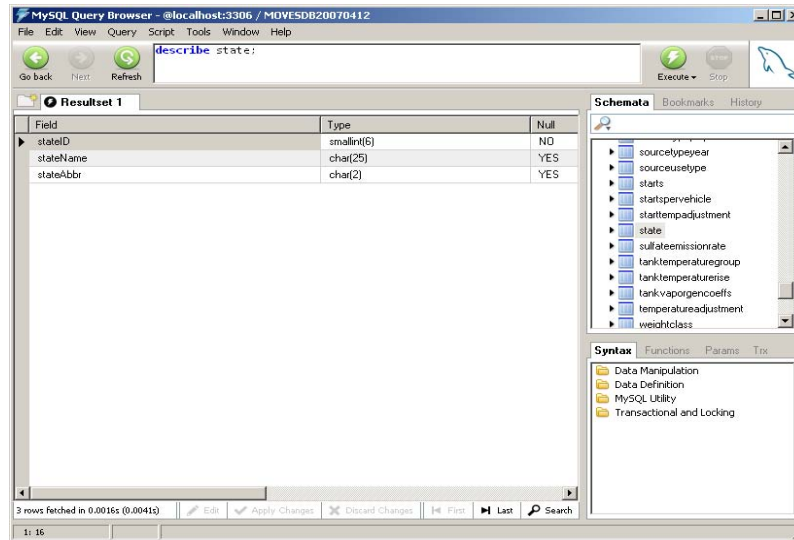
17

Double-clicking on the MOVES default database opens up tables



18

DESCRIBE state;
(or, show columns from state;)

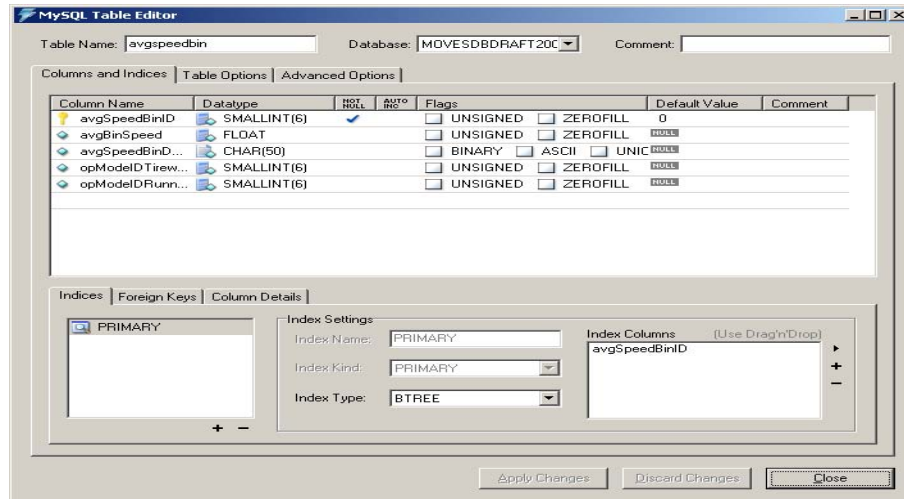


19

Describe command is used to show you the structure of a table.

You can also right click and edit a table, which will display it's structure as well.

Highlighting Table + F2
will also show the table structure.



20

The F2 function key will show the structure of the table.

You can also right click and edit a table, which will display it's structure as well.

Commonly Used Field Types in MOVES Databases

- INT (± 8388608)
- SMALLINT (± 32767)
- FLOAT(M,D), (default is (10,2))
- VARCHAR (variable length string between 1 & 255 characters)
- CHAR (fixed length string between 1 & 255 characters)

“SELECT” statements in Query Browser

- Double-clicking on table names generates a “SELECT *” (Select All) statement
- Can also drag and drop from the table listing into the results window
- You can include conditions in Select statements to filter data or locate individual data elements

SELECT * FROM state;

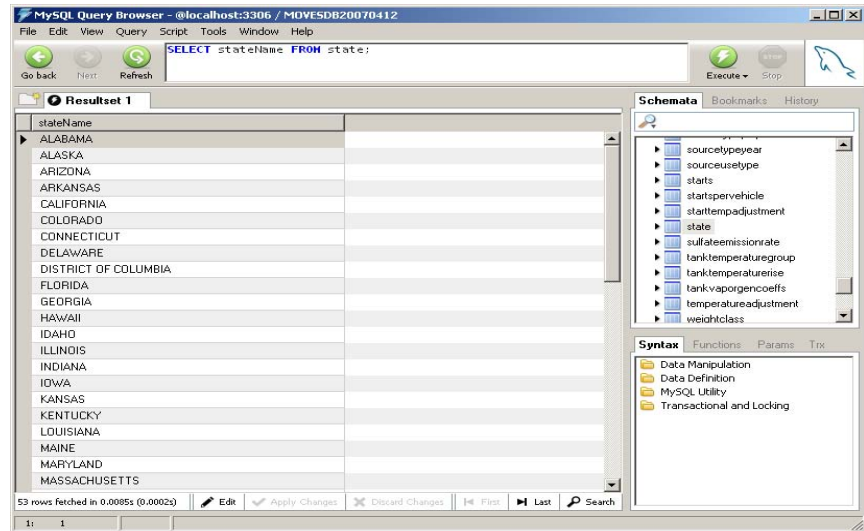
The screenshot shows the MySQL Query Browser interface. The query 'SELECT * FROM state;' is entered in the query field. The 'Execute' button is highlighted. The 'Resultset 1' tab is active, displaying a table with 25 rows of US state data. The table has three columns: stateID, stateName, and stateAbbr. The data is as follows:

stateID	stateName	stateAbbr
1	ALABAMA	AL
2	ALASKA	AK
4	ARIZONA	AZ
5	ARKANSAS	AR
6	CALIFORNIA	CA
8	COLORADO	CO
9	CONNECTICUT	CT
10	DELAWARE	DE
11	DISTRICT OF COLUMBIA	DC
12	FLORIDA	FL
13	GEORGIA	GA
15	HAWAII	HI
16	IDAHO	ID
17	ILLINOIS	IL
18	INDIANA	IN
19	IOWA	IA
20	KANSAS	KS
21	KENTUCKY	KY
22	LOUISIANA	LA
23	MAINE	ME
24	MARYLAND	MD
25	MASSACHUSETTS	MA

The status bar at the bottom indicates '53 rows fetched in 0.0084s (0.0003s)'.

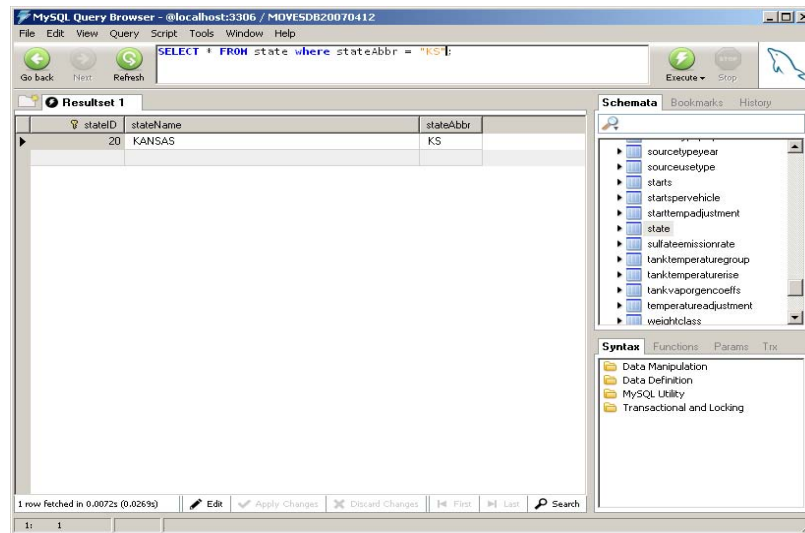
23

SELECT stateName FROM state;



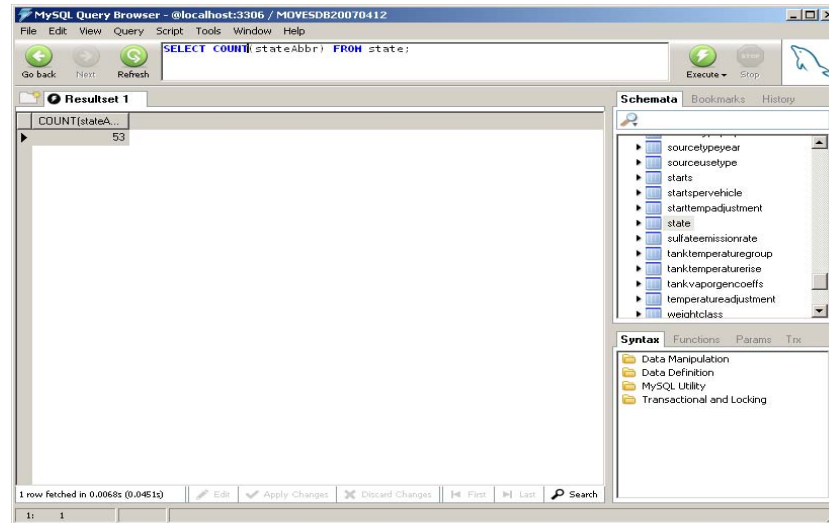
24

SELECT * FROM state where stateAbbr = "KS";



25

SELECT COUNT(stateAbbr) FROM state;



26

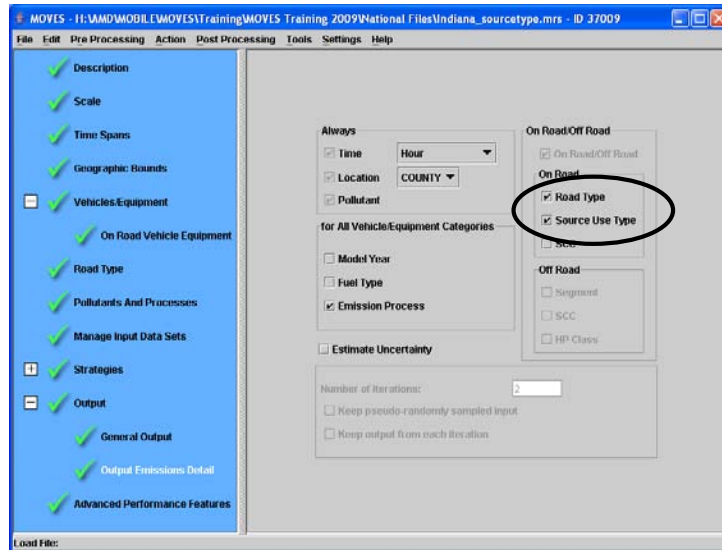
Working with MOVES output

- **MySQL commands can be used to sort or sum MOVES output by year, month, pollutant, road type, etc.**
 - Can filter results in MySQL and export to Excel
- **The Output Emissions Detail options can also be used to perform some output aggregation, but once applied, the underlying detail is not saved**
 - MySQL or Excel can be used to aggregate the output after the MOVES run, so that the detail in the native MOVES output is still available if it's needed in the future
 - In some cases, you need aggregation factors (e.g., travel time fractions)—total *emissions* of a given pollutant are additive, but *emissions rates* need to be weighted

Working with output: examples

- 1) Summing emissions output across roadtypes
- 2) Filtering MOVES lookup output for desired emissions rates

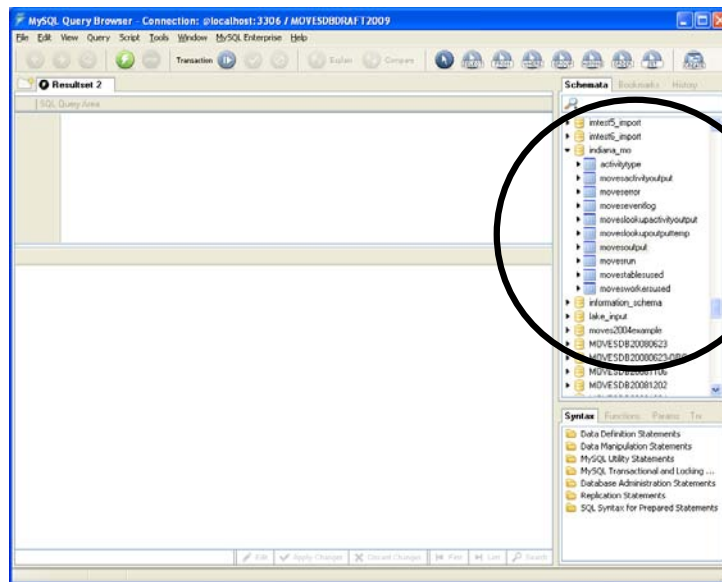
Specifying Detail in Output Emissions Detail screen



29

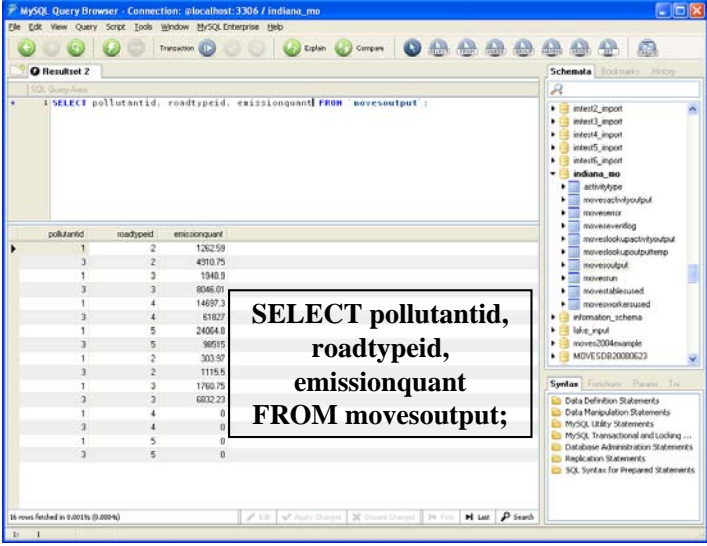
There is only 1 emission process in Indiana.mrs (running), but emission process is selected so that it is identified in the output database rather than being NULL.

MOVES output tables



30

Emissions output table, filtered for pollutant, roadtype, emissions



The screenshot shows the MySQL Query Browser interface. The query window contains the following SQL statement:

```
SELECT pollutantid, roadtypeid, emissionquant FROM movesoutput;
```

The results are displayed in a table with the following data:

pollutantid	roadtypeid	emissionquant
1	2	1262.59
3	2	4910.75
1	3	1948.9
3	3	8046.01
1	4	14097.3
3	4	61827
1	5	24064.0
3	5	98515
1	2	303.97
3	2	1115.6
1	3	1760.75
3	3	6802.23
1	4	0
3	4	0
1	5	0
3	5	0

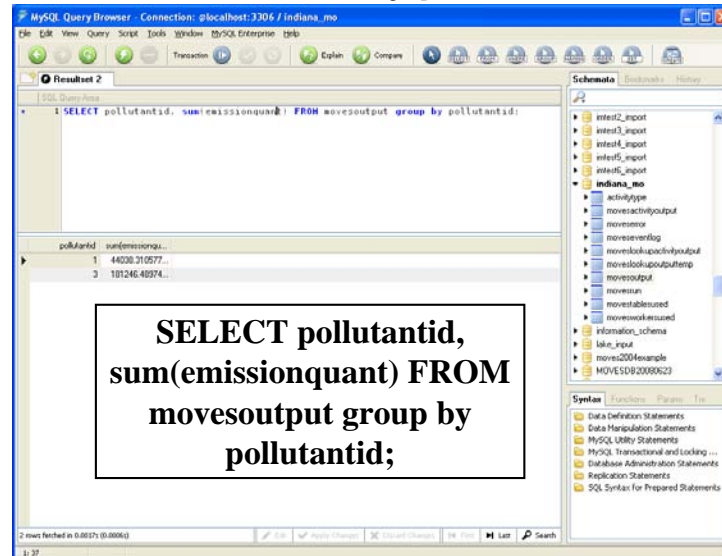
A text box is overlaid on the table with the following text:

```
SELECT pollutantid,  
roadtypeid,  
emissionquant  
FROM movesoutput;
```

The status bar at the bottom indicates: 16 rows fetched in 0.0021s (0.0004%).

31

Summing emissions across road types



The screenshot shows the MySQL Query Browser interface. The title bar indicates a connection to 'localhost:3306 / indiana_mo'. The SQL statement entered is: `SELECT pollutantid, sum(emissionquant) FROM movesoutput group by pollutantid;`. The results pane shows two rows of data:

pollutantid	sum(emissionquant)
1	44038.210577...
3	101246.40374...

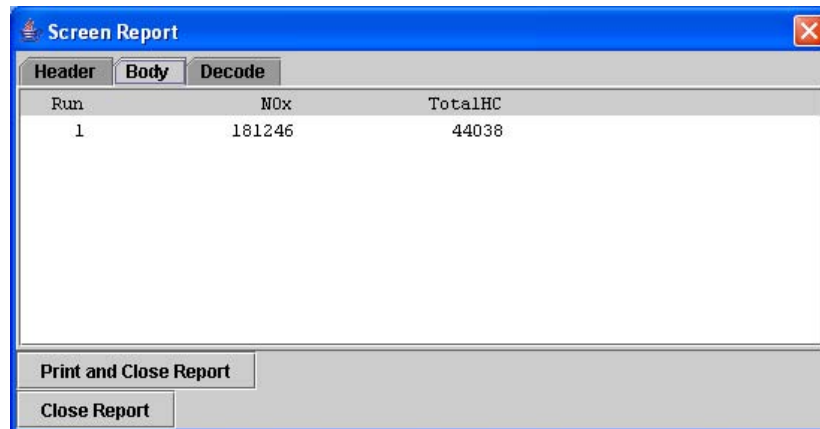
Overlaid on the screenshot is a text box containing the following SQL query:

```
SELECT pollutantid,  
sum(emissionquant) FROM  
movesoutput group by  
pollutantid;
```

The status bar at the bottom indicates '2 rows fetched in 0.00137s (0.0006s)'.

32

MOVES summary report for the same run:



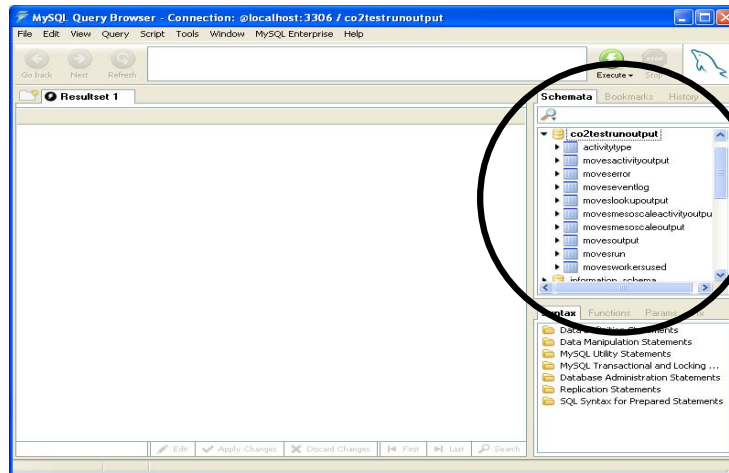
The screenshot shows a 'Screen Report' window with a blue title bar and a close button. It contains a table with three columns: 'Run', 'NOx', and 'TotalHC'. The table has one data row. Below the table are two buttons: 'Print and Close Report' and 'Close Report'.

Run	NOx	TotalHC
1	181246	44038

Other example scripts for summing output

- `SELECT MOVESRunID, count(*), sum(distance)
FROM MOVESActivityOutput GROUP BY
MOVESRunID;`
 - (count(*) returns the total number of records summed)
- `SELECT MOVESRunID, pollutantID, processID,
count(*), sum(emissionQuant) FROM MOVESOutput
GROUP BY MOVESRunID, pollutantID, processID;`

Working with MOVES Lookup Output Tables

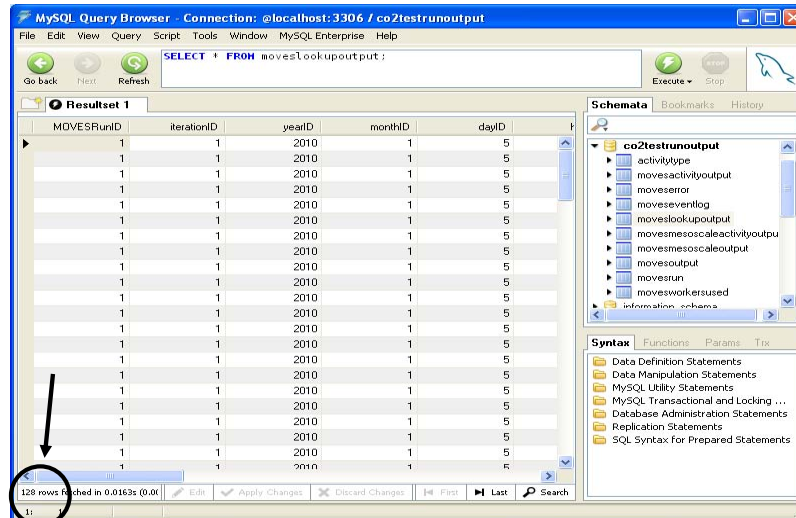


35

Participants do not have the database used in the following slides.

```
SELECT * FROM moveslookupoutput;
```

(returns 128 rows in this example)



SELECT * FROM moveslookupoutput
where roadtypeid = 2;
(returns 32 rows)

MySQL Query Browser - Connection: @localhost:3306 / co2testrunoutput

File Edit View Query Script Tools Window MySQL Enterprise Help

Go back Next Refresh **SELECT * FROM moveslookupoutput where roadtypeid = 2;** Execute Stop

Resultset 1

pollutantID	processID	avgSpeedBinID	temperature	relHumidity	emissionRate
91	MALE	16	16	58.7	0.00509537
90	MALE	16	16	58.7	382.61
91	MALE	15	16	58.7	0.00492462
90	MALE	15	16	58.7	369.788
91	MALE	14	16	58.7	0.00474055
90	MALE	14	16	58.7	355.967
91	MALE	13	16	58.7	0.00459657
90	MALE	13	16	58.7	345.158
91	MALE	12	16	58.7	0.00454631
90	MALE	12	16	58.7	341.385
91	MALE	11	16	58.7	0.0045643
90	MALE	11	16	58.7	342.737
91	MALE	10	16	58.7	0.00464463
90	MALE	10	16	58.7	348.772
91	MALE	9	16	58.7	0.00474506
90	MALE	9	16	58.7	356.313
91	MALE	8	16	58.7	0.00467423
90	MALE	8	16	58.7	366.012
91	MALE	7	16	58.7	0.00506742
90	MALE	7	16	58.7	380.519
91	MALE	6	16	58.7	0.00556448
90	MALE	6	16	58.7	417.844

32 rows fetched in 0.0149s (0.00)

Syntax Functions Params Trx

- Data Definition Statements
- Data Manipulation Statements
- MySQL Utility Statements
- MySQL Transactional and Locking ...
- Database Administration Statements
- Replication Statements
- SQL Syntax for Prepared Statements

37

SELECT * FROM moveslookupoutput
where roadtypeid=2 and pollutantid=90;
(Returns 16 rows)

MySQL Query Browser - Connection: @localhost:3306 / co2testrunoutput

File Edit View Query Script Tools Window MySQL Enterprise Help

Go back Next Refresh

FROM moveslookupoutput where roadtypeid = 2 and pollutantid = 90;

Execute Stop

Resultset 1

pollutantID	processID	avgSpeedBinID	temperature	relHumidity	emissionRate
90	16	15	16	58.7	382.61
90	15	15	16	58.7	369.788
90	14	16	16	58.7	355.967
90	13	16	16	58.7	345.158
90	12	16	16	58.7	341.395
90	11	16	16	58.7	342.737
90	10	16	16	58.7	348.772
90	9	16	16	58.7	356.313
90	8	16	16	58.7	366.012
90	7	16	16	58.7	380.519
90	6	16	16	58.7	417.844
90	5	16	16	58.7	473.836
90	4	16	16	58.7	550.268
90	3	16	16	58.7	703.348
90	2	16	16	58.7	1158.94
90	1	16	16	58.7	2128.1

16 rows fetched in 0.0142s (0.00)

1 1

Schemata Bookmarks History

co2testrunoutput

- activitytype
- movesactivityoutput
- movesensor
- moveseventlog
- moveslookupoutput
- movesmesoscaleactivityoutput
- movesmesoscaleoutput
- movesoutput
- movesrun
- movesworkersused
- information_schemas

Syntax Functions Params Trx

- Data Definition Statements
- Data Manipulation Statements
- MySQL Utility Statements
- MySQL Transactional and Locking ...
- Database Administration Statements
- Replication Statements
- SQL Syntax for Prepared Statements

38

SELECT * FROM moveslookupoutput
 where roadtypeid = 2 and pollutantID = 90
 group by avgSpeedBinID;
 (now ordered by speed bin)

MySQL Query Browser - Connection: @localhost:3306 / co2testrunoutput

File Edit View Query Script Tools Window MySQL Enterprise Help

Go back Next Refresh

ut where roadtypeid = 2 and pollutantid = 90 group by avgSpeedBinID;

Execute Stop

Resultset 1

pollutantID	processID	avgSpeedBinID	temperature	relHumidity	emissionRate
90	MALE	1	16	58.7	2128.1
90	MALE	2	16	58.7	1158.94
90	MALE	3	16	58.7	703.348
90	MALE	4	16	58.7	550.268
90	MALE	5	16	58.7	473.836
90	MALE	6	16	58.7	417.844
90	MALE	7	16	58.7	380.519
90	MALE	8	16	58.7	366.012
90	MALE	9	16	58.7	356.313
90	MALE	10	16	58.7	348.772
90	MALE	11	16	58.7	342.737
90	MALE	12	16	58.7	341.385
90	MALE	13	16	58.7	345.158
90	MALE	14	16	58.7	355.967
90	MALE	15	16	58.7	369.788
90	MALE	16	16	58.7	382.61

16 rows fetched in 0.0141s (0.00)

1: 1

Schemata Bookmarks History

co2testrunoutput

- activitytype
- movesactivityoutput
- moveserror
- moveseventlog
- moveslookupoutput
- movesmesoscaleactivityoutput
- movesmesoscaleoutput
- movesoutput
- movesrun
- movesworkersused
- information_schema

Syntax Functions Params Trx

- Data Definition Statements
- Data Manipulation Statements
- MySQL Utility Statements
- MySQL Transactional and Locking ...
- Database Administration Statements
- Replication Statements
- SQL Syntax for Prepared Statements

39

SELECT pollutantID, avgspeedbinID,
emissionrate FROM moveslookupoutput
where roadtypeID = 2 and pollutantID = 90
group by avgspeedbinID;

MySQL Query Browser - Connection: @localhost:3306 / co2testrunoutput

File Edit View Query Script Tools Window MySQL Enterprise Help

Go back Next Refresh

SELECT pollutantID, avgspeedbinID, emissionrate FROM moveslookupoutput

Execute Stop

Resultset 1

pollutantID	avgspeedbinID	emissionrate
90	1	21.281
90	2	1158.94
90	3	703.349
90	4	550.268
90	5	473.836
90	6	417.844
90	7	380.519
90	8	366.012
90	9	356.313
90	10	348.772
90	11	342.737
90	12	341.395
90	13	345.158
90	14	355.967
90	15	369.788
90	16	382.61

16 rows fetched in 0.0032s (0.00%)

1: 1

Schemata

co2testrunoutput

- activitytype
- movesactivityoutput
- moveserror
- moveseventlog
- moveslookupoutput
- movesmesoscaleactivityoutput
- movesmesoscaleoutput
- movesoutput
- movesrun
- movesworkersused

Information - schemas

Syntax

- Data Definition Statements
- Data Manipulation Statements
- MySQL Utility Statements
- MySQL Transactional and Locking ...
- Database Administration Statements
- Replication Statements
- SQL Syntax for Prepared Statements

40

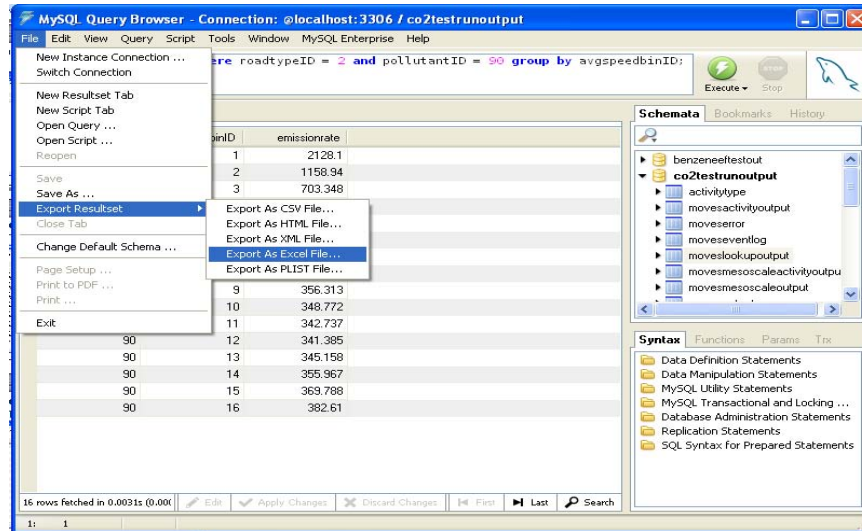
Other handy things Query Browser can do

- Export result tables.
- Import data.
- Edit data.
- History function.

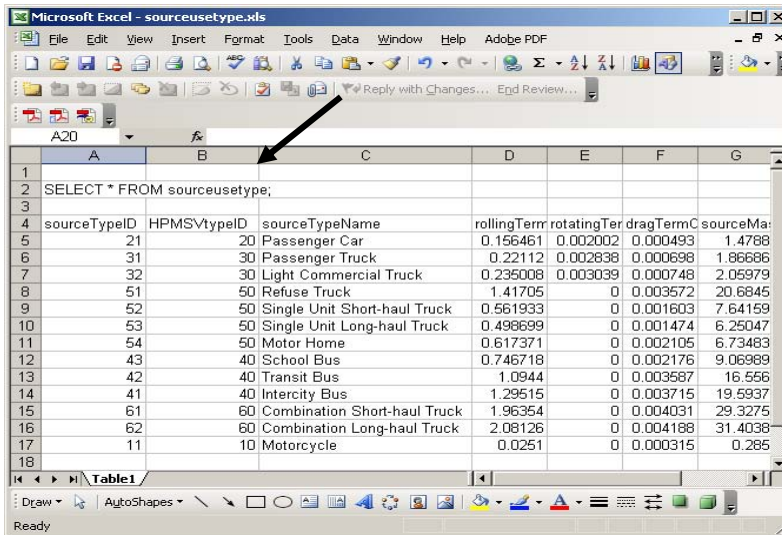
Export tables using the Browser

- One useful way to build input databases is to first copy the relevant information from the MOVES default database
- Use **SELECT** and **WHERE** statements as necessary to filter the information and get the values that you want
- In Query Browser file menu, select “Export Resultset”—can save as .csv, .xls, or other file formats
- Then you can work with these values in Excel or other programs

Export resultsets from Browser



Tip: Copy query statements into exported spreadsheets for future reference



The screenshot shows a Microsoft Excel window titled "Microsoft Excel - sourcecasetype.xls". The spreadsheet contains a table with 7 columns: sourceTypeID, HPMSVTypeID, sourceTypeName, rollingTerm, rotatingTerm, dragTermC, and sourceMa. The data is organized into rows, with the first row (row 4) containing the column headers. The table is named "Table1" in the bottom left corner. An arrow points to the query statement in cell A20: "SELECT * FROM sourcecasetype;".

sourceTypeID	HPMSVTypeID	sourceTypeName	rollingTerm	rotatingTerm	dragTermC	sourceMa
21	20	Passenger Car	0.156461	0.002002	0.000493	1.4788
31	30	Passenger Truck	0.22112	0.002838	0.000698	1.86686
32	30	Light Commercial Truck	0.235008	0.003039	0.000748	2.05979
51	50	Refuse Truck	1.41705	0	0.003572	20.6845
52	50	Single Unit Short-haul Truck	0.561933	0	0.001603	7.64159
53	50	Single Unit Long-haul Truck	0.498699	0	0.001474	6.25047
54	50	Motor Home	0.617371	0	0.002105	6.73483
43	40	School Bus	0.746718	0	0.002176	9.06989
42	40	Transit Bus	1.0944	0	0.003587	16.556
41	40	Intercity Bus	1.29515	0	0.003715	19.5937
61	60	Combination Short-haul Truck	1.96354	0	0.004031	29.3275
62	60	Combination Long-haul Truck	2.08126	0	0.004188	31.4038
11	10	Motorcycle	0.0251	0	0.000315	0.285

44

Load data infile

- This statement can be used to import information from .csv or .txt (not .xls) files *into* MySQL tables
- Example script to update age values:
 - load data infile 'c:/MOVES/Change/NewAgeDist.csv'
 - Replace into table SourceTypeAgeDistribution
 - fields terminated by ','
 - ignore 1 lines
 - (sourceTypeId,yearId,ageId,ageFraction);

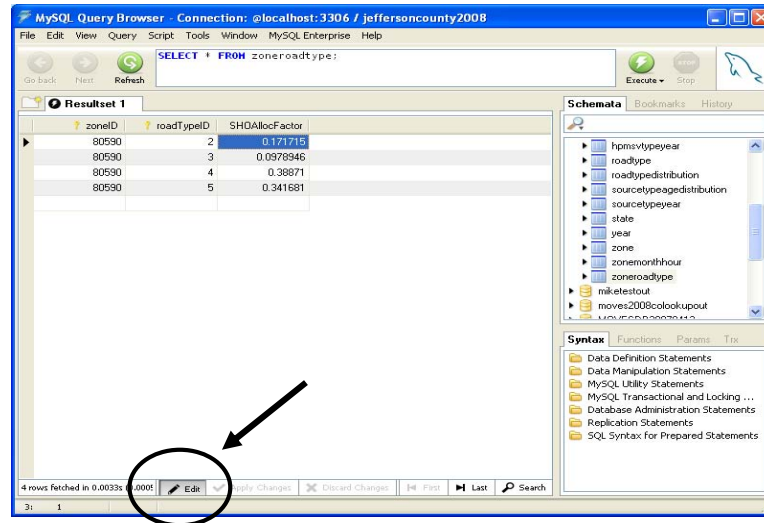
Editing data from within Query Browser

- Cell values in tables with primary keys can also be edited from within Query Browser itself, rather than importing alternative data:
 - Highlight a cell
 - Click “Edit” button
 - Make any necessary changes
 - Click “Apply Changes”
 - Click “Edit” button again to close

Editing data from within Query Browser

- MOVES output tables cannot be readily edited this way, since they do not have primary keys.
- **Don't edit the default database!!!**
 - Check the title bar at the top of the Query Browser window to make sure that the database you *want* to edit is active

Editing data from within Query Browser



48

Copy default information into new database

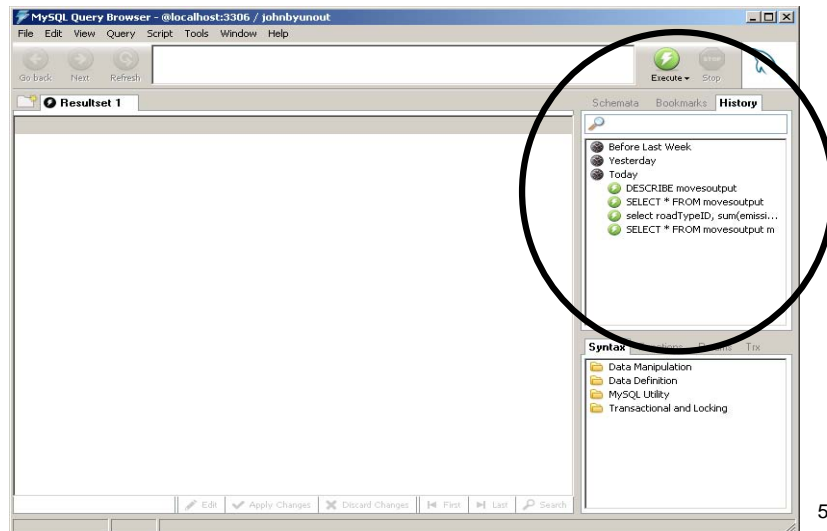
- You can also copy information from the default database directly into another database without first exporting and then importing
- This is useful when you want to edit data within Query Browser, and don't have many values to change
- Example: `insert into myrun.fuelformulation SELECT * FROM MOVESDBDRAFT2009.fuelformulation where fuelformulationid = 1061;`

In this example, "myrun" is the user input database and "MOVESDBDRAFT2009" is the default database (syntax is "databasename.tablename")

Using Query Browser's "history" function

- Query Browser has a history function, so that previous statements can be selected and applied, rather than retyping them
- Can also pull up a previous query statement and then modify it to select the desired data

Query Browser history function



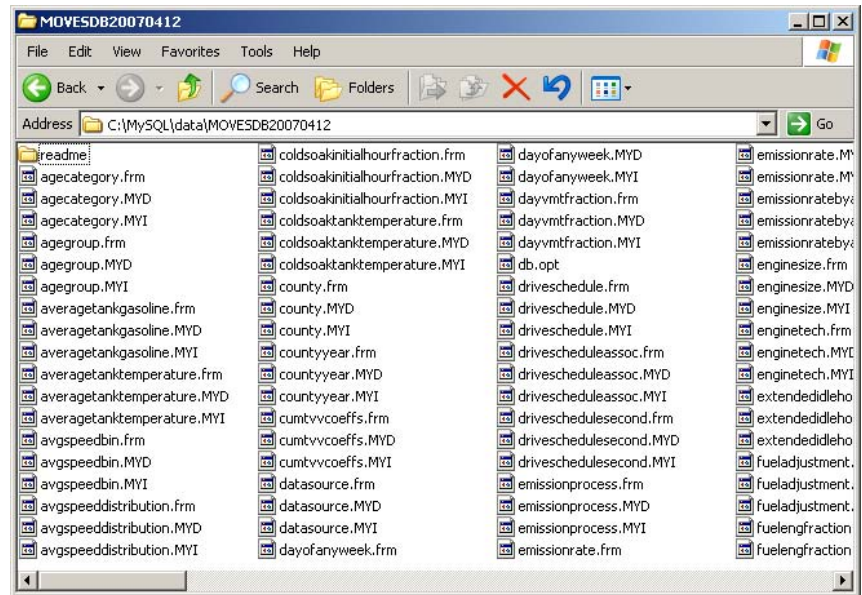
51

Generating local input databases for MOVES when GUI options don't exist

- MOVES has several GUI options for importing common types of local inputs (County Data Manager, I/M Coverage Editor, etc.)
 - These are covered in other parts of the course
- In some cases, non-default local data would need to be supplied through additional input databases (as opposed to being entered through the GUI)
- Since the tables in these databases must adhere to the MOVES default names and formats, it is safer (and easier) to base input databases on existing tables, rather than creating them from scratch.

One way: Copying default tables

- Create a new database in MySQL
- In Explorer, locate the files associated with each necessary table in the MOVES default database, and COPY (don't MOVE) them into the new database folder
- Each table has three associated files (*.frm, *.MYD, and *.MYI), all of which must be copied
- These can then be edited using MySQL commands



54

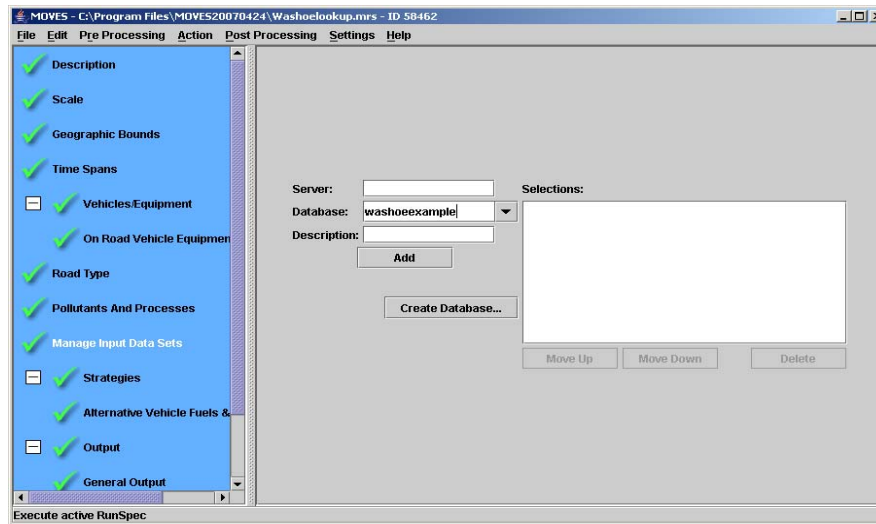
Note that all tables have three parts (frm, myd and myi). All three parts must be copied.

Use the “flush tables;” command to unlock tables that were in use.

Creating a Template Database in MOVES

- A blank database can be created in MOVES from the Manage Input Data Sets screen—this database contains all the tables that the default database does, with the same format, but no data
- This database can then be populated with necessary data

Creating input databases



56

Never, Ever, EVER Modify the MOVES Default Database!!!

- **To be safe, ALWAYS create a new database in MOVES, or copy tables from the MOVES default database into a new database, and THEN work on the tables**
 - MOVES will not alert you if you accidentally modify the default database
 - Any changes you make will be applied to all future MOVES runs using this database
 - If you modify the default database, it will also make it extremely difficult to troubleshoot puzzling model results
- **Make a backup copy of the MOVES default database, just in case . . .**