ADVANCED

SQL DATABASE ADMINISTRATION

★★★★★ *With Expert SQL Instructor* ***John Pauler***

COURSE STRUCTURE

This is a **project-based course**, for students looking for a *practical*, *hands-on*, and *highly engaging* approach to building & maintaining databases with MySQL

Additional resources include:

- **Downloadable Ebook** to serve as a helpful reference when you're offline or on the go
- **Quizzes** & **Homework Exercises** to test and reinforce key concepts, with step-by-step solutions
- **Bonus Projects** to test your abilities and apply the skills developed throughout the course

COURSE OUTLINE

1	**Introduction & Setup**	*Discuss the basic prerequisites, download Community Server and Workbench, and review the Workbench Interface*
2	**File Import, Alter, Update & Keys**	*Insert large datasets to the database using the wizard. Review alter and update, and map primary and foreign keys in our new schema*
3	**Replication, Backup & Recovery**	*Learn about replication, and how and when we use it. Discuss methods for backup and recovery of data to ensure nothing is lost*

[MID-COURSE PROJECT]

4	**Automating Database Activity**	*Use automation like triggers, stored procedures and scheduled events, and learn how website data is written to a DB in real-time*
5	**Creating Views & EER Diagrams**	*Learn how Views & EER Diagrams are used as tools for reporting and understanding data in your database*
6	**Database Security**	*Discuss best practices in database security, and learn what can happen when security isn't taken seriously*

[FINAL PROJECT]

INTRODUCING THE **COURSE PROJECT**

THE SITUATION

You've just been hired as a **Database Administrator** for **Maven Bear Builders**, an online retailer which has just launched their first product.

THE BRIEF

As a member of the startup team, you will work with the *CEO*, the *Head of Marketing*, and the *Website Manager* to help build their data infrastructure so they can grow their business.

You will create schemas and tables, load large datasets to the database, use automation to keep things running smoothly, and tackle serious issues like security, backup, and data recovery.

THE OBJECTIVE

Use SQL to:

- *Build and maintain the Maven Bear Builders database*
- *Become the data expert for the company, and the go-to person for mission critical data enhancements*
- *Leverage automation, and make decisions about security and backup to plan for long-term success*

SETTING EXPECTATIONS

1 You'll learn **MySQL Database Administration** using **MySQL Workbench**

- *In your career, you may use other "flavors" of SQL (T-SQL, PL/SQL, PostgreSQL, etc.)*
- *Each flavor is very similar. The principles you learn here will apply universally, but syntax will vary slightly*

2 We will focus on the **creation and maintenance of databases**

- *We will design and create schemas and tables from scratch, and alter existing tables to improve them*

3 This course is meant for people who **already have basic SQL + DBA skills**

- *This course skips over the SQL basics, and jumps straight into a simulated real-world experience where students can practice building and maintaining a real-world database*
- *Prior completion of **MySQL Database Administration for Beginners** and **MySQL for Data Analysis** is recommended*

4 We will **NOT** go deep on **data analysis** or into **engineering**

- *This course will focus on Database Administration concepts, like creating and maintaining databases*
- *We offer separate courses that focus on using SQL for data analysis and business intelligence (beginner + advanced-level)*
- *We will not be covering languages like PHP, Python, or Ruby*

PREREQUISITE SKILLS REVIEW

PREREQUISITE DBA SKILLS

- You should be familiar with **the basic statements** used in Data Definition and Data Manipulation: **CREATE**, **ALTER**, **INSERT**, **UPDATE**, & **DELETE**

- You should have some understanding of automation using database tools like **TRIGGERS**, and **STORED PROCEDURES**

- You should be able to create **EER DIAGRAMS**, and should be familiar with Database Design concepts like **NORMALIZATION**, & **CARDINALITY**

COMMON MySQL DATA TYPES

Data Type	Specifications
TINYINT	*Integer (-128 to 127)*
SMALLINT	*Integer (-32768 to 32767)*
MEDIUMINT	*Integer (-8388608 to 8388607)*
INT	*Integer (-2147483648 to 2147483647)*
BIGINT	*Integer (-9223372036854775808 to 9223372036854775807)*
FLOAT	*Decimal (precise to 23 digits)*
DOUBLE	*Decimal (23 to 53 digits)*
DECIMAL	*Decimal (to 65 digits – most precise)*

Data Type	Specifications
CHAR	*String (0 – 255)*
VARCHAR	*String (0 – 255)*
TINYTEXT	*String (0 – 255)*
TEXT	*String (0 – 65535)*
DATE	*YYYY-MM-DD*
DATETIME	*YYYY-MM-DD HH:MM:SS*
TIMESTAMP	*YYYYMMDDHHMMSS*
ENUM	*One of a number of preset options*

THE "**BIG 6**" **ELEMENTS** OF A SQL SELECT STATEMENT

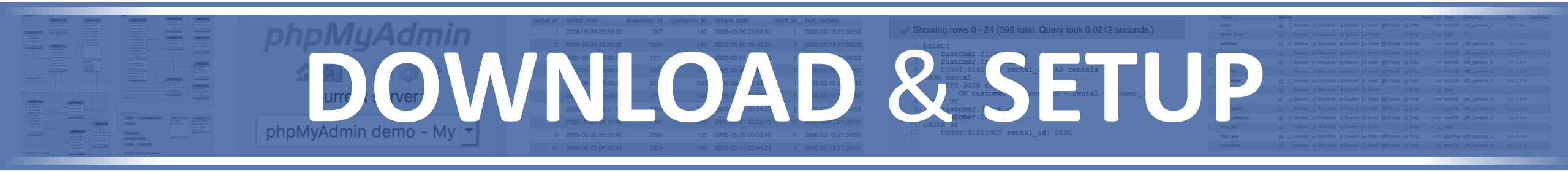
START OF STATEMENT

END OF STATEMENT

SELECT	*Identifies the column(s) you want your query to select for your results*	**SELECT** columnName
FROM	*Identifies the table(s) your query will pull data from*	**FROM** tableName
WHERE	*(Optional) Specifies record-filtering criteria for filtering your results*	**WHERE** logicalCondition
GROUP BY	*(Optional) Specifies how to group the data in your results*	**GROUP BY** columnName
HAVING	*(Optional) Specifies group-filtering criteria for filtering your results*	**HAVING** logicalCondition
ORDER BY	*(Optional) Specifies the order in which your query results are displayed*	**ORDER BY** columnName

COMMON **JOIN** TYPES

INNER JOIN	*Returns records that exist in **BOTH** tables, and excludes unmatched records from either table*	**FROM** leftTableName **INNER JOIN** rightTableName
LEFT JOIN	*Returns ALL records from the **LEFT** table, and any matching records from the **RIGHT** table*	**FROM** leftTableName **LEFT JOIN** rightTableName
RIGHT JOIN	*Returns ALL records from the **RIGHT** table, and any matching records from the **LEFT** table*	**FROM** leftTableName **RIGHT JOIN** rightTableName
FULL OUTER JOIN	*Returns ALL records from **BOTH** tables, including non-matching records*	**FROM** leftTableName **FULL JOIN** rightTableName
UNION	*Returns all data from one table, with all data from another table **appended to the end***	**SELECT FROM** firstTableName **UNION** **SELECT FROM** secondTableName



DOWNLOAD & SETUP

MySQL DOWNLOAD & SETUP – **OVERVIEW**

Step 1	**Download Community Server**	*This allows SQL to run on your machine*
Step 2	**Download MySQL Workbench**	*This is the program you'll use to write and run SQL queries (it's intuitive, and works across operating systems)*
Step 3	**Connect Workbench to Server**	*We'll get you connected to the server so you can use Workbench to start running your own SQL queries*
Step 4	**Review Workbench Interface**	*We'll take a quick tour of the Workbench interface to get you familiar with the layout and key components*
Step 5	**Create the Database**	*We'll run the SQL code to build the database which we'll be exploring throughout the course (this part is easy!)*

HEY THIS IS IMPORTANT!

If you took one of my other courses, and have already installed Community Server and MySQL Workbench, then you can skip ahead to creating the database. No need to re-install. Whatever version you have is great.

STEP 1: COMMUNITY SERVER (MAC)

HEY THIS IS IMPORTANT!

If you took one of my other courses, and have already installed Community Server and MySQL Workbench, then you can skip ahead to creating the database. No need to re-install. Whatever version you have is great.

MySQL COMMUNITY SERVER – **MAC DOWNLOAD GUIDE**

1. Go to https://dev.mysql.com/downloads and download **MySQL Community Server**

2. Select the **MacOS** operating system, and download the **DMG Archive** version
   - ***Note**: you'll likely see a later version than the one shown (just download the latest)*

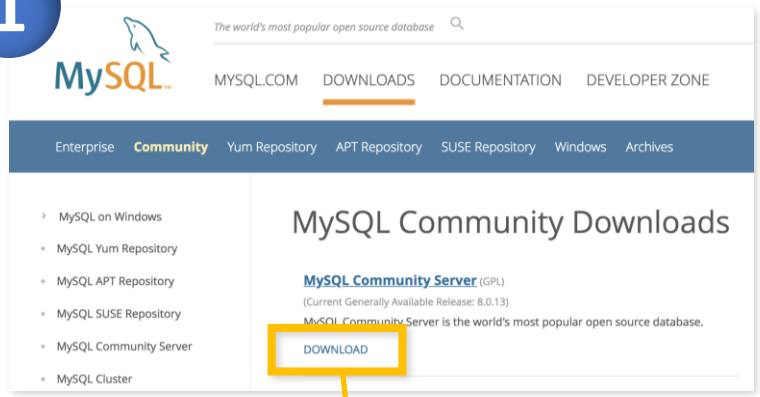
3. No need to Login or Sign Up, just click "***No thanks, just start my download***"

4. Find the install file in your downloads, then double click to run the installer package

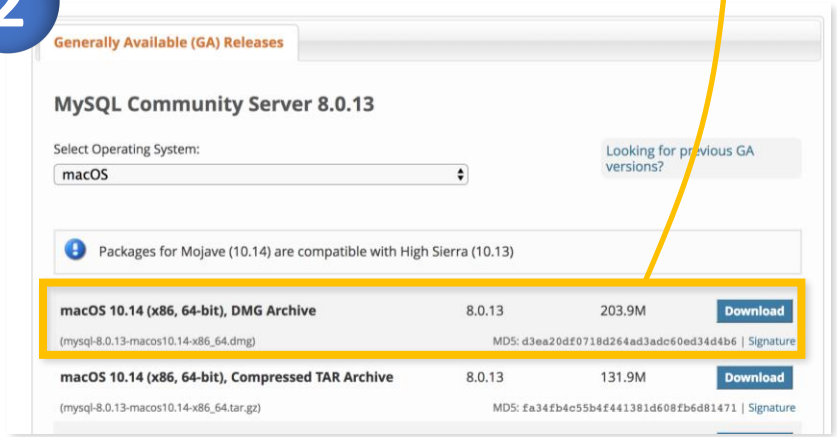
5. Click through each install step, leaving defaults unless you need customized settings
   - ***Note**: Make sure to store your **root password** somewhere, you'll need this later!*

MySQL COMMUNITY SERVER – **MAC DOWNLOAD GUIDE**

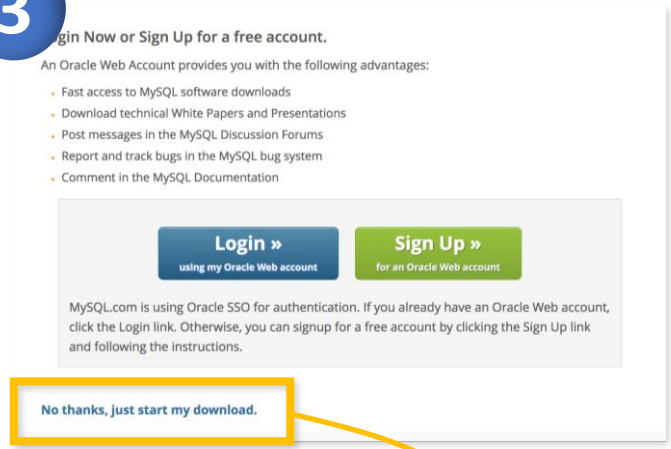
1



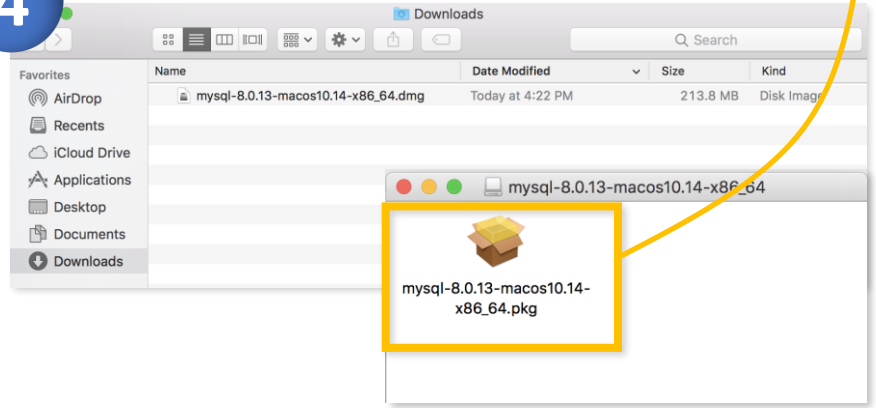
2



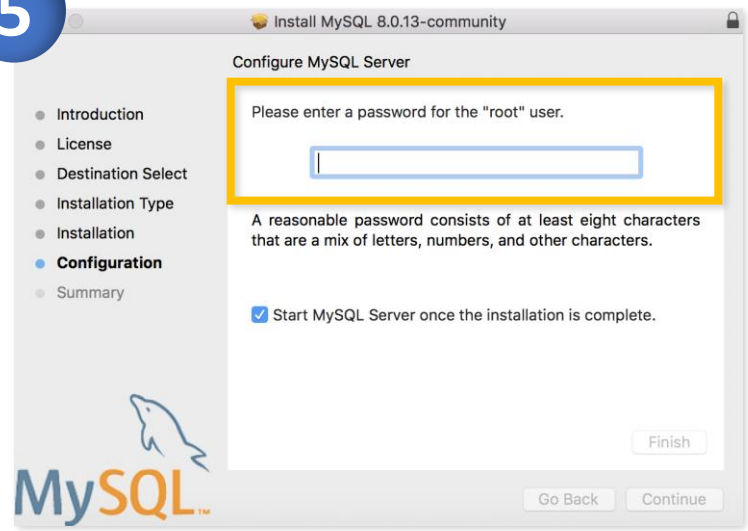
3



4



5



STEP 1: COMMUNITY SERVER (PC)

HEY THIS IS IMPORTANT!

If you took one of my other courses, and have already installed Community Server and MySQL Workbench, then you can skip ahead to creating the database. No need to re-install. Whatever version you have is great.

MySQL COMMUNITY SERVER – **PC DOWNLOAD GUIDE**

1. Go to https://dev.mysql.com/downloads and download **MySQL Community Server**

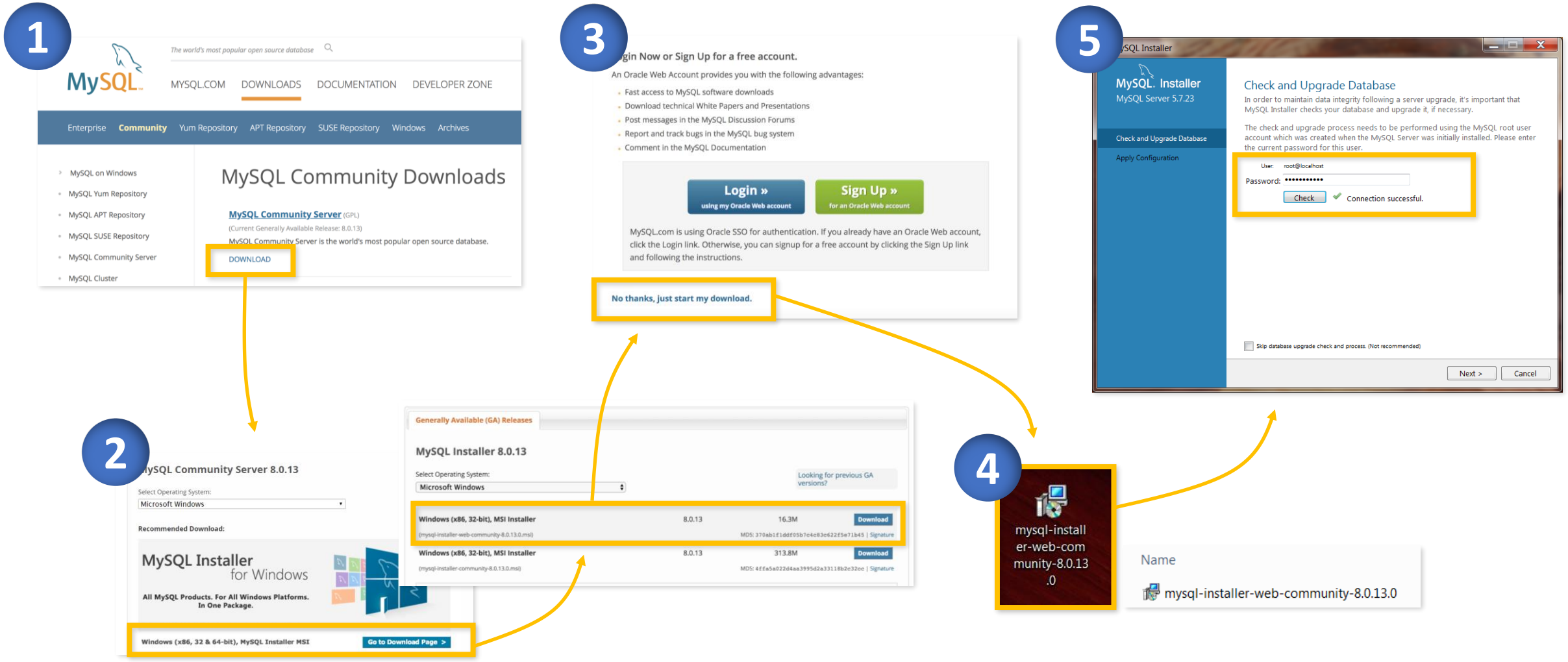
2. Select the **Microsoft Windows** operating system, and the **Installer MSI** download
   - ***Note**: On the download page you may see two versions: select **mysql-installer-web-community** if you are connected to the internet, and keep in mind that you may see a later version than the one shown (just download the latest)*

3. No need to Login or Sign Up, just click "***No thanks, just start my download***"

4. Find the install file in your downloads, then double click to run the installer package

5. Click through each install step, leaving defaults unless you need customized settings
   - ***Note**: Make sure to store your **root password** somewhere, you'll need this later!*

MySQL COMMUNITY SERVER – **PC DOWNLOAD GUIDE**



STEP 2: MySQL WORKBENCH (MAC)

HEY THIS IS IMPORTANT!

If you took one of my other courses, and have already installed Community Server and MySQL Workbench, then you can skip ahead to creating the database. No need to re-install. Whatever version you have is great.

MySQL WORKBENCH – **MAC DOWNLOAD GUIDE**

1. Go to https://dev.mysql.com/downloads/workbench, scroll down to **Generally Available (GA) Releases**, and select the **MacOS** operating system

2. We'll be using version **8.0.16** for this course, so you can either click ***"Looking for previous GA versions?"*** to search for the same one, or simply download the latest available

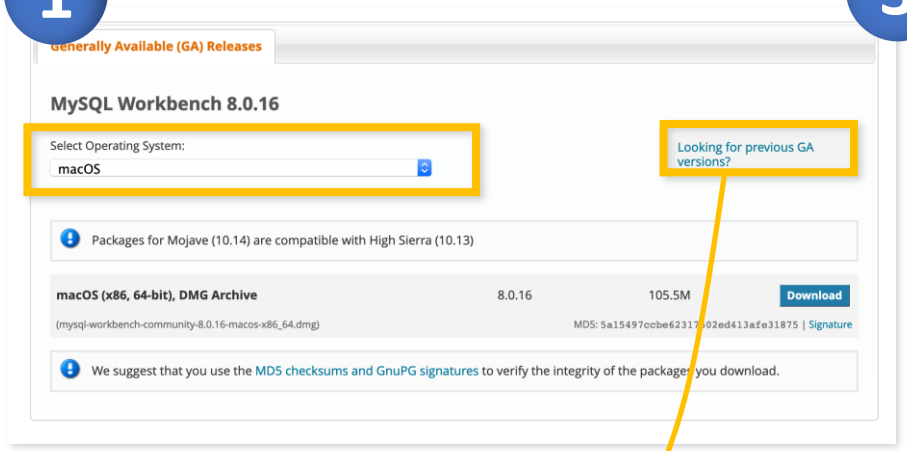
3. No need to Login or Sign Up, just click "***No thanks, just start my download***"

4. Find the install file in your downloads, click the MySQL Workbench logo (*with the dolphin*) and drag it into your **Applications** folder

5. Look for MySQL workbench in your list of applications, double click to launch, then proceed to ***Step 3: Connecting to the server***

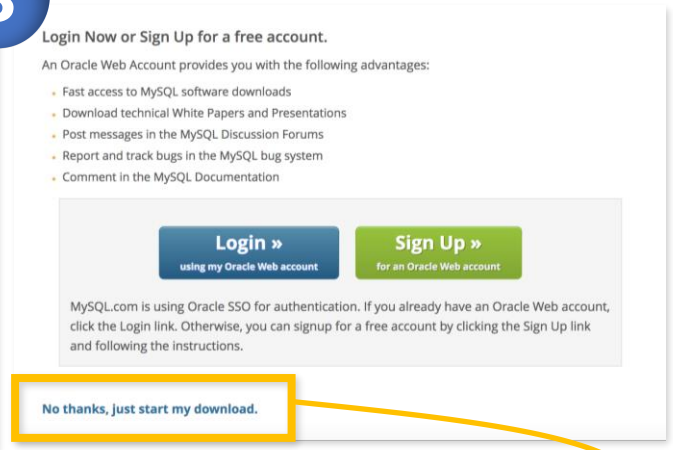
MySQL WORKBENCH – **MAC DOWNLOAD GUIDE**

1



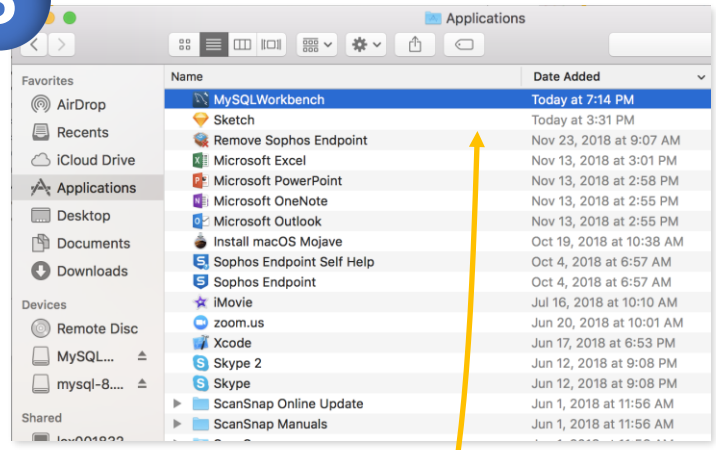
2 *Look for version **8.0.16**, or download the latest*

3

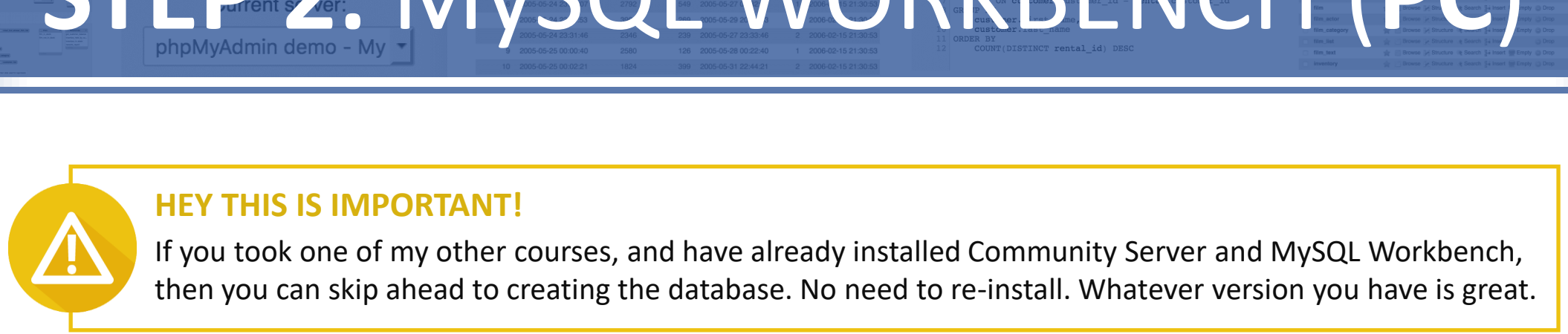


4

5



STEP 2: MySQL WORKBENCH (PC)



HEY THIS IS IMPORTANT!

If you took one of my other courses, and have already installed Community Server and MySQL Workbench, then you can skip ahead to creating the database. No need to re-install. Whatever version you have is great.

MySQL WORKBENCH – **PC DOWNLOAD GUIDE**

1. Go to https://dev.mysql.com/downloads/workbench, scroll down to **Generally Available (GA) Releases**, and select the **Microsoft Windows** operating system

2. We'll be using version **8.0.13** for this course, so you can either click ***"Looking for previous GA versions?"*** to search for the same one, or simply download the latest available

3. No need to Login or Sign Up, just click "***No thanks, just start my download***"

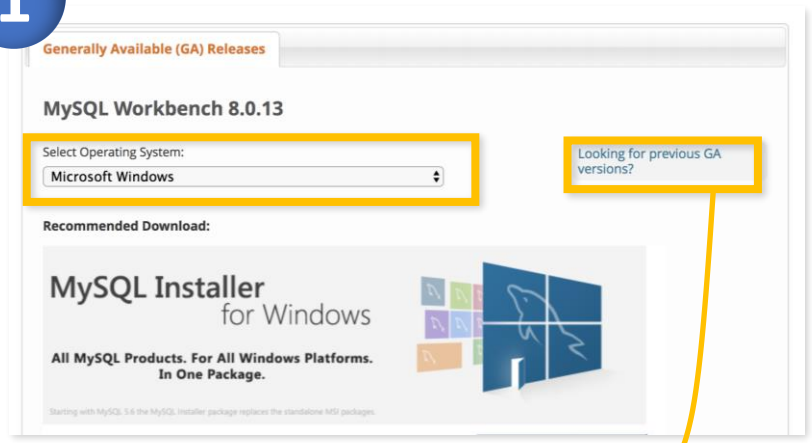
4. Find the install file in your downloads, double click to run the installation process, and stick with default settings unless you need a custom configuration

5. Look for MySQL workbench in your list of programs, double click to launch, then proceed to ***Step 3: Connecting to the server***

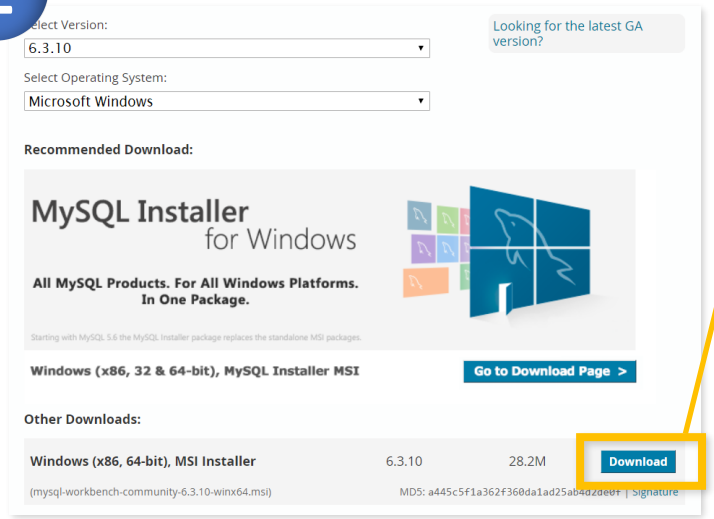
   - ***Note:*** *You may see a warning if you aren't on* ***Windows 10+****, but most older systems (i.e. Windows 7) should be compatible*

MySQL WORKBENCH – **PC DOWNLOAD GUIDE**

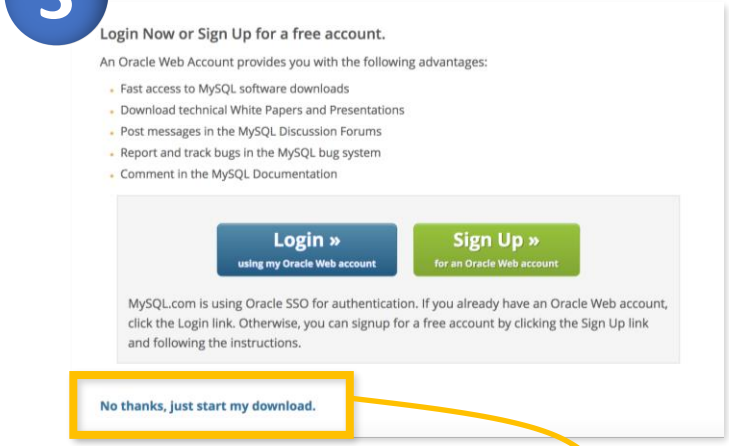
1



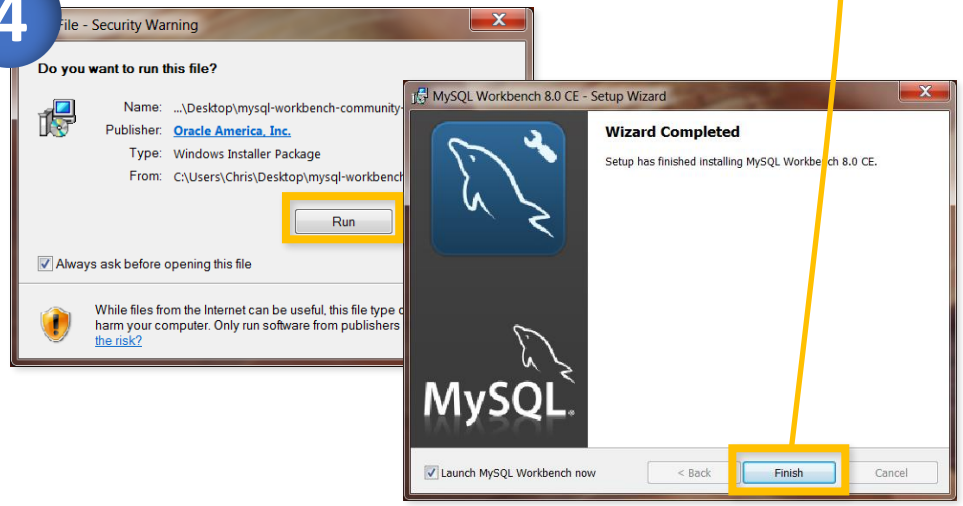
2 *Look for version **8.0.13**, or download the latest*



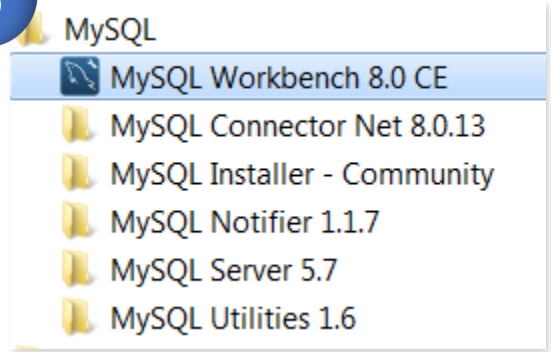
3



4



5



STEP 3: CONNECTING TO THE SERVER

CONNECTING TO THE SERVER

1. After launching Workbench, check the **MySQL Connections** section on the welcome page
   - *If you see a connection already, right-click to **Edit Connection**, otherwise click the **plus sign (+)** to add a new one*

2. Name the connection "**mavenmovies**", confirm that the Username is "**root**", and click **OK**

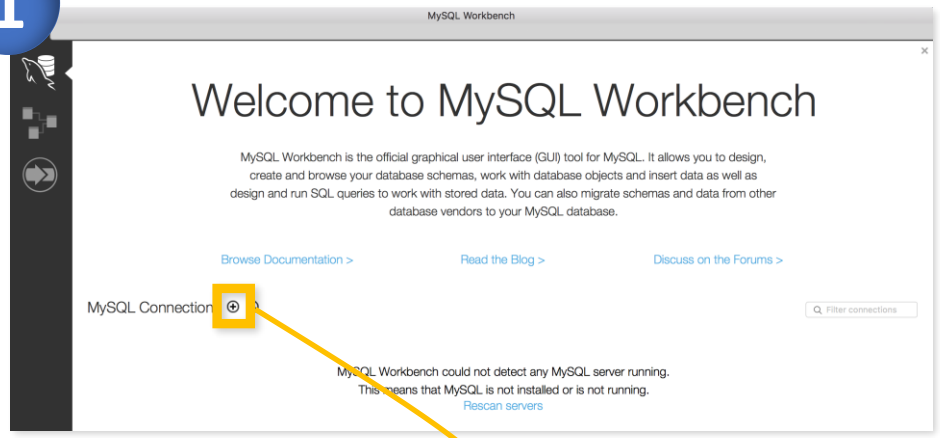
3. Once you see the **mavenmovies** connection on your welcome screen, simply click the tile and enter your **root password** to complete the connection

Fun Fact!

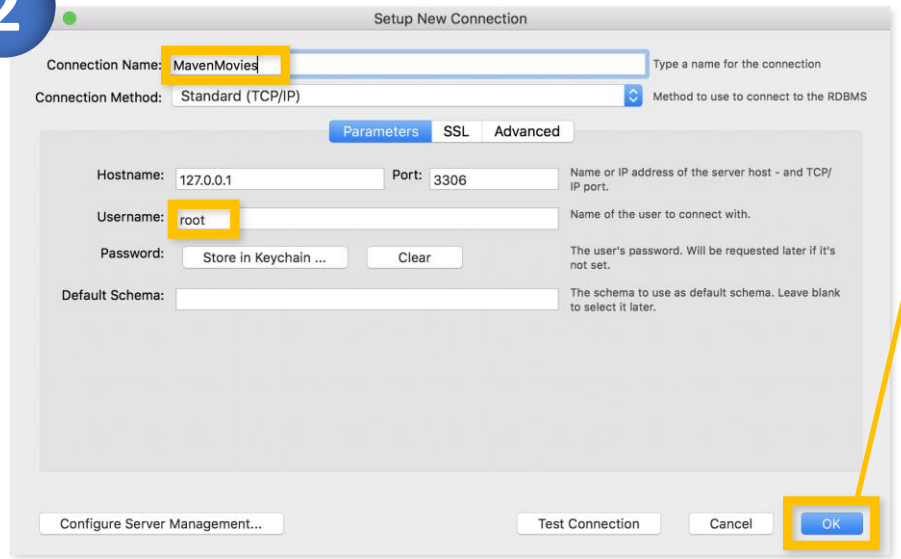
Maven Movies is the name of the database I used when I made my first course. I always name my connections 'mavenmovies' as tribute. **It does not matter what you name your connection. Name it anything you want!**

CONNECTING TO THE SERVER

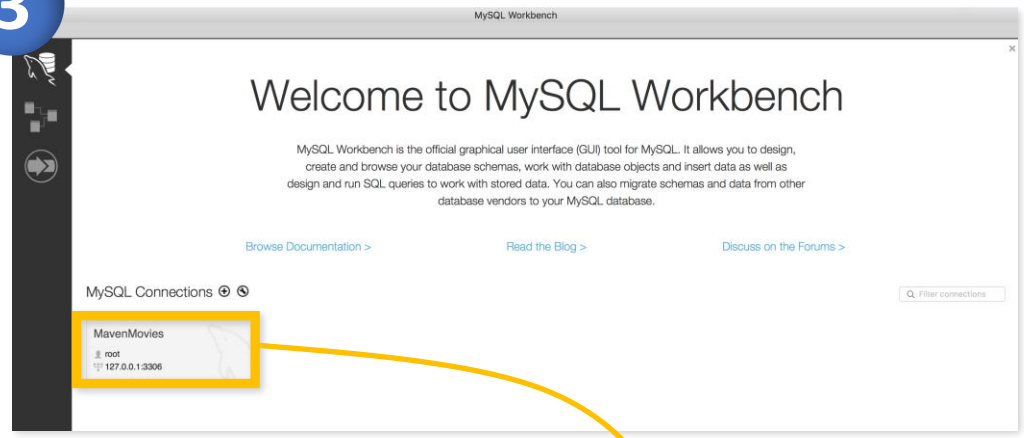
1



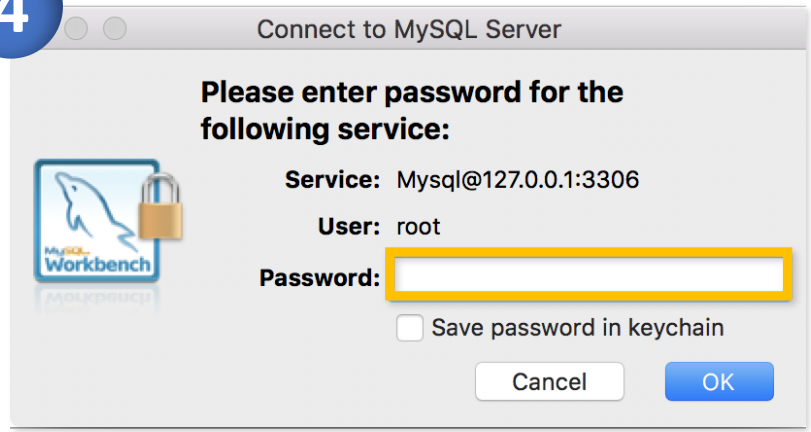
2



3



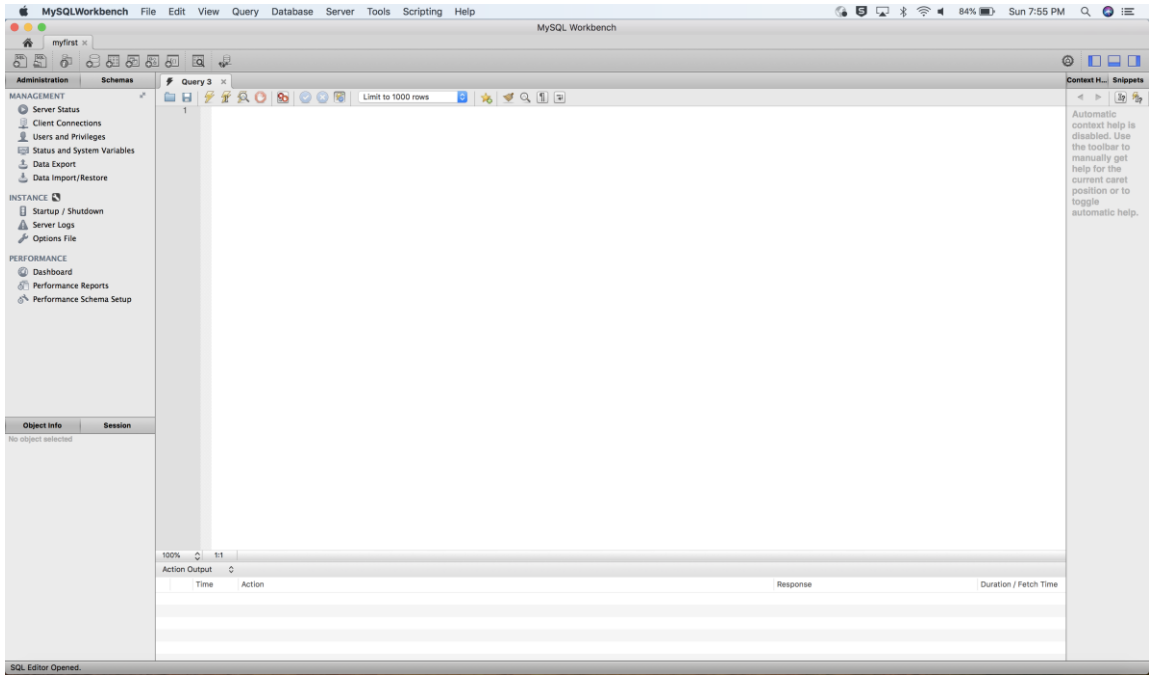
4



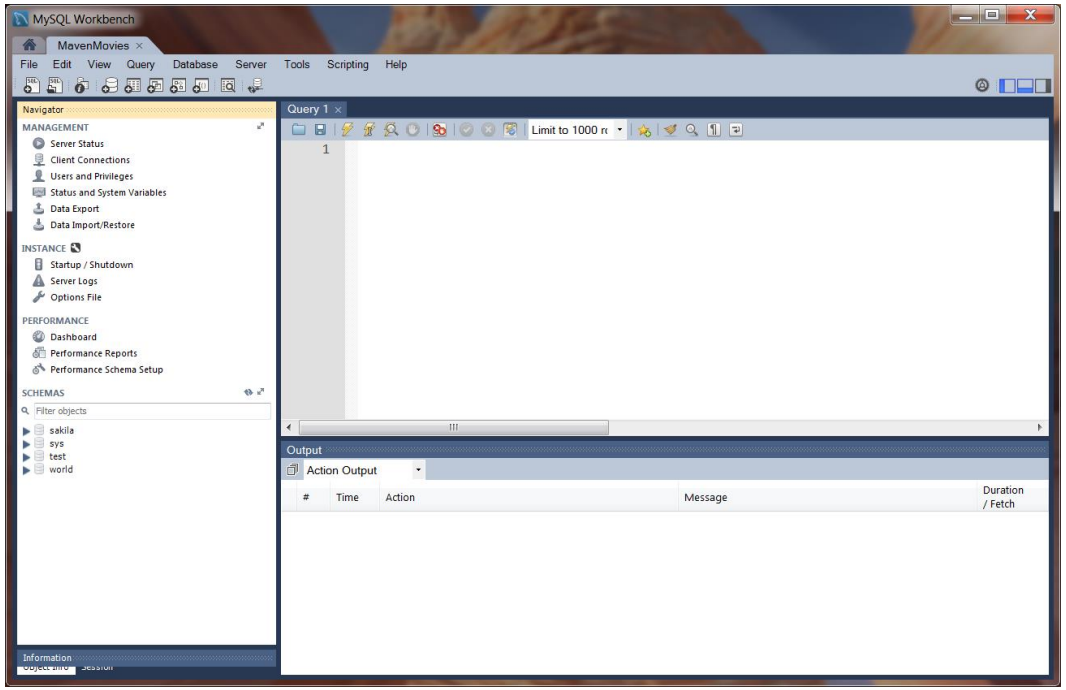
STEP 4: MySQL WORKBENCH INTERFACE

MySQL WORKBENCH INTERFACE (**MAC** VS. **PC**)

Mac interface



PC interface



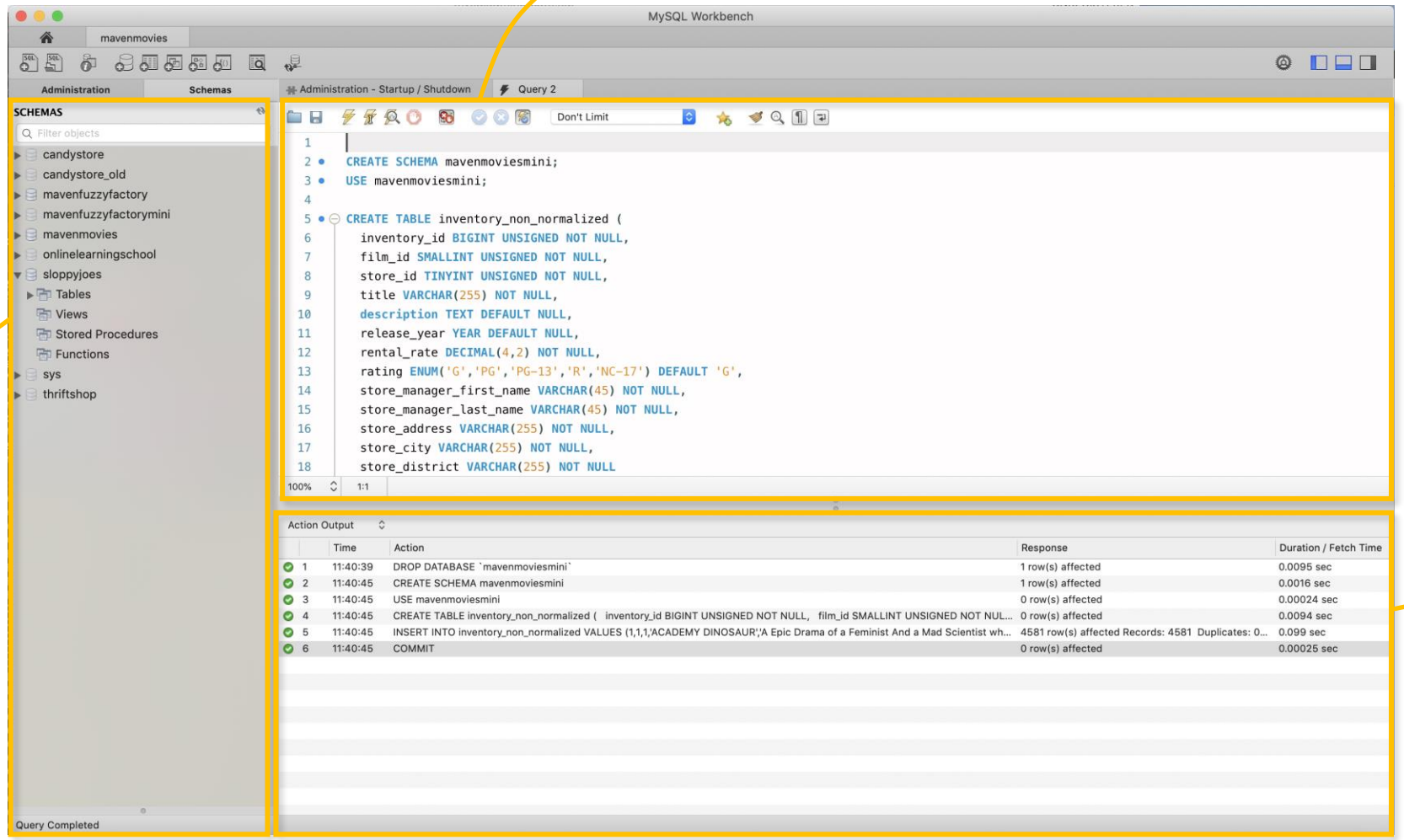
HEY THIS IS IMPORTANT!

Workbench looks slightly different on **Mac** vs. **PC**, but everything you need is found in the same place. While the course is recorded on a Mac, but you should have no problem keeping up on a PC

QUICK TOUR: THE WORKBENCH INTERFACE

Query Editor Window

This is where you write and run your code



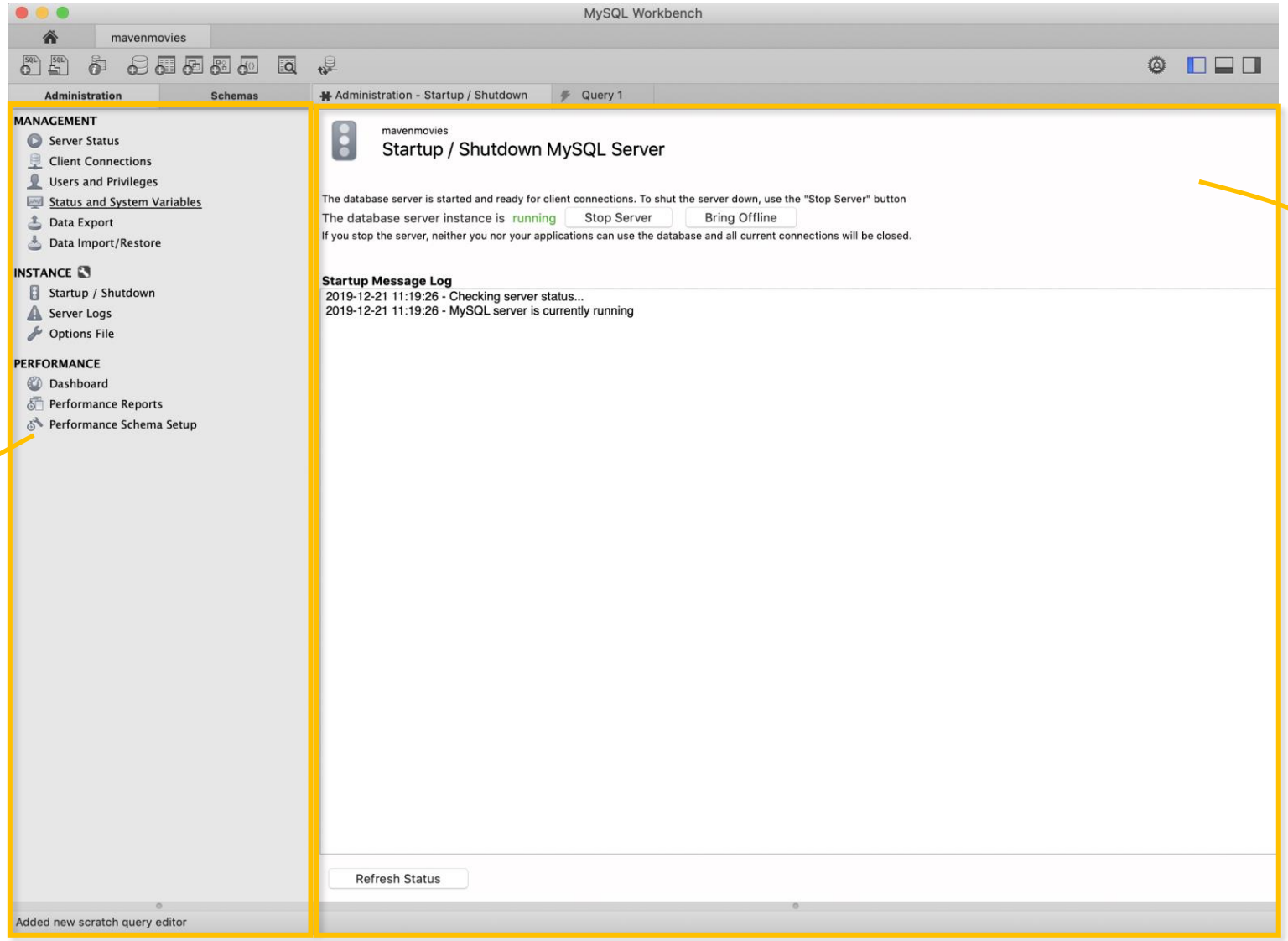
Schemas Tab

Here you can view tables and views in your schemas

Action Output

*This is a summary of actions taken by the server (**TIP**: the **Response** column is great for troubleshooting!)*

QUICK TOUR: THE WORKBENCH INTERFACE



Administration Tab

Here you can select the Management, Instance, and Performance tools

Tool Tabs

This is where the various Management, Instance, and Performance tools show up after they are selected in the Administration Tab.

Note: these show up as additional tabs alongside tabs you have open for SQL query editing. They co-exist in the same section of Workbench, even thought their functions are different

IMPORTANT: PREPARE WORKBENCH SETTINGS

THIS IS IMPORTANT!

Before we get started, we will quickly configure a few settings in Workbench which will dramatically cut down on problems you might otherwise face later in the course. Please do not skip this step.

1. First, we'll adjust a date setting, so we won't run into issues with our dates
2. Next, we'll update the GROUP BY setting to only allow FULL GROUP BY
3. Then, we'll adjust our MAX ALLOWED PACKET size, so any larger files we use will work
4. Last, we will shut down Workbench and restart, which will launch our new settings

INSERTING DATA INTO THE DATABASE

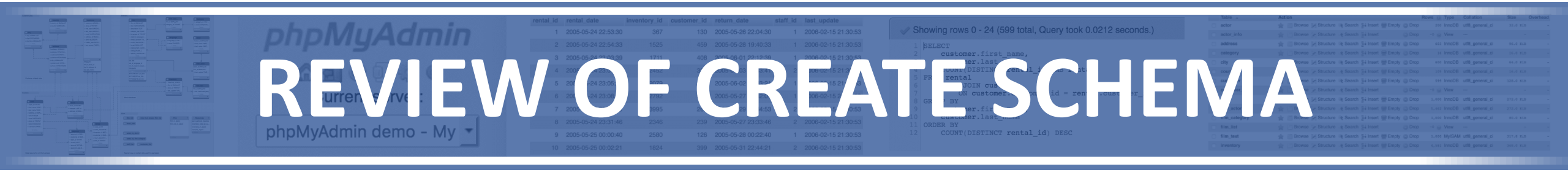
REVIEW OF CREATE, INSERT, & DELETE

1. First, we'll review CREATE SCHEMA to get started
2. Then, we'll review using CREATE TABLE to add a table to the schema
3. Next, we'll review the INSERT statement, used to add records to tables
4. Finally, we'll review the DELETE statement, used to remove records from tables

THIS IS REVIEW!

We covered the basics of CREATE, INSERT, & DELETE in our MySQL Database Administration for Beginners course. If you already have these concepts covered, feel free to skip ahead to the lecture on Importing Data From a File.

REVIEW OF CREATE SCHEMA



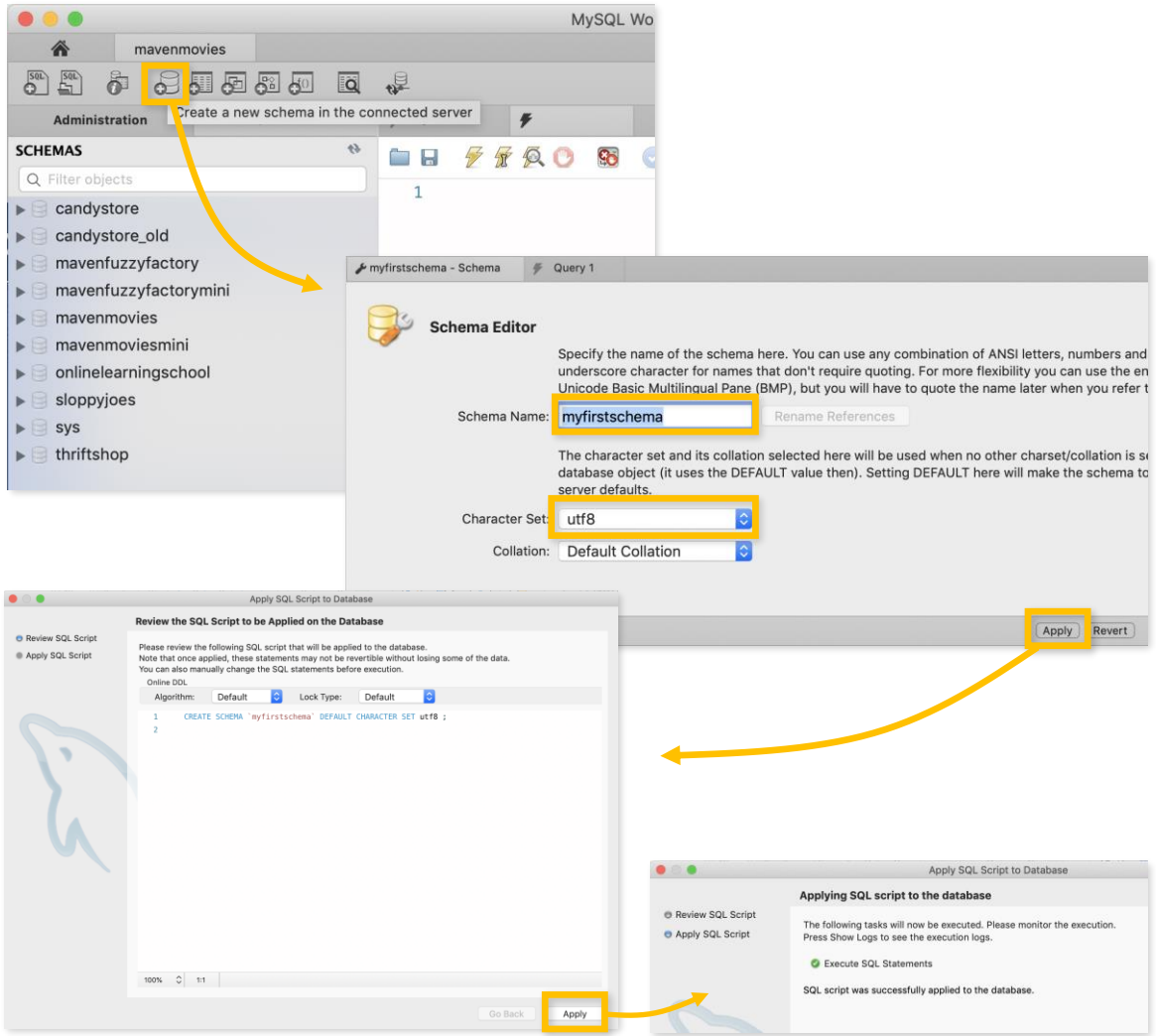
THIS IS REVIEW!

We covered the basics of creating schemas in our MySQL Database Administration for Beginners course.
If you already have this concept covered, feel free to skip this lecture.

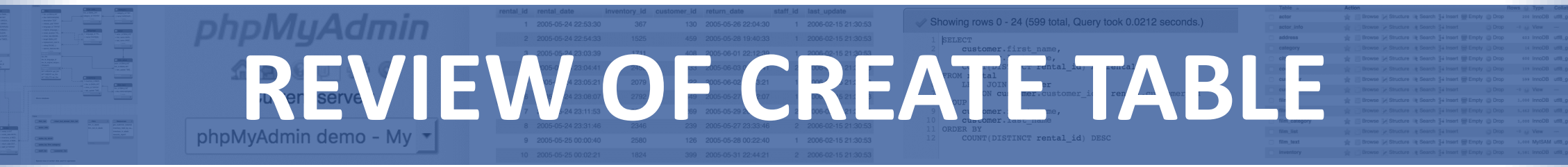
CREATE SCHEMA W/ UI

- In MySQL, the terms '**Schema**' and '**Database**' will be used interchangeably to discuss a collection of data tables which relate to one another
- We can create a new schema using Workbench's user interface tools, with just a few quick clicks
- Whether we create schemas using the UI tools or using code is a matter of preference

MySQL WORKBENCH TOOLS IN ACTION:



REVIEW OF CREATE TABLE



THIS IS REVIEW!

We covered the basics of creating tables in our MySQL Database Administration for Beginners course. If you already have this concept covered, feel free to skip this lecture.

CREATE TABLE W/ CODE

- After our schema is created, we can begin populating our schema with tables that will contain records of data
- Creating a new table with code typically involves:
  - Selecting a schema to add the table to
  - Giving the new table a name
  - Specifying which columns to include
  - Setting data formats and types

MySQL QUERY IN ACTION:

```
1
2   USE myfirstcodeschema; -- specifying the schema to use
3
4   CREATE TABLE myfirstcodetable (
5     myfirstcodetable_id BIGINT NOT NULL,
6     my_character_field VARCHAR(50),
7     my_text_field TEXT,
8     my_created_at TIMESTAMP
9   );
10
```

ACTION OUTPUT:

Action Output

	Time	Action	Response
1	14:59:51	USE myfirstcodeschema	0 row(s) affected
2	14:59:54	CREATE TABLE myfirstcodetable (myfirstcodetable_i...	0 row(s) affected

REVIEW OF INSERTING RECORDS

THIS IS REVIEW!

We covered the basics of inserting records into tables in our MySQL Database Administration for Beginners course. If you already have this concept covered, feel free to skip this lecture.

INSERTING NEW RECORDS

- MySQL makes it easy to insert additional records into a table using an **INSERT INTO** statement
- We can insert one row at a time, or we can batch insert many rows in one single SQL statement
- We may choose to specify values for every column in our **INSERT INTO** statement, or we may omit values for some columns. If we omit values, MySQL places the default value for that column

MySQL QUERY IN ACTION:

```
USE thriftshop; -- sets the schema

SELECT * FROM inventory; -- just to see what's already there

INSERT INTO inventory (inventory_id, item_name, number_in_stock) VALUES
(10,'wolf skin hat',1);
```

(Another) MySQL QUERY IN ACTION:

```
-- this time we will batch insert
-- we will also not name the columns (this is optional)
INSERT INTO inventory VALUES
(11,'fur fox skin',1),
(12,'plaid button up shirt',8),
(13,'flannel zebra jammies',6);
```

REVIEW OF DELETING RECORDS

THIS IS REVIEW!

We covered the basics of deleting records into tables in our MySQL Database Administration for Beginners course. If you already have this concept covered, feel free to skip this lecture.

DELETE RECORDS

- MySQL allows us to target one or more records to **DELETE FROM** a given data table
- We will do this using the **DELETE FROM** statement, followed by the tablename
- As with **UPDATE** statements, we will usually want to specify which records to delete using a **WHERE** clause
- If we do not specify a **WHERE** clause, **DELETE FROM** will delete ALL records

MySQL QUERY IN ACTION:

```
-- deleting ski blankets, no longer stocked
DELETE FROM inventory
WHERE inventory_id = 7
```

ACTION OUTPUT:

inventory_id	item_name	number_in_stock
1	fur coat	0
2	mocassins	4
3	velour jumpsuit	12
4	house slippers	6
5	brown leather jacket	3
6	broken keyboard	6
7	ski blanket	1
8	kneeboard	2
9	pro wings sneakers	0
10	wolf skin hat	1
11	fur fox skin	1
12	plaid button up shirt	8
13	flannel zebra jammies	6

inventory_id	item_name	number_in_stock
1	fur coat	0
2	mocassins	4
3	velour jumpsuit	12
4	house slippers	6
5	brown leather jacket	3
6	broken keyboard	6
8	kneeboard	2
9	pro wings sneakers	0
10	wolf skin hat	1
11	fur fox skin	1
12	plaid button up shirt	8
13	flannel zebra jammies	6

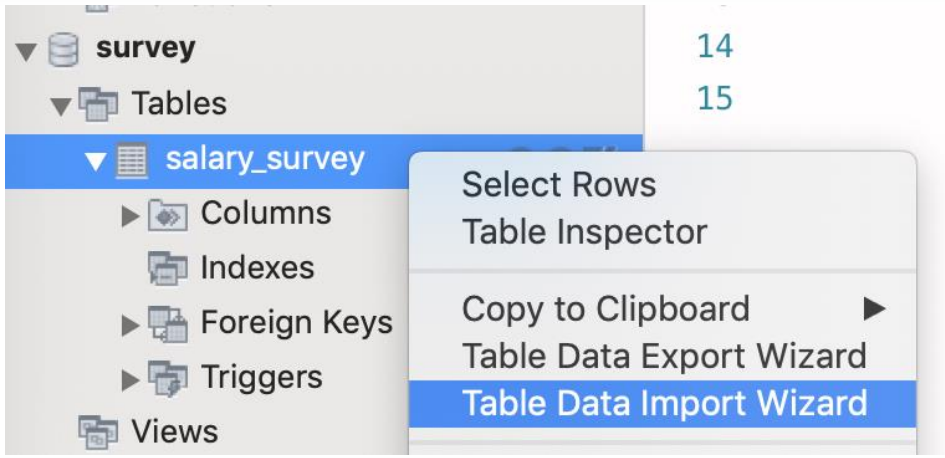
IMPORT DATA FROM A FILE

- Sometimes we will want to import a large amount of data at once
- Writing an INSERT statement and specifying every row of a large data set would be very tedious work
- Luckily, MySQL Workbench provides a couple of ways to import data in bulk
- The method we will primarily be using is the Table Data Import Wizard

MySQL QUERY IN ACTION:

```
1  CREATE SCHEMA survey;
2
3  USE survey;
4
5  CREATE TABLE salary_survey (
6      country VARCHAR(50),
7      years_experience INT,
8      employment_status VARCHAR(50),
9      job_title VARCHAR(50),
10     is_manager VARCHAR(50),
11     education_level VARCHAR(50)
12     );
```

Accessing the Table Data Import Wizard:



NEW MESSAGE
April 2, 2012

From: **Sally Bleu** *(CEO)*

Subject: **Help get order data into the database**

Good morning,

I'm excited to get going on the data centralization project! As a first step, let's get the items customers are ordering stored in the database.

I've attached 2 files from our sales software. Can you create a table and import this data into our database?

Thanks!

-Sally

01.order_items_2012_Mar
02.order_items_2012_Apr

Reply Forward

Helpful Hints

```
-- first, you'll want to create a new schema
CREATE SCHEMA mavenbearbuilders;

-- make sure to set it as the default schema too
USE mavenbearbuilders;

-- then, create a new table, order_items with columns...
    -- order_item_id
    -- created_at
    -- order_id
    -- price_usd
    -- cogs_usd
    -- website_session_id

-- if you get stuck on the syntax, check out the review videos
```

TEST YOUR SKILLS: **IMPORT ORDER ITEM DATA**

NEW MESSAGE
April 2, 2012

From: **Sally Bleu** *(CEO)*

Subject: **Help get order data into the database**

Good morning,

I'm excited to get going on the data centralization project! As a first step, let's get the items customers are ordering stored in the database.

I've attached 2 files from our sales software. Can you create a table and import this data into our database?

Thanks!

-Sally

01.order_items_2012_Mar
02.order_items_2012_Apr

Reply Forward

Solution

```
29    -- if you've done this right, you'll have 159 records
30 •  SELECT COUNT(*) AS total_records FROM order_items;
```

100% 1:18

Result Grid Filter Rows: Search Export:

totaLrecords
159

TEST YOUR SKILLS: **IMPORT ORDER ITEM DATA**

NEW MESSAGE
April 5, 2012

From: **Sally Bleu** *(CEO)*

Subject: **Can we aggregate refunds too?**

Hey there,

Now that we have **order_items** built out, could you also import this attached April refund data (weren't any in March) in a new table called **order_item_refunds**?

It will be great to start tracking refund rates better so we can keep an eye on product quality.

Thanks!

-Sally

03.order_item_refunds_2012_Apr

Reply Forward

Helpful Hints

```
-- make sure to set it as the default schema
USE mavenbearbuilders;

-- then, create a new table, order_item_refunds with columns...
    -- order_item_refund_id
    -- created_at
    -- order_item_id
    -- order_id
    -- refund_amount_usd
```

TEST YOUR SKILLS: **IMPORT REFUND DATA**

NEW MESSAGE
April 5, 2012

From: **Sally Bleu** *(CEO)*

Subject: **Can we aggregate refunds too?**

Hey there,

Now that we have **order_items** built out, could you also import this attached April refund data (weren't any in March) in a new table called **order_item_refunds**?

It will be great to start tracking refund rates better so we can keep an eye on product quality.

Thanks!

-Sally

03.order_item_refunds_2012_Apr

Reply Forward

Solution

```
22    -- if you've done this right, you'll have 10 records
23 •  SELECT COUNT(*) AS total_records FROM order_item_refunds;
```

100% 3:19

Result Grid Filter Rows: Search Export:

total_records
10

TEST YOUR SKILLS: **IMPORT REFUND DATA**

NEW MESSAGE
April 8, 2012

From: **Sally Bleu** *(CEO)*

Subject: **Help Cleaning Up Some Bad Data**

Well…

Turns out the new guy messed up some of our data. He flagged order_items **131**, **132**, **145**, **151**, and **153** as refunds, but they were actually customer inquiries.

Can you remove these from **order_item_refunds** to clean up our data?

Thanks!

-Sally

Reply Forward

Helpful Hints

```
-- I like to do a simple SELECT * to see what's in there
SELECT * FROM order_item_refunds;

-- TIP: be careful deleting without using the PK
```

TEST YOUR SKILLS: **DELETING RECORDS**

NEW MESSAGE

April 8, 2012

From: **Sally Bleu** *(CEO)*

Subject: **Help Cleaning Up Some Bad Data**

Well…

Turns out the new guy messed up some of our data. He flagged order_items **131**, **132**, **145**, **151**, and **153** as refunds, but they were actually customer inquiries.

Can you remove these from **order_item_refunds** to clean up our data?

Thanks!

-Sally

Reply Forward

Solution

```
31    -- if you've done this right, you'll have 5 records remaining
32 •  SELECT COUNT(*) AS total_records FROM order_item_refunds;
```

100% 23:27

Result Grid Filter Rows: Search Export:

total_records
5

TEST YOUR SKILLS: **DELETING RECORDS**

NEW MESSAGE
January 02, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Time to pull in all 2012 data**

Hey,

The business finished 2012 strong, and I would like to get all of our **order_items** and **order_item_refunds** data updated through the end of the year.

Can you help me import the 2 attached files into the correct tables?

Thanks!

-Sally

04.order_items_2012_May-Dec
05.order_item_refunds_May-Dec

Reply Forward

Helpful Hint

```
-- first, import the order_items data

-- then, import the order_item_refunds data
```

TEST YOUR SKILLS: **IMPORT REMAINING 2012 DATA**

NEW MESSAGE
January 02, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Time to pull in all 2012 data**

Hey,

The business finished 2012 strong, and I would like to get all of our **order_items** and **order_item_refunds** data updated through the end of the year.

Can you help me import the 2 attached files into the correct tables?

Thanks!

-Sally

04.order_items_2012_May-Dec
05.order_item_refunds_May-Dec

Reply Forward

Solution

```
-- if you did this right, you should see 2,586 TOTAL order_item records
-- note: this includes records from the previous assignment
SELECT COUNT(*) AS total_records FROM order_items;


-- if you did this right, you should see 169 TOTAL refund records
-- note: this includes records from the previous assignment
-- note: if you have more, you may have missed the DELETE assignment
SELECT COUNT(*) AS total_records FROM order_item_refunds;
```

TEST YOUR SKILLS: **IMPORT REMAINING 2012 DATA**

LOAD DATA

- Another method for importing large datasets is to use LOAD DATA
- One major benefit is that using LOAD DATA can be faster when your data sets get large
- Another advantage is that Engineers can automate scripts that contain LOAD DATA scripts

MySQL LOAD DATA SYNTAX

13.2.7 LOAD DATA Statement

```
LOAD DATA
    [LOW_PRIORITY | CONCURRENT] [LOCAL]
    INFILE 'file_name'
    [REPLACE | IGNORE]
    INTO TABLE tbl_name
    [PARTITION (partition_name [, partition_name] ...)]
    [CHARACTER SET charset_name]
    [{FIELDS | COLUMNS}
        [TERMINATED BY 'string']
        [[OPTIONALLY] ENCLOSED BY 'char']
        [ESCAPED BY 'char']
    ]
    [LINES
        [STARTING BY 'string']
        [TERMINATED BY 'string']
    ]
    [IGNORE number {LINES | ROWS}]
    [(col_name_or_user_var
        [, col_name_or_user_var] ...)]
    [SET col_name={expr | DEFAULT}
        [, col_name={expr | DEFAULT}] ...]
```

NOT FOR EVERYONE!

LOAD DATA is not enabled by default, and enabling it requires editing your configuration file using command line. This is too technical for many in our audience, so we are not going to teach it in detail.

ALTER, UPDATE, & KEYS

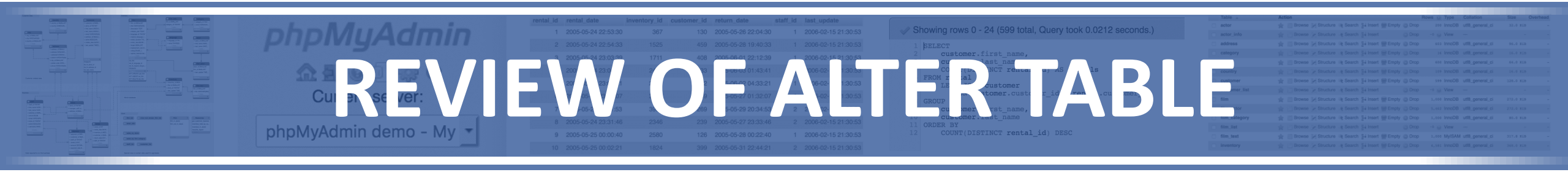
REVIEW OF ALTER, UPDATE, AND KEY RELATIONSHIPS

1. First, we'll review the ALTER TABLE statement used for adding and removing columns
2. Then, we'll review using UPDATE to SET values of records already in the database
3. Next, we'll review Cardinality and Primary Key to Foreign Key relationships
4. After our review is complete, we'll use each of these concepts to update the database as the business rolls out a new product and data tracking needs become more complex

THIS IS REVIEW!

We covered the basics of ALTER TABLE, UPDATE and Primary and Foreign Key relationships in our MySQL Database Administration for Beginners course. If you already have these concepts covered, feel free to skip ahead.

REVIEW OF ALTER TABLE



THIS IS REVIEW!

We covered the basics of modifying tables to add or delete columns using ALTER TABLE in our MySQL Database Administration for Beginners course. If you already have this concept covered, feel free to skip this lecture.

ALTER TABLE W/ CODE

- We can use SQL code to alter tables by adding and dropping columns
- We specify **ALTER TABLE** tablename, and then we can either **ADD** or **DROP** a column
- When we **ADD** a column, we must also specify its data type
- Optionally when we **ADD** a column, we may specify where it should appear, by using **FIRST** or **AFTER**

MySQL QUERY IN ACTION:

```
SELECT * FROM customer_purchases; -- just to see what's here to start

ALTER TABLE customer_purchases
DROP COLUMN customer_id; -- maybe this needs to get dropped for privacy

ALTER TABLE customer_purchases
ADD COLUMN purchase_amount DECIMAL(6,2) AFTER customer_purchase_id;
-- after is optional

SELECT * FROM customer_purchases; -- what's here at the end
```

ACTION OUTPUT:

Result Grid Filter Rows: Search Export:

customer_purchase_id	purchase_amount	inventory_id	purchased_at
1	NULL	3	NULL
2	NULL	2	NULL
3	NULL	4	NULL
4	NULL	7	NULL
5	NULL	5	NULL
6	NULL	1	NULL
7	NULL	6	NULL
8	NULL	8	NULL
9	NULL	9	NULL
10	NULL	3	NULL
11	NULL	2	NULL
12	NULL	6	NULL

REVIEW OF UPDATING RECORDS

THIS IS REVIEW!

We covered the basics of updating records in our MySQL Database Administration for Beginners course.
If you already have this concept covered, feel free to skip this lecture.

UPDATING RECORDS

- We can update the values of one or more records using an **UPDATE** statement
- Our **UPDATE** statement will always require a **SET** clause, which tells the server what values to set
- In most cases, we will also use a **WHERE** clause (optional) to specify which records to update. If we do not use a **WHERE** clause, the **UPDATE** will process on all records

MySQL QUERY IN ACTION:

```
UPDATE inventory
SET number_in_stock = 0 -- these are sold out
WHERE inventory_id = 9; -- only do this for id #9
```

(ANOTHER) MySQL QUERY IN ACTION:

```
-- this time we'll do it for two records simultaneously!
UPDATE inventory
SET number_in_stock = 0 -- these are sold out
WHERE inventory_id IN(1,9); -- update 2 records at once
```

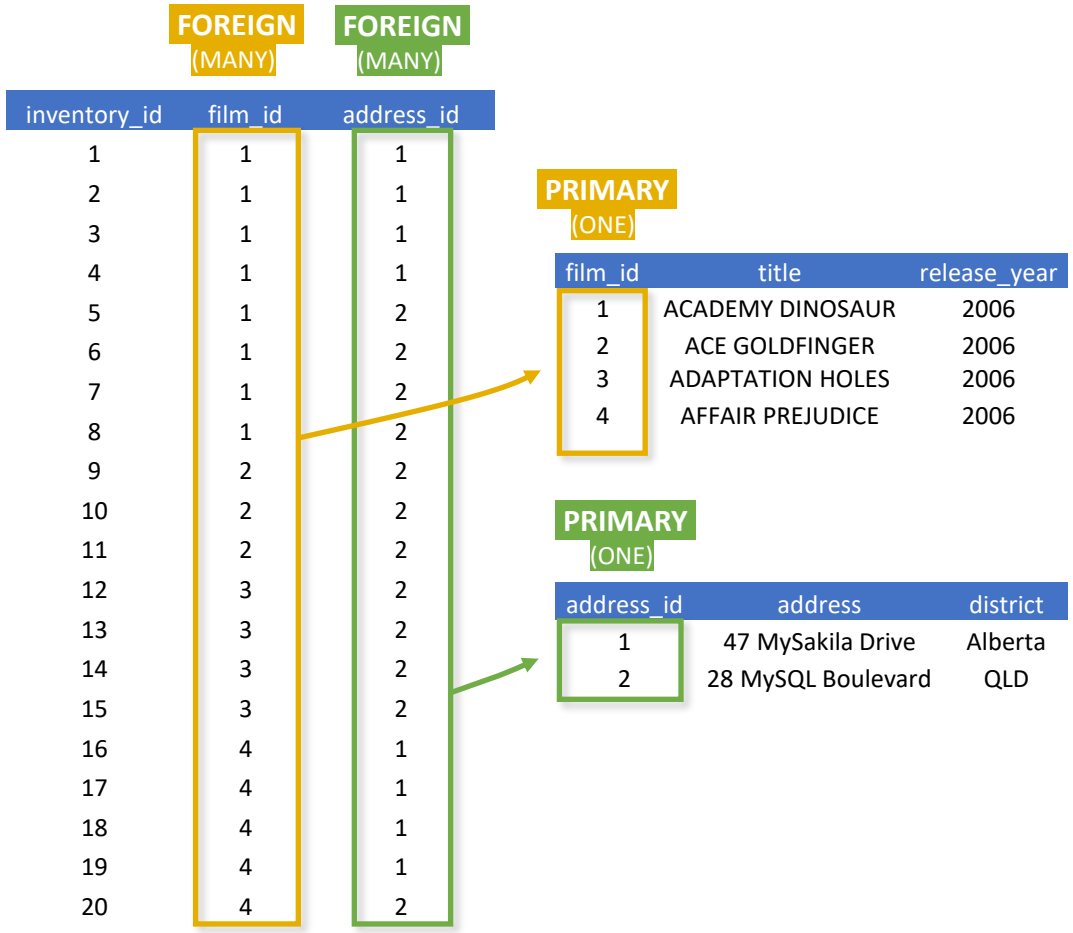
REVIEW OF TABLE RELATIONSHIPS

THIS IS REVIEW!

We covered the basics of table relationships in our MySQL Database Administration for Beginners course.
If you already have this concept covered, feel free to skip this lecture.

TABLE **RELATIONSHIPS** & **CARDINALITY**

Cardinality refers to the **uniqueness of values** in a column (*or attribute*) of a table and is commonly used to describe how two tables relate (***one-to-one***, ***one-to-many***, *or* ***many-to-many***). For now, here are the key points to grasp:



- **Primary keys** are **unique**
  - *They **cannot** repeat, so there is only **one** instance of each primary key value in a column*
- **Foreign keys** are **non-unique**
  - *They **can** repeat, so there may be **many** instances of each foreign key value in a column*
- We can create a **one-to-many** relationship by connecting a **foreign key** in one table to a **primary key** in another

NEW MESSAGE
January 05, 2013

From: **Sally Bleu** *(CEO)*

Subject: **New table to track products**

Hey,

Tomorrow we're launching a new product called The Forever Love Bear to complement The Original Mr. Fuzzy.

Could you please create a **products** table in the database? Track when they launched (2012-03-09 at 9am and 2013-01-06 at 1pm, respectively), the product name, and assign an id so we can link to other tables later.

Thanks!
-Sally

Reply Forward

Result Preview

```
-- first, create a products table with columns...
    -- product_id
    -- created_at
    -- product_name

-- then, populate it with 2 records like this...
```

product_id	created_at	product_name
1	2012-03-19 09:00:00	The Original Mr. Fuzzy
2	2013-01-06 13:00:00	The Forever Love Bear

TEST YOUR SKILLS: **ADDING A NEW PRODUCT**

NEW MESSAGE
January 05, 2013

From: **Sally Bleu** *(CEO)*

Subject: **New table to track products**

Hey,

Tomorrow we're launching a new product called The Forever Love Bear to complement The Original Mr. Fuzzy.

Could you please create a **products** table in the database? Track when they launched (2012-03-09 at 9am and 2013-01-06 at 1pm, respectively), the product name, and assign an id so we can link to other tables later.

Thanks!
-Sally

Reply Forward

Solution Queries

```
CREATE TABLE products (
    product_id BIGINT,
    created_at DATETIME,
    product_name VARCHAR(50),
    PRIMARY KEY (product_id)
);

INSERT INTO products VALUES
(1,'2012-03-19 09:00:00','The Original Mr. Fuzzy'),
(2,'2013-01-06 13:00:00','The Forever Love Bear');
```

product_id	created_at	product_name
1	2012-03-19 09:00:00	The Original Mr. Fuzzy
2	2013-01-06 13:00:00	The Forever Love Bear

TEST YOUR SKILLS: **ADDING A NEW PRODUCT**

NEW MESSAGE
January 06, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Add product data to item sales**

Good morning,

Later today, we'll have multiple products selling, I would love to be able to tie our **order_items** data to the product sold.

Can you please add **product_id** to the **order_items** table?

Thanks for the help!
-Sally

Reply Forward

Result Preview

```
-- don't worry about back-populating
-- for this assignment, just add the column
```

```
8    -- you should see a product_id column in order_items
9 •  SELECT * FROM order_items;
```

100% 1:4

Result Grid Filter Rows: Search Edit: Export/Impor

order_item_id	create_at	order_id	price_usd	cogs_usd	website_session_id	product_id
1	2012-03-19 10:42:46	1	49.99	19.49	20	NULL
2	2012-03-19 19:27:37	2	49.99	19.49	104	NULL
3	2012-03-20 06:44:45	3	49.99	19.49	147	NULL
4	2012-03-20 09:41:45	4	49.99	19.49	160	NULL
5	2012-03-20 11:28:15	5	49.99	19.49	177	NULL
6	2012-03-20 16:12:47	6	49.99	19.49	232	NULL
7	2012-03-20 17:03:41	7	49.99	19.49	241	NULL
8	2012-03-20 23:35:27	8	49.99	19.49	295	NULL
9	2012-03-21 02:35:01	9	49.99	19.49	304	NULL
10	2012-03-21 06:45:58	10	49.99	19.49	317	NULL

TEST YOUR SKILLS: **ADDING PRODUCT TO ORDER_ITEMS**

NEW MESSAGE
January 06, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Add product data to item sales**

Good morning,

Later today, we'll have multiple products selling, I would love to be able to tie our **order_items** data to the product sold.

Can you please add **product_id** to the **order_items** table?

Thanks for the help!
-Sally

Reply Forward

Solution Query

```
ALTER TABLE order_items
ADD product_id BIGINT;
```

```
8    -- you should see a product_id column in order_items
9 •  SELECT * FROM order_items;
```

100% 1:4

Result Grid Filter Rows: Search Edit: Export/Impor

order_item_id	create_at	order_id	price_usd	cogs_usd	website_session_id	product_id
1	2012-03-19 10:42:46	1	49.99	19.49	20	NULL
2	2012-03-19 19:27:37	2	49.99	19.49	104	NULL
3	2012-03-20 06:44:45	3	49.99	19.49	147	NULL
4	2012-03-20 09:41:45	4	49.99	19.49	160	NULL
5	2012-03-20 11:28:15	5	49.99	19.49	177	NULL
6	2012-03-20 16:12:47	6	49.99	19.49	232	NULL
7	2012-03-20 17:03:41	7	49.99	19.49	241	NULL
8	2012-03-20 23:35:27	8	49.99	19.49	295	NULL
9	2012-03-21 02:35:01	9	49.99	19.49	304	NULL
10	2012-03-21 06:45:58	10	49.99	19.49	317	NULL

TEST YOUR SKILLS: **ADDING PRODUCT TO ORDER_ITEMS**

NEW MESSAGE
January 07, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Back-populate sales with product_id**

Hey there,

I noticed that all records in **order_items** show NULL for **product_id**.

All of the sales reflected in the database are for product 1, so could you update the records to reflect that? Then we'll have perfect data to use in the future.

Thanks for the help!
-Sally

Reply Forward

Result Preview

```
5    -- product_id should be 1 for all records
6 •  SELECT * FROM order_items;
7
```

100% 1:11

Result Grid Filter Rows: Search Edit: Export/Import

order_item_id	create_at	order_id	price_usd	cogs_usd	website_session_id	product_id
1	2012-03-19 10:42:46	1	49.99	19.49	20	1
2	2012-03-19 19:27:37	2	49.99	19.49	104	1
3	2012-03-20 06:44:45	3	49.99	19.49	147	1
4	2012-03-20 09:41:45	4	49.99	19.49	160	1
5	2012-03-20 11:28:15	5	49.99	19.49	177	1
6	2012-03-20 16:12:47	6	49.99	19.49	232	1
7	2012-03-20 17:03:41	7	49.99	19.49	241	1
8	2012-03-20 23:35:27	8	49.99	19.49	295	1
9	2012-03-21 02:35:01	9	49.99	19.49	304	1
10	2012-03-21 06:45:58	10	49.99	19.49	317	1

TEST YOUR SKILLS: **UPDATING ORDER_ITEMS**

NEW MESSAGE
January 07, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Back-populate sales with product_id**

Hey there,

I noticed that all records in **order_items** show NULL for **product_id**.

All of the sales reflected in the database are for product 1, so could you update the records to reflect that? Then we'll have perfect data to use in the future.

Thanks for the help!
-Sally

Reply Forward

Solution Query

```
1  UPDATE order_items
2  SET product_id = 1
3  WHERE order_item_id > 0;
4
5  -- product_id should be 1 for all records
6  SELECT * FROM order_items;
7
```

100% 1:11

Result Grid Filter Rows: Search Edit: Export/Import

order_item_id	create_at	order_id	price_usd	cogs_usd	website_session_id	product_id
1	2012-03-19 10:42:46	1	49.99	19.49	20	1
2	2012-03-19 19:27:37	2	49.99	19.49	104	1
3	2012-03-20 06:44:45	3	49.99	19.49	147	1
4	2012-03-20 09:41:45	4	49.99	19.49	160	1
5	2012-03-20 11:28:15	5	49.99	19.49	177	1
6	2012-03-20 16:12:47	6	49.99	19.49	232	1
7	2012-03-20 17:03:41	7	49.99	19.49	241	1
8	2012-03-20 23:35:27	8	49.99	19.49	295	1
9	2012-03-21 02:35:01	9	49.99	19.49	304	1
10	2012-03-21 06:45:58	10	49.99	19.49	317	1

TEST YOUR SKILLS: **UPDATING ORDER_ITEMS**

NEW MESSAGE
January 07, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Primary and foreign keys?**

Hey there,

I would like to make sure the newly-related **order_items** and **products** tables have the right relationships specified in the database.

Can you set up the proper primary and foreign key relationships between those two tables?

Thank you!
-Sally

Reply Forward

Helpful Hint

```
-- TIP: you can do this with code or with the UI
```

TEST YOUR SKILLS: **PRIMARY AND FOREIGN KEYS**

1 **NEW MESSAGE**

January 07, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Primary and foreign keys?**

Hey there,

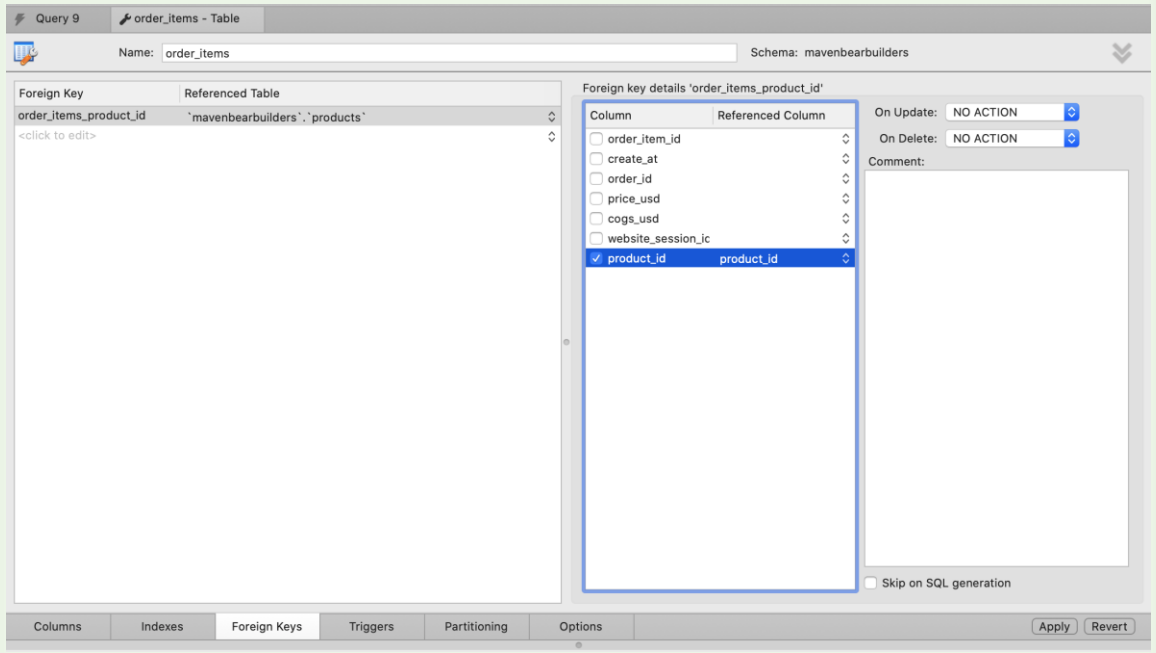
I would like to make sure the newly-related **order_items** and **products** tables have the right relationships specified in the database.

Can you set up the proper primary and foreign key relationships between those two tables?

Thank you!
-Sally

Reply Forward

Solution



TEST YOUR SKILLS: **PRIMARY AND FOREIGN KEYS**

NEW MESSAGE
April 1, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Help get order data into the database**

Hey there,

Now that you've done all the work to get our products and order_items tables synced up, let's import the attached Q1 data into **order_items** and **order_item_refunds**.

Really curious to start digging into sales trends!

Thanks!
-Sally

06.order_items_2013_Jan-Mar
07.order_item_refunds_2013_Jan-Mar

Reply Forward

Helpful Hint

```
-- TIP: remember order matters because of your keys

-- if you've done this correctly, there should be 3,859 records
SELECT COUNT(*) AS total_records FROM order_items ;

-- if you've done this correctly, there should be 234 records
SELECT COUNT(*) AS total_records FROM order_item_refunds ;
```

TEST YOUR SKILLS: **IMPORT 2013 Q1 DATA**

NEW MESSAGE
April 1, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Help get order data into the database**

Hey there,

Now that you've done all the work to get our products and order_items tables synced up, let's import the attached Q1 data into **order_items** and **order_item_refunds**.

Really curious to start digging into sales trends!

Thanks!
-Sally

06.order_items_2013_Jan-Mar
07.order_item_refunds_2013_Jan-Mar

Reply Forward

Solution

```
-- See solution video for more detail
```

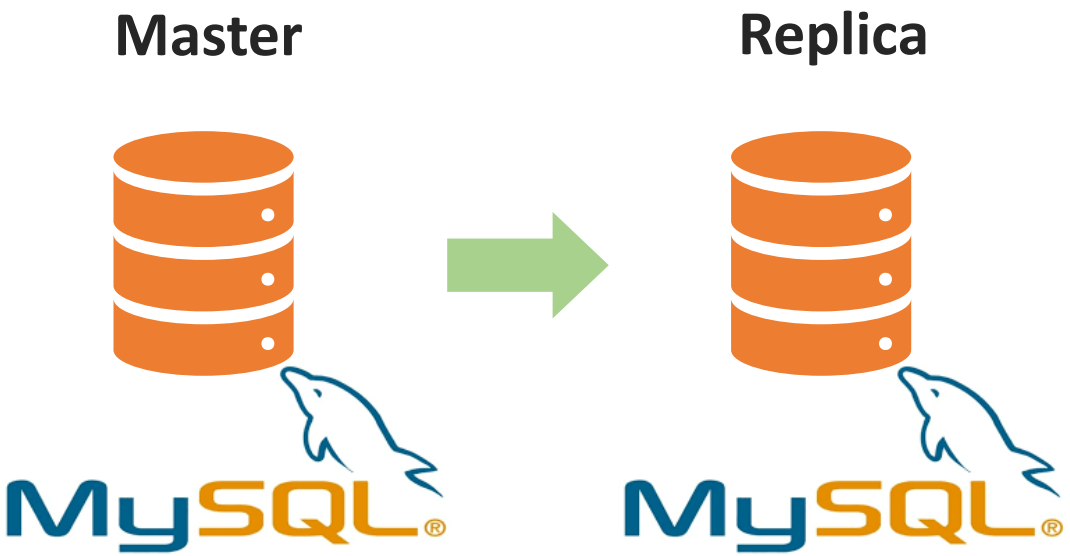
```
-- if you've done this correctly, there should be 3,859 records
SELECT COUNT(*) AS total_records FROM order_items ;

-- if you've done this correctly, there should be 234 records
SELECT COUNT(*) AS total_records FROM order_item_refunds ;
```

TEST YOUR SKILLS: **IMPORT 2013 Q1 DATA**

DATABASE REPLICATION

INTRODUCTION TO REPLICATION

Replication enables us to store the same data on two or more servers by creating copies, known as replicas

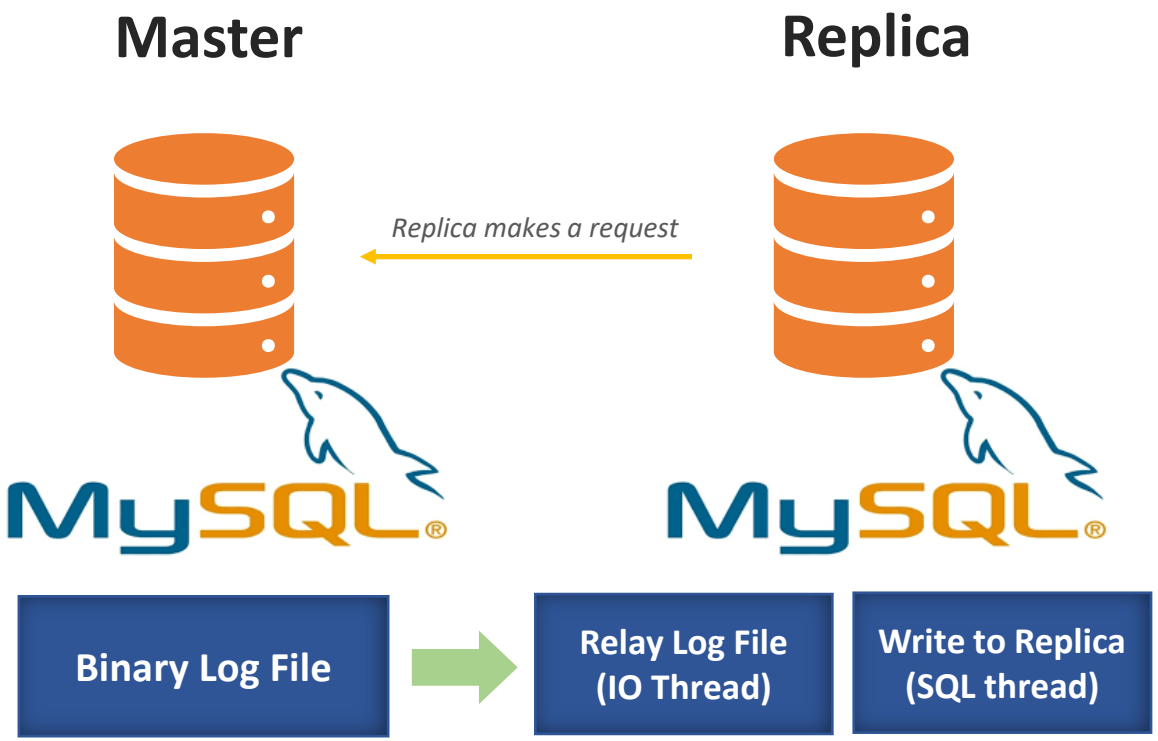


BENEFITS & USES:

- Replication can serve as a form of backup
- Creating a read only replica for reporting purposes keeps analysis out of production
- In certain cases, splitting the load between two servers can improve application performance

ROW-BASED BINARY LOG FILE REPLICATION

The first method of Replication we are going to focus on is **Row-based Replication using Binary Log files**

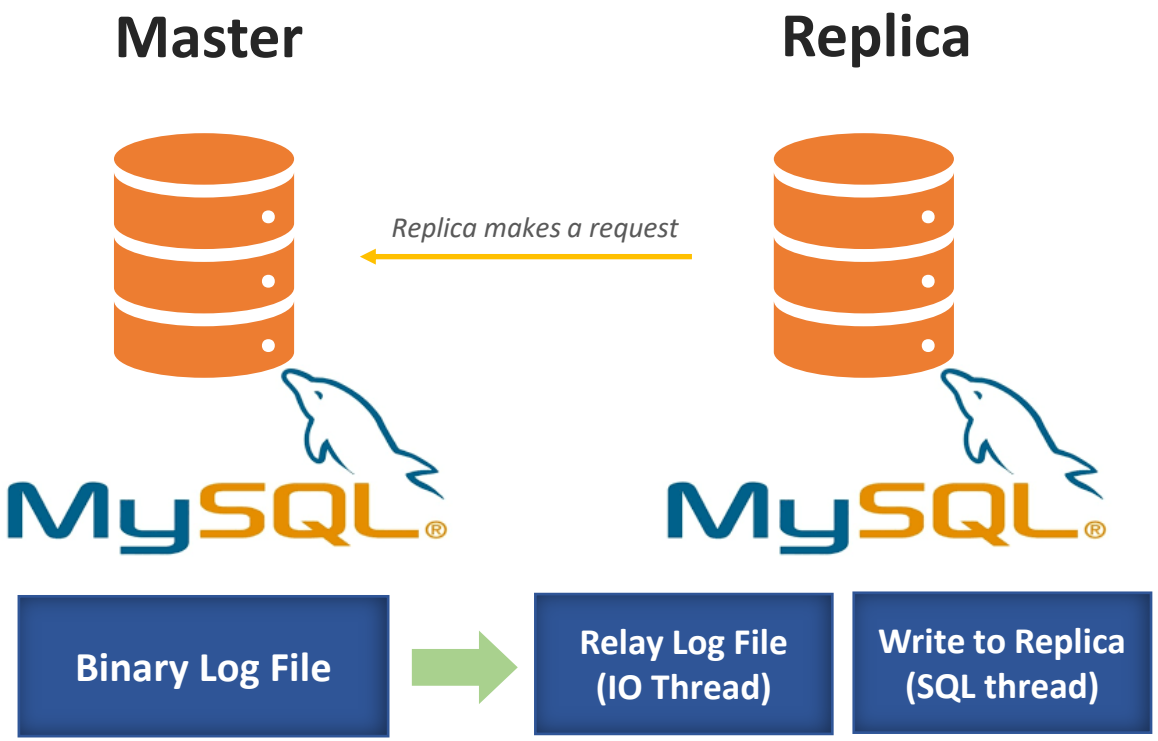


IMPORTANT POINTS

- Binary Log files record changes made to records of data (think “before and after”)
- Replication typically (not always) happens asynchronously, so the Replica can lag behind
- Row-based replication is typically faster than statement-based replication, but can be harder to audit because it does not contain SQL statements

STATEMENT-BASED BINARY LOG FILE REPLICATION

The next method of Replication we are going to discuss is **Statement-based Replication using Binary Log files.** Much of this will feel similar to row-based replication.



IMPORTANT POINTS

- Binary Log files record the actual SQL statements that make changes to the database
- Statement-based replication is typically easier to read and audit compared to row-based
- Non-deterministic queries can be big trouble

ROW-BASED vs STATEMENT-BASED REPLICATION

	ROW-BASED	STATEMENT-BASED
Changing Few Rows	✓ **PRO:** statements which update very few rows will execute very quickly	X **CON:** statements which update very few rows will usually perform more slowly than Row-Based
Changing Many Rows	X **CON:** statements which update a large number of rows will perform very slowly	✓ **PRO:** statements updating a large number of rows can still execute quickly
Consistent Data	✓ **PRO:** does not encounter issues with non-deterministic queries like statement-based	X **CON:** non-deterministic queries can spell trouble (example, INSERT w/ auto-incrementing PK)
Auditing Changes	X **CON:** harder to audit, because you only see changes, not the statements themselves	✓ **PRO:** easier to audit, because you see the statements themselves
Handling Triggers	✓ **PRO:** no problem handling stored routines and triggers	X **CON:** can create problems with stored routines and triggers.

BACKUP & RECOVERY

INTRODUCTION TO BACKUP & RECOVERY

We use Backups to **make sure our data is protected and recoverable** in the event of loss

Physical vs **Logical** Backups

- **Physical** backups store the raw data in a file, where **Logical** backups store the SQL statements needed to recreate the database and populate it
- With Logical backups, (ex: MySQL Dump), you'll be storing CREATE and INSERT statements

Online vs **Offline** Backups

- **Online** backups occur while the server is running. The advantage you don't have to take the server down, so it won't interfere with other clients using the server.
- **Offline** backups happen when the server is stopped. Simpler, but other clients won't have access either.

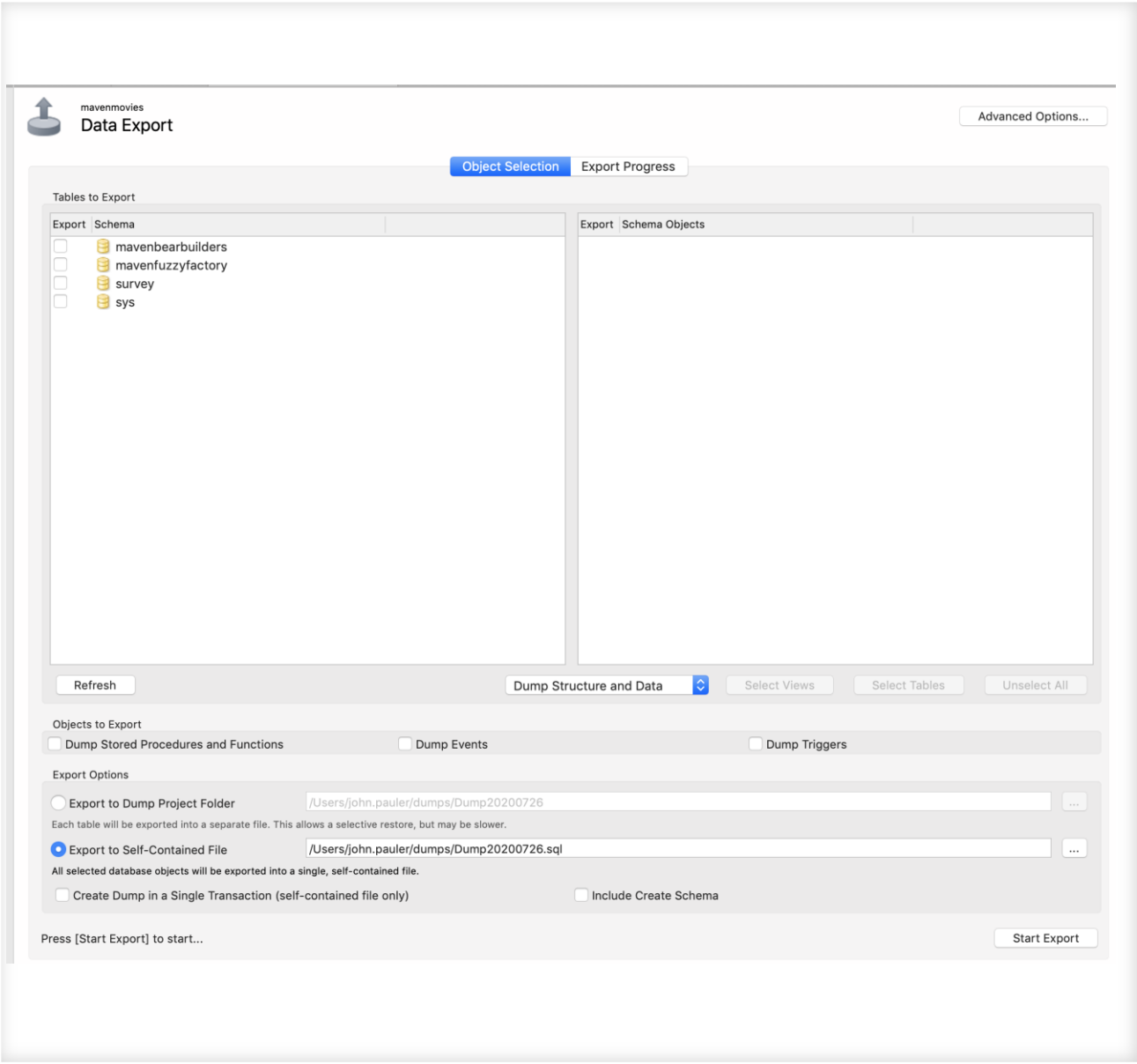
Local vs **Remote** Backups

- **Local** backups happen on the same host that the MySQL server is running
- **Remote** backups are written somewhere else. This could be another host, a local machine, etc.

mysqldump

- mysqldump is a database backup method we can use to make sure we can restore data if we lose it
- You can initiate mysqldump from command line, or in Workbench using the wizard
- When you use mysqldump, the output will be a .sql file that contains the create statements and insert statements for all of the tables you have dumped

MySQL WORKBENCH IN ACTION:



NEW MESSAGE
April 10, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Need a Backup & Recover Plan**

Good morning,

One of our board members is upset because we haven't taken any steps to backup the data we're aggregating on the MySQL Server.

I would like you to put together a written proposal for everything we should be doing in terms of replication, backup, and recovery. And take a snapshot of the data while you're at it. Thanks!

-Sally

Reply Forward

Helpful Hints

```
-- articulate the risks of data loss

-- talk about how you will mitigate them

-- for the snapshot, your output should be a .sql file
    -- this file will recreate your database
    -- if you don't know this part, review prior lectures
```

TEST YOUR SKILLS: **BACKUP & RECOVERY**

NEW MESSAGE

April 10, 2013

From: **Sally Bleu** *(CEO)*

Subject: **Need a Backup & Recover Plan**

Good morning,

One of our board members is upset because we haven't taken any steps to backup the data we're aggregating on the MySQL Server.

I would like you to put together a written proposal for everything we should be doing in terms of replication, backup, and recovery. And take a snapshot of the data while you're at it. Thanks!

-Sally

Reply Forward

Solution Query

```
-- Ideally, you would have addressed each of these...

   -- data loss risks

   -- statement-based vs row-based replication

   -- creating a backup snapshot (mysqldump)
```

TEST YOUR SKILLS: **BACKUP & RECOVERY**

INTRODUCING THE **MID COURSE PROJECT**

THE SITUATION

Maven Bear Builders has been up and running for a little over a year. You and your CEO have made some improvements to the database, but as the business continues to change, she needs more help tweaking the structure and importing additional data sets.

THE OBJECTIVE

Use SQL to:

- Import additional data into the bearbuilders database
- Enhance the data structure to accommodate new tracking needs for the business

As a Database Administrator, part of your job is executing on specific tasks like altering tables. Another major focus area is staying on top of things like backup, recovery, and database security. Use any opportunities you see as chance to flex your muscle as a thought leader in these areas!

MID COURSE PROJECT QUESTIONS

1

~ 0:38

Import **Q2 orders and refunds** into the database using the files below:

08.order_items_2013_Apr-June
09.order_item_refunds_2013_Apr-Jun

2

~ 4:20

Next, help update the structure of the **order_items** table:

- The company is going to start cross-selling products and will want to track whether each item sold is the **primary** item (the first one put into the user's shopping cart) or a **cross-sold** item
- Add a binary column to the **order_items** table called **is_primary_item**

3

~ 6:15

Update all *previous* records in the **order_items** table, setting **is_primary_item = 1** for all records

- Up until now, all items sold were the primary item (since cross-selling is new)
- **Confirm this change has executed successfully**

MID COURSE PROJECT QUESTIONS

4

~ 9:00

Add two new products to the **products** table, then **import the remainder of 2013 orders and refunds**, using the product details and files shown below:

10.order_items_2013_Jul-Dec
11.order_item_refunds_2013_Jul-Dec

product_id	created_at	product_name
3	2013-12-12 09:00:00	The Birthday Sugar Panda
4	2014-02-05 10:00:00	The Hudson River Mini bear

5

~ 13:38

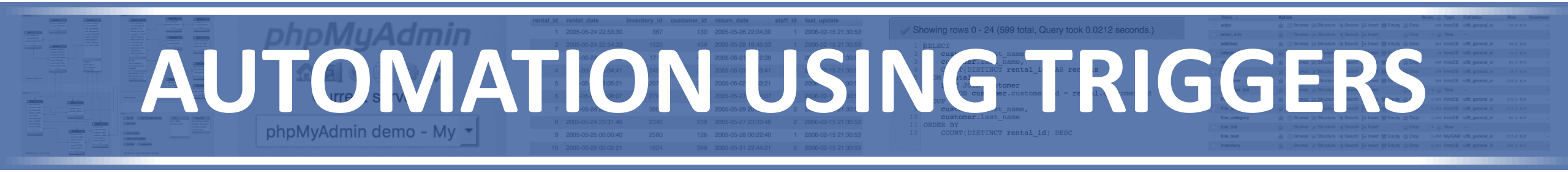
Your CEO would like to make sure the database has a high degree of data integrity and avoid potential issues as more people start using the database. If you see any opportunities to **ensure data integrity by using constrains like *NON-NULL***, add them to the relevant columns in the tables you have created.

6

~ 17:30

One of the company's board advisors is pressuring your CEO on data risks and making sure she has a great backup and recovery plan. **Prepare a report on possible risks for data loss and steps the company can take to mitigate these concerns.**

AUTOMATION USING TRIGGERS



THIS FIRST TRIGGERS VIDEO IS REVIEW!

We covered the basics of TRIGGERS in our MySQL Database Administration for Beginners course. If you already have these concepts covered, feel free to skip ahead. If you want review of this important concept, check it out!

TRIGGERS REVIEW

- MySQL allows us to create Triggers, where we can prescribe certain actions on a table to trigger one or more other actions to occur
- We may prescribe that our triggered action occurs either BEFORE or AFTER an INSERT, UPDATE, or a DELETE
- Triggers are a very common way to make sure related tables remain in sync as they are updated over time

MySQL Workbench in Action

```
CREATE TRIGGER purchaseUpdateInventory
AFTER INSERT ON customer_purchases
FOR EACH ROW
    UPDATE inventory
        -- subtracting an item for each purchase
        SET number_in_stock = number_in_stock - 1
    WHERE inventory_id = NEW.inventory_id;

INSERT INTO customer_purchases VALUES
(13,NULL,3,NULL), -- inventory_id = 3, velour jumpsuit
(14,NULL,4,NULL) -- inventory_id = 4, house slippers
;

SELECT * FROM inventory;
```

*Result Preview (SELECT * FROM inventory)*

Result Grid — Filter Rows: Search

inventory_id	item_name	number_in_stock
1	fur coat	0
2	mocassins	4
3	velour jumpsuit	12
4	house slippers	6
5	brown leather jacket	3
6	broken keyboard	6
8	kneeboard	2
9	pro wings sneakers	0
10	wolf skin hat	1
11	fur fox skin	1
12	plaid button up shirt	8
13	flannel zebra jammies	6

Result Grid — Filter Rows: Search

inventory_id	item_name	number_in_stock
1	fur coat	0
2	mocassins	4
3	velour jumpsuit	11
4	house slippers	5
5	brown leather jacket	3
6	broken keyboard	6
8	kneeboard	2
9	pro wings sneakers	0
10	wolf skin hat	1
11	fur fox skin	1
12	plaid button up shirt	8
13	flannel zebra jammies	6

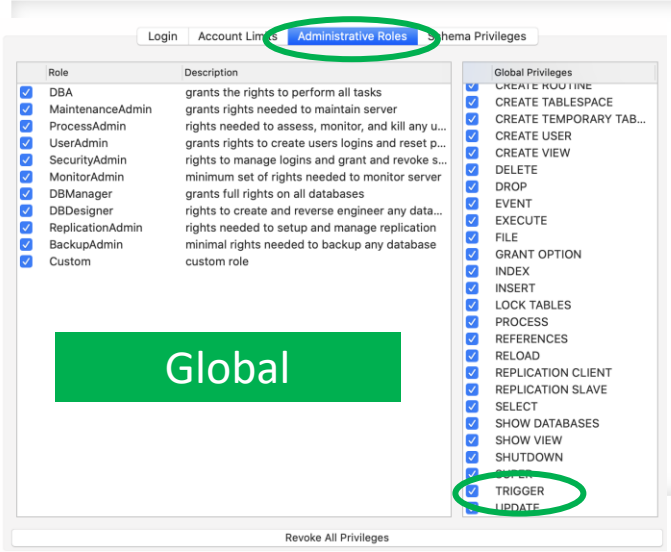
TRIGGERS DEEP DIVE

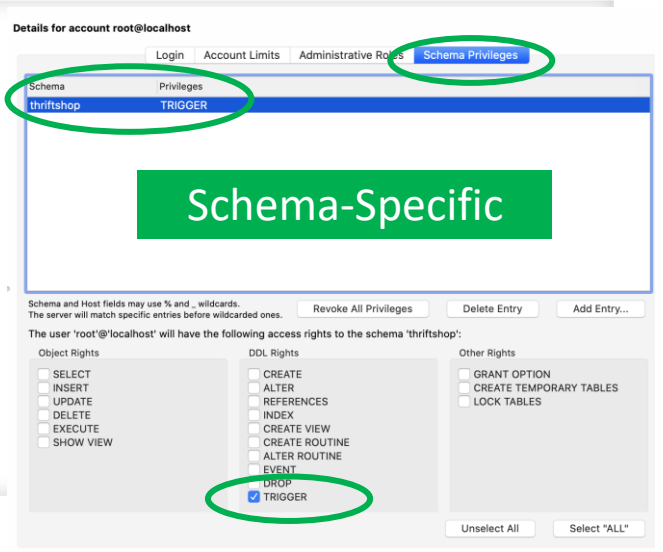
- MySQL supports only row-level Triggers, however in other syntax languages, statement-level Triggers are supported
- MySQL does not support ALTER TRIGGER, so to modify an existing Trigger, you'll use DROP TRIGGER and then CREATE TRIGGER
- In order to CREATE, DROP or view the Triggers on a table, you will need to have the Triggers privilege enabled for that table

MySQL Workbench in Action

```
1    -- this removes the Trigger
2    -- you might do this to stop using the Trigger
3    -- or you could DROP it to recreate a modified Trigger
4    DROP TRIGGER purchaseUpdateInventory;
```

Trigger Privileges





TRIGGERS DEEP DIVE

- In some cases, using Triggers with BEFORE and AFTER will produce different results
- Multiple Triggers can exist on the same table. By default, they will fire in creation order. This can be modified using FOLLOWS or PROCEDES
- To see Triggers in a schema, use SHOW TRIGGERS. Or you can query the INFORMATION_SCHEMA to see all triggers in your instance

MySQL Workbench in Action

```
CREATE TRIGGER purchaseUpdatePurchaseSummary_before
BEFORE INSERT ON customer_purchases
FOR EACH ROW
UPDATE purchase_summary
   SET purchases_excluding_last = (
       SELECT COUNT(customer_purchase_id)
       FROM customer_purchases
       WHERE customer_purchases.customer_id = purchase_summary.customer_id
           )
WHERE customer_id = NEW.customer_id
AND purchase_summary_id > 0; -- this is just to handle safe update mode;
```

```
CREATE TRIGGER purchaseUpdatePurchaseSummary_after
AFTER INSERT ON customer_purchases
FOR EACH ROW
UPDATE purchase_summary
   SET total_purchases = (
       SELECT COUNT(customer_purchase_id)
       FROM customer_purchases
       WHERE customer_purchases.customer_id = purchase_summary.customer_id
           )
WHERE customer_id = NEW.customer_id
AND purchase_summary_id > 0; -- this is just to handle safe update mode;
```

Check Out Active Triggers

```
SHOW TRIGGERS;
```

100% 7:12 1 error found

Result Grid Filter Rows: Search Export:

Trigger	Event	Table	Statement	Timing	Created	sql_mode
purchaseUpdateInventory	INSERT	customer_purchases	UPDATE inventory -- subtracting an item for…	AFTER	2020-08-15 12:02:13.19	ONLY_FULL_GROUP_BY,

```
SELECT * FROM INFORMATION_SCHEMA.TRIGGERS;
```

100% 1:9 1 error found

Result Grid Filter Rows: Search Export:

TRIGGER_CATALOG	TRIGGER_SCHEMA	TRIGGER_NAME	EVENT_MANIPULATION	EVENT_OBJECT_CATALOG	EVENT_OBJECT_SCHEMA	EVEN
def	sys	sys_config_insert_set_user	INSERT	def	sys	sys_c
def	sys	sys_config_update_set_user	UPDATE	def	sys	sys_c
def	thriftshop	purchaseUpdateInventory	INSERT	def	thriftshop	custor

TRIGGERS DEEP DIVE

- When creating a Trigger, BEFORE or AFTER is required, as is INSERT, UPDATE, or DELETE, naming the table, and including a valid SQL statement
- FOLLOWS or PRECEDES are optional
- Specifying OLD or NEW when referencing columns in your statement is an optional modifier
- Note that INSERT Triggers do not work with OLD (no OLD value exists) and DELETE Triggers won't work with NEW

MySQL Workbench in Action

RECAP: Triggers Syntax w/ Optional Modifiers

```
CREATE TRIGGER trigger_name
{BEFORE|AFTER}{INSERT|UPDATE|DELETE}
ON table_name FOR EACH ROW
{{FOLLOWS|PRECEDES} existing_trigger_name}
[[some statement we want to execute by trigger]]
{WHERE column_name = {OLD|NEW}.column_name}
```

NEW MESSAGE
January 03, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Create an order summary table**

Morning! Now that we're selling multiple products, it would be great to have a table summarizing full orders.

Can you create a table to capture **order_id**, a **created_at** timestamp, **website_session_id**, primary **product_id**, **# of items** purchased, **price** and **cogs** in USD?

Could you also back-populate the table using the records from our **order_items** table?

Thank you!
-Sally

Reply Forward

Helpful Hint

```sql
-- back-populating the orders table
INSERT INTO orders
SELECT
    order_id,
    MIN(created_at) AS created_at,
    MIN(website_session_id) AS website_session_id,
    SUM(CASE
        WHEN is_primary_item = 1 THEN product_id
        ELSE NULL
        END) AS primary_product_id,
    COUNT(order_item_id) AS items_purchased,
    SUM(price_usd) AS price_usd,
    SUM(cogs_usd) AS cogs_usd
FROM order_items
GROUP BY 1
ORDER BY 1;
```

TEST YOUR SKILLS: **ORDER SUMMARY DATA**

NEW MESSAGE
January 03, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Create an order summary table**

Morning! Now that we're selling multiple products, it would be great to have a table summarizing full orders.

Can you create a table to capture **order_id**, a **created_at** timestamp, **website_session_id**, primary **product_id**, **# of items** purchased, **price** and **cogs** in USD?

Could you also back-populate the table using the records from our **order_items** table?

Thank you!
-Sally

Reply Forward

Solution Query

```sql
-- back-populating the orders table
INSERT INTO orders
SELECT
    order_id,
    MIN(created_at) AS created_at,
    MIN(website_session_id) AS website_session_id,
    SUM(CASE
        WHEN is_primary_item = 1 THEN product_id
        ELSE NULL
        END) AS primary_product_id,
    COUNT(order_item_id) AS items_purchased,
    SUM(price_usd) AS price_usd,
    SUM(cogs_usd) AS cogs_usd
FROM order_items
GROUP BY 1
ORDER BY 1;
```

TEST YOUR SKILLS: **ORDER SUMMARY DATA**

NEW MESSAGE
January 05, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Automation to update orders table**

Hey there,

The new **orders** table you created is so helpful!

Next, would you be able to set up some automation so that anytime **order_items** records are inserted into the database, the **orders** table is updated as well?

Thank you!
-Sally

Reply Forward

Helpful Hints

```
-- You're going to want your trigger to insert a record into orders
-- You want this to happen whenever new order_items records are added
REPLACE INTO orders
SELECT
    order_id,
    MIN(created_at) AS created_at,
    MIN(website_session_id) AS website_session_id,
    SUM(CASE
        WHEN is_primary_item = 1 THEN product_id
        ELSE NULL
        END) AS primary_product_id,
    COUNT(order_item_id) AS items_purchased,
    SUM(price_usd) AS price_usd,
    SUM(cogs_usd) AS cogs_usd
FROM order_items
GROUP BY 1
ORDER BY 1;
```

TEST YOUR SKILLS: **AUTOMATED ORDERS TRIGGER**

NEW MESSAGE
January 05, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Automation to update orders table**

Hey there,

The new **orders** table you created is so helpful!

Next, would you be able to set up some automation so that anytime **order_items** records are inserted into the database, the **orders** table is updated as well?

Thank you!
-Sally

Reply Forward

Solution Query

```
CREATE TRIGGER insert_new_orders
AFTER INSERT ON order_items
FOR EACH ROW
    REPLACE INTO orders
    SELECT
        order_id,
        MIN(created_at) AS created_at,
        MIN(website_session_id) AS website_session_id,
        SUM(CASE
            WHEN is_primary_item = 1 THEN product_id
            ELSE NULL
            END) AS primary_product_id,
        COUNT(order_item_id) AS items_purchased,
        SUM(price_usd) AS price_usd,
        SUM(cogs_usd) AS cogs_usd
    FROM order_items
    WHERE order_id = new.order_id
    GROUP BY 1
    ORDER BY 1;
```

TEST YOUR SKILLS: **AUTOMATED ORDERS TRIGGER**

NEW MESSAGE
March 01, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Putting your automation to the test**

Hey,

It's time to see if your trigger to sync the **order_items** and **orders** tables is working correctly.

Why don't you go ahead and update the **order_items** and **order_item_refunds** tables with the attached data and we'll see how everything is working.

Thank you!
-Sally

12.order_items_2014_Jan-Feb
13.order_item_refunds_2014_Jan-Feb

Reply Forward

Result Preview

```sql
-- You should see 10033 records before trigger fires
SELECT COUNT(*) AS total_records FROM orders;

-- Then, test the trigger by inserting into order_items

-- TIP: after insert, find the max order_id
SELECT MAX(order_id) AS max_order_id
FROM order_items;

-- You should see a new row count that matches max_order_id
SELECT COUNT(*) AS total_records FROM orders;
```

TEST YOUR SKILLS: **TESTING YOUR TRIGGER**

NEW MESSAGE
March 01, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Putting your automation to the test**

Hey,

It's time to see if your trigger to sync the **order_items** and **orders** tables is working correctly.

Why don't you go ahead and update the **order_items** and **order_item_refunds** tables with the attached data and we'll see how everything is working.

Thank you!
-Sally

12.order_items_2014_Jan-Feb
13.order_item_refunds_2014_Jan-Feb

Reply Forward

Solution

```
-- See solution video for more detail
```

```sql
-- You should see 10033 records before trigger fires
SELECT COUNT(*) AS total_records FROM orders;

-- Then, test the trigger by inserting into order_items

-- TIP: after insert, find the max order_id
SELECT MAX(order_id) AS max_order_id
FROM order_items;

-- You should see a new row count that matches max_order_id
SELECT COUNT(*) AS total_records FROM orders;
```

TEST YOUR SKILLS: **TESTING YOUR TRIGGER**

VIEWS

- A view is like a “virtual table”; contents are defined by a query, and data is not physically created
- One advantage of views is the ability to aggregate data from multiple tables into one view.
- Another advantage is security – you can grant someone access to certain views, exclusively
- Finally, altering views will not affect any downstream systems

MySQL Workbench in Action

```sql
-- creating a view
CREATE VIEW country_averages AS
SELECT
    country,
    AVG(years_experience) AS yrs_experience,
    AVG(CASE WHEN is_manager = 'Yes' THEN 1 ELSE 0 END) AS pct_mgrs,
    AVG(CASE WHEN education_level = 'Masters' THEN 1 ELSE 0 END) AS pct_masters
FROM salary_survey
GROUP BY 1;

-- selecting from the view
SELECT * FROM country_averages;
```

Result Preview

country	yrs_experience	pct_mgrs	pct_masters
United States	23.0379	0.2613	0.1731
United Kingdom	10.1005	0.2487	0.1005
Germany	10.6379	0.3276	0.3448
Netherlands	9.7955	0.2727	0.1364
Canada	11.3556	0.2593	0.0815
Poland	7.0000	0.3333	0.7083
Ukraine	7.1250	0.5000	0.7500
India	6.4795	0.3973	0.3699
Portugal	8.6667	0.3333	0.2222
South Africa	13.0645	0.2581	0.0000
Greece	10.5000	0.6667	0.4167
Israel	9.2727	0.2727	0.1818
Spain	7.4118	0.4118	0.4118
Norway	9.6667	0.0000	0.0000
France	9.7333	0.2000	0.4667
Jersey	11.6000	0.0000	0.2000
Argentina	6.4000	0.2000	0.0000
Belgium	9.7273	0.0909	0.0909
Switzerland	11.0909	0.2727	0.3182
Ireland	11.3478	0.4348	0.2174
Italy	10.8571	0.4286	0.1905

NEW MESSAGE
March 07, 2014

From: **Brent Cheeseman** *(Marketing Lead)*

Subject: **Tying website activity to sales**

Hey data rockstar!

Sally tells me the great things you've done with our order data. I have this website session data that I would love to tie into that order data so we can better understand where sales are coming from.

Can you create a **website_sessions** table and help me import the attached files?

-Brent

14.website_sessions_2014_Jan
15.website_sessions_2014_Feb

Reply Forward

Helpful Hints

```
-- take a look at the datasets to determine table structure

-- then create the table and import the records
```

```
22    -- when you're done, there should be 31,110 records
23 •  SELECT COUNT(*) AS total_records FROM website_sessions;
```

100% 1:20

Result Grid Filter Rows: Search Export:

total_records
31110

TEST YOUR SKILLS: **TRACKING WEBSITE SESSIONS**

NEW MESSAGE
March 07, 2014

From: **Brent Cheeseman** *(Marketing Lead)*
Subject: **Tying website activity to sales**

Hey data rockstar!

Sally tells me the great things you've done with our order data. I have this website session data that I would love to tie into that order data so we can better understand where sales are coming from.

Can you create a **website_sessions** table and help me import the attached files?

-Brent

14.website_sessions_2014_Jan
15.website_sessions_2014_Feb

Reply Forward

Solution Query

```
CREATE TABLE website_sessions (
    website_session_id BIGINT,
    created_at DATETIME,
    user_id BIGINT,
    is_repeat_session INT,
    utm_source VARCHAR(50),
    utm_campaign VARCHAR(50),
    utm_content VARCHAR(50),
    device_type VARCHAR(50),
    http_referer VARCHAR(100),
    PRIMARY KEY (website_session_id)
);
```

TEST YOUR SKILLS: **TRACKING WEBSITE SESSIONS**

NEW MESSAGE
March 09, 2014

From: **Brent Cheeseman** *(Marketing Lead)*

Subject: **Reporting Views**

Hey, thanks for getting that **website_session** data imported!

Next, would you be able to create a view summarizing performance for January and February? I would like to see the **number of sessions** sliced by **year, month, utm_source**, and **utm_campaign** if possible.

Thanks!

-Brent

Reply Forward

Helpful Hints

```
15    -- the view should look like this...
16 •  SELECT * FROM monthly_sessions;
```

100% 18:14

Result Grid Filter Rows: Search

year	month	utm_source	utm_campaign	number_of_sessions
2014	1	gsearch	nonbrand	7500
2014	1	NULL	NULL	2724
2014	1	bsearch	nonbrand	1614
2014	1	gsearch	brand	1107
2014	1	bsearch	brand	262
2014	1	socialbook	pilot	1618
2014	2	bsearch	nonbrand	1646
2014	2	NULL	NULL	2819
2014	2	gsearch	nonbrand	8223
2014	2	gsearch	brand	1090
2014	2	socialbook	pilot	2237
2014	2	bsearch	brand	270

TEST YOUR SKILLS: **CREATING VIEWS FOR REPORTING**

NEW MESSAGE
March 09, 2014

From: **Brent Cheeseman** *(Marketing Lead)*

Subject: **Reporting Views**

Hey, thanks for getting that **website_session** data imported!

Next, would you be able to create a view summarizing performance for January and February? I would like to see the **number of sessions** sliced by **month, utm_source**, and **utm_campaign** if possible.

Thanks!

-Brent

Reply Forward

Solution Query

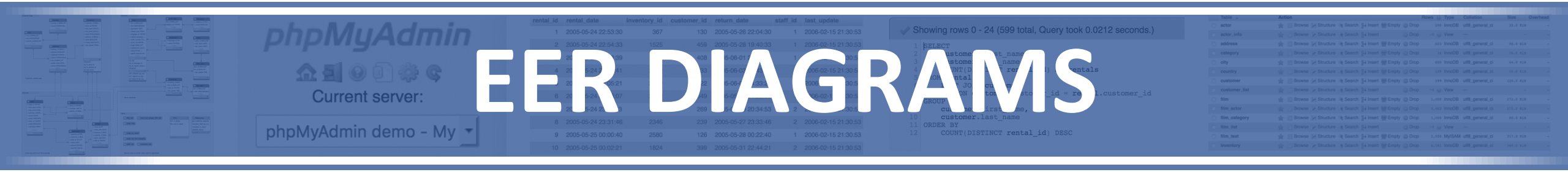
```
6  • CREATE VIEW monthly_sessions AS
7    SELECT
8        YEAR(created_at) AS year,
9        MONTH(created_at) AS month,
10       utm_source,
11       utm_campaign,
12       COUNT(website_session_id) AS number_of_sessions
13   FROM website_sessions
14   GROUP BY 1,2,3,4;
15
16 • SELECT * FROM monthly_sessions;
```

100% 27:16

Result Grid Filter Rows: Search Export:

year	month	utm_source	utm_campaign	number_of_sessions
2014	1	gsearch	nonbrand	7500
2014	1	NULL	NULL	2724
2014	1	bsearch	nonbrand	1614
2014	1	gsearch	brand	1107
2014	1	bsearch	brand	262
2014	1	socialbook	pilot	1618
2014	2	bsearch	nonbrand	1646
2014	2	NULL	NULL	2819
2014	2	gsearch	nonbrand	8223
2014	2	gsearch	brand	1090
2014	2	socialbook	pilot	2237
2014	2	bsearch	brand	270

TEST YOUR SKILLS: **CREATING VIEWS FOR REPORTING**



EER DIAGRAMS

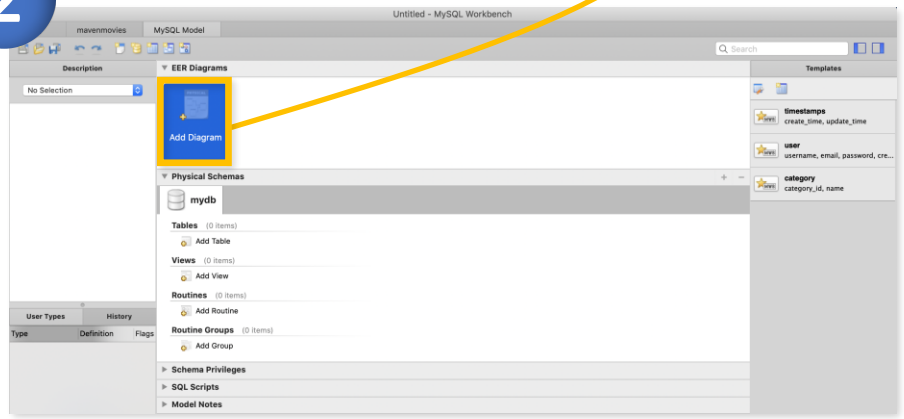
THIS FIRST PART IS REVIEW!

We covered the basics of EER Diagrams in our MySQL Database Administration for Beginners course. If you already have these concepts covered, feel free to skip ahead. If you want to review this important concept, watch the video.

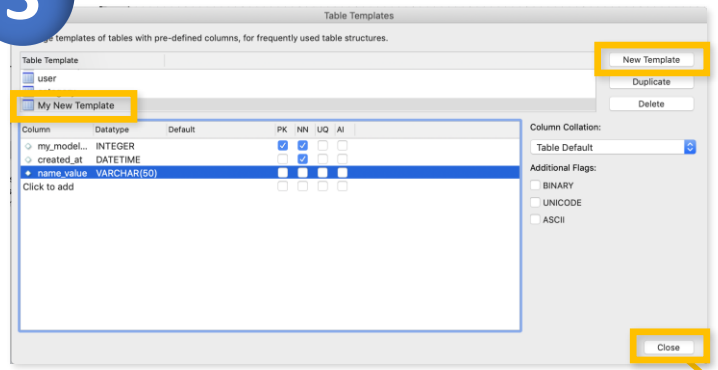
CREATE AN EER DIAGRAM FROM SCRATCH

1

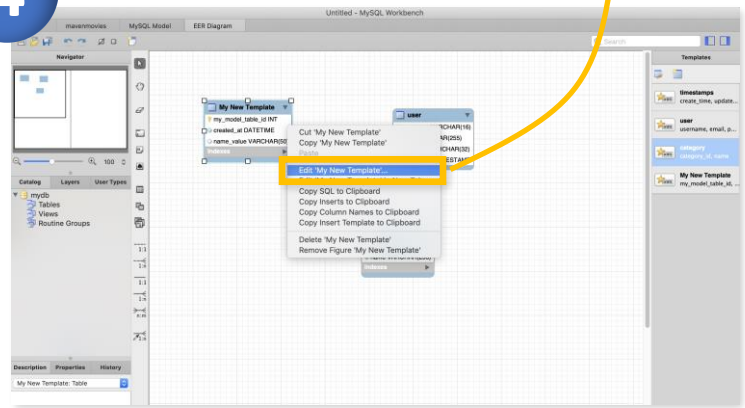
2



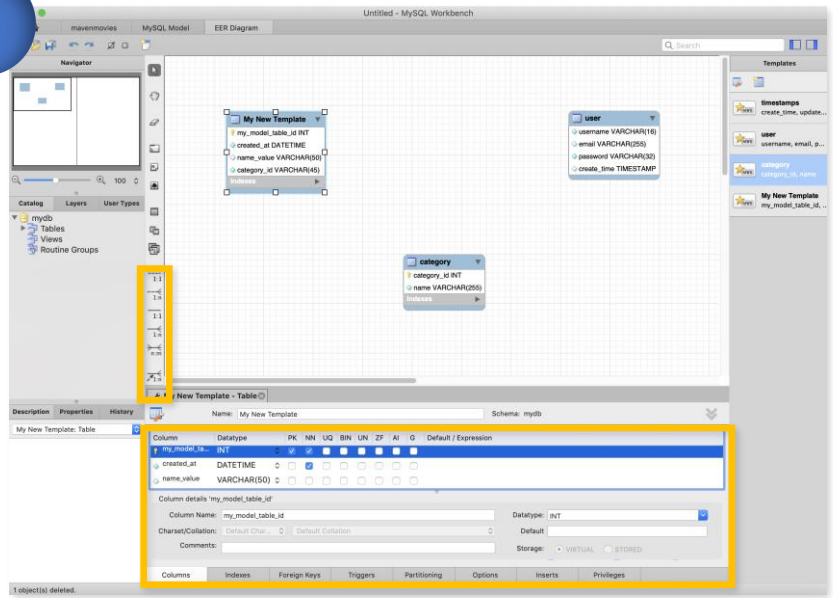
3



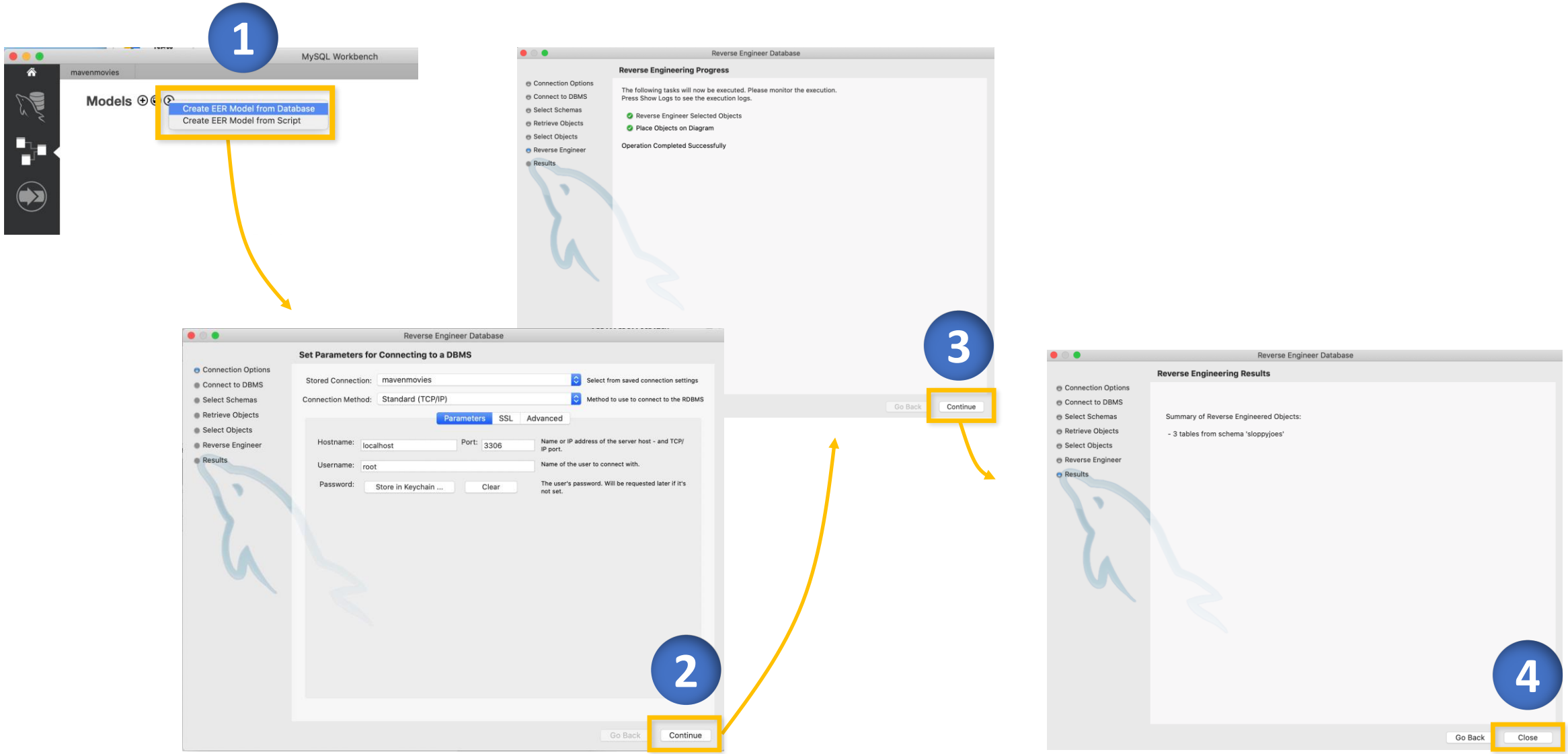
4



5



CREATE AN EER DIAGRAM VIA REVERSE ENGINEERING



NEW MESSAGE
March 15, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Can we make an EER Diagram?**

Good morning,

Another CEO friend of mine just walked me through their database using an EER diagram. It was so easy to understand how everything connected. We should have one!

Can you put together an EER Diagram for us?

Thanks!

-Sally

Reply Forward

Helpful Hints

```
/*
If you've seen a similar EER diagram in
another course, don't just copy it!

There are some nuances to this database,
so create one from scratch.

No shortcuts!
*/
```

TEST YOUR SKILLS: **CREATING EER DIAGRAMS**

NEW MESSAGE
March 15, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Can we make an EER Diagram?**

Good morning,

Another CEO friend of mine just walked me through their database using an EER diagram. It was so easy to understand how everything connected. We should have one!

Can you put together an EER Diagram for us?

Thanks!

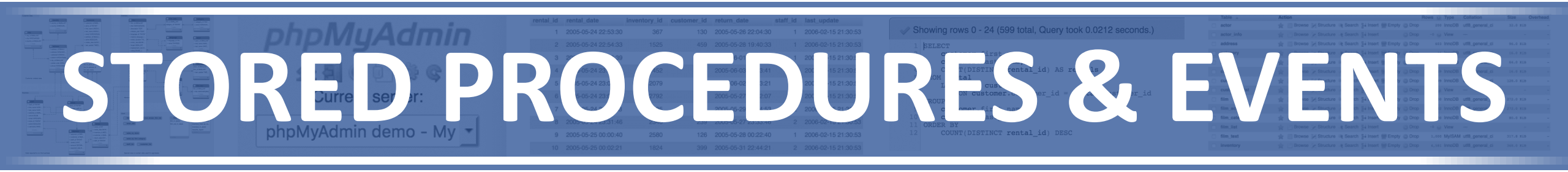
-Sally

Reply Forward

Solution Query

```
-- see video for solution
```

TEST YOUR SKILLS: **CREATING EER DIAGRAMS**



STORED PROCEDURES & EVENTS

STORED PROCEDURES

- MySQL gives us the ability to store and call frequently used queries on the server. These are then referred to as 'Procedures', or commonly 'Stored Procedures'
- Benefits include more efficient query writing and performance, and the ability to share complex procedures more easily between Analysts and database users
- We invoke the stored procedure using a CALL statement

MySQL Workbench in Action

```
USE thriftshop;

-- changing the delimiter
DELIMITER //
-- creating the procedure
CREATE PROCEDURE sp_selectAllInventory()
BEGIN
    SELECT * FROM inventory;
END //
-- changing the delimiter back to the default
DELIMITER ;

-- calling the procedure that we have created
CALL sp_selectAllInventory();

-- if we later want to  DROP the procedure, we can use the this...
-- DROP PROCEDURE IF EXISTS sp_selectAllInventory();
```

STORED PROCEDURES

- Stored Procedures can be used with parameters, which can be IN, OUT, or INOUT
- IN parameters take in a value from the user or application. Values are not returned by the Procedure
- OUT parameters have their values returned by the Procedure
- INOUT is a combination of IN and OUT parameter modes. It can take in a value, which will also output

MySQL Workbench in Action

```
USE mavenmovies;

-- changing the delimiter
DELIMITER //
-- creating the procedure
CREATE PROCEDURE customer_rentals
(IN custid BIGINT)
BEGIN
    SELECT * FROM rental WHERE customer_id = custid;
END //
-- next, changing the delimiter back
DELIMITER ;

-- calling the procedure, with 135 as the input value
CALL customer_rentals(135)
```

STORED PROCEDURES

- OUT parameters have their values returned by the Procedure
- If you want, you can specify a variable to store the output returned in an OUT parameter
- INOUT is a combination of IN and OUT parameter modes. It can take in a value, which will also output

MySQL Workbench in Action

```
-- next, we'll add an OUT...

-- changing the delimiter
DELIMITER //
-- creating the procedure
CREATE PROCEDURE customer_rentals_w_total
(IN custid BIGINT, OUT total_rentals BIGINT) -- this time, we've added an OUT parameter
BEGIN
    SELECT COUNT(rental_id) INTO total_rentals FROM rental WHERE customer_id = custid;

    SELECT * FROM rental WHERE customer_id = custid;
END //
-- next, changing the delimiter back
DELIMITER ;


CALL customer_rentals_w_total (135,@total_rentals);

SELECT @total_rentals;
```

SCHEDULED EVENTS

- MySQL allows us to specify Events, often referred to as Scheduled Events
- Similar to Stored Procedures, Events are created with a user-defined SQL operation
- Unlike Procedures, Events will fire at a certain time, or at regular recurring intervals

MySQL Workbench in Action

```
-- just making a new schema
CREATE SCHEMA schema_for_events;

USE schema_for_events;

-- we'll create timestamp values for this table using events
CREATE TABLE sillytable (
timestamps_via_event DATETIME
);

CREATE EVENT myfirstevent
    ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 MINUTE
    DO
      INSERT INTO sillytable VALUES (NOW());

-- run this quick, wait a minute, try again. Magic!
SELECT * FROM sillytable;

-- now we'll create a recurring event
CREATE EVENT mysecondevent
    ON SCHEDULE
        EVERY 20 SECOND
    DO
      INSERT INTO sillytable VALUES (NOW());

SELECT * FROM sillytable;

DROP EVENT mysecondevent; -- to keep it from running forever
```

NEW MESSAGE
April 1, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Help me pull reports on my own?**

Hey there!

With all this great data, I would love to be able to quickly pull together the total orders and revenue for a given time period. I'm not a SQL guru though.

Is there a way I could specify a **startDate** and **endDate** and see total orders and revenue during that period?

Thanks for the help!
-Sally

Reply Forward

Result Preview

```
-- you'll want her to be able to type something like this...

CALL order_performance('2013-11-01','2013-12-31');
```

```
5    -- the results should look like this...
6
```

100% 1:2

Result Grid Filter Rows: Search Export

total_orders	total_revenue
1908	104893.62

TEST YOUR SKILLS: **STORED PROCEDURES**

NEW MESSAGE
April 1, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Help me pull reports on my own?**

Hey there!

With all this great data, I would love to be able to quickly pull together the total orders and revenue for a given time period. I'm not a SQL guru though.

Is there a way I could specify a **startDate** and **endDate** and see total orders and revenue during that period?

Thanks for the help!
-Sally

Reply Forward

Solution Query

```
DELIMITER //

CREATE PROCEDURE order_performance
(IN startdate DATE, IN enddate DATE)
BEGIN
SELECT
    COUNT(order_id) AS total_orders,
    SUM(price_usd) AS total_revenue
FROM orders
WHERE DATE(created_at) BETWEEN startdate AND enddate;
END //

DELIMITER ;
```

```
CALL order_performance('2013-11-01','2013-12-31');
```

TEST YOUR SKILLS: **STORED PROCEDURES**

WRITING DATA FROM WEBSITES

WRITING DATA FROM WEBSITES

TYPES OF DATA WEBSITES CAPTURE AT SESSION START

TRAFFIC SOURCE DATA

website_session_id	created_at	user_id	is_repeat_session	utm_source	utm_campaign	utm_content	device_type	http_referer
237966	2014-04-28 23:02:55	204524	0	gsearch	nonbrand	g_ad_1	desktop	https://www.gsearch.com
319940	2014-09-11 12:38:39	271374	0	bsearch	nonbrand	b_ad_1	desktop	https://www.bsearch.com
326645	2014-09-22 13:50:39	271374	1	gsearch	brand	g_ad_2	desktop	https://www.gsearch.com
325116	2014-09-19 11:42:44	275579	0	socialbook	desktop_targeted	social_ad_2	desktop	https://www.socialbook.com
349691	2014-10-26 19:24:17	275579	1	gsearch	brand	g_ad_2	desktop	https://www.gsearch.com
357769	2014-11-06 22:10:25	275579	1	NULL	NULL	NULL	mobile	https://www.gsearch.com
367395	2014-11-19 14:56:33	275579	1	NULL	NULL	NULL	mobile	NULL

USER-RELATED DATA

TECHNOLOGY DATA

TYPES OF DATA WEBSITES CAPTURE ALONG THE WAY

website_pageview_id	created_at	website_session_id	pageview_url
500157	2014-03-09 20:50:49	210749	/cart
500158	2014-03-09 20:50:49	210751	/lander-4
500159	2014-03-09 20:51:48	210747	/shipping
500160	2014-03-09 20:53:49	210747	/billing-2
500161	2014-03-09 20:54:21	210752	/lander-3
500162	2014-03-09 20:54:47	210753	/home
500163	2014-03-09 20:54:51	210754	/lander-3
500164	2014-03-09 20:55:50	210753	/products
500165	2014-03-09 20:56:34	210753	/the-original-mr-fuzzy
500166	2014-03-09 20:56:45	210754	/products
500167	2014-03-09 20:58:27	210755	/home
500168	2014-03-09 20:59:08	210747	/thank-you-for-your-order
500169	2014-03-09 21:04:39	210756	/home
500170	2014-03-09 21:06:14	210757	/lander-3
500171	2014-03-09 21:07:14	210757	/products

Pageview and Other Event / Activity Data:

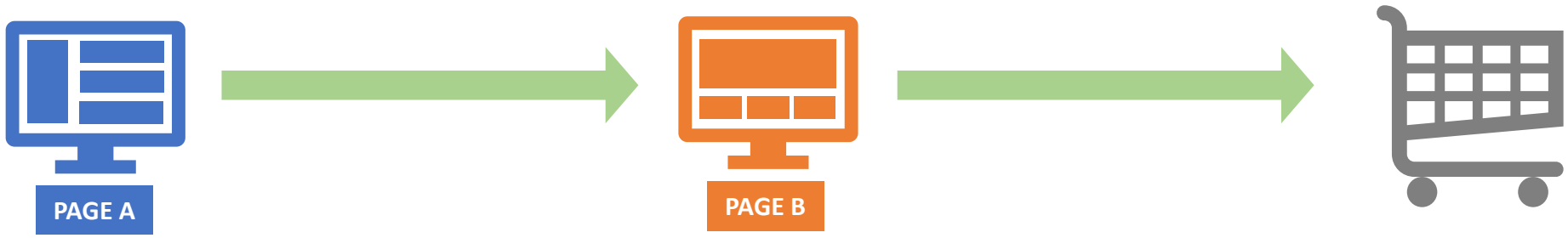
- As activities happen on the website, data records are created to track and measure behavior
- This can happen on a page load, the download of a white paper, etc.

Purchases + Key Business Activities

- Business will always want to capture data related to purchases, subscriptions, lead signups, etc.
- Many times this data is triggered by website activities

order_id	created_at	website_session_id	user_id	primary_product_id	items_purchased	price_usd	cogs_usd
20649	2014-09-23 16:17:08	327437	277442	1	1	49.99	19.49
20650	2014-09-23 16:20:46	327448	277450	2	1	59.99	22.49
20651	2014-09-23 16:41:23	327474	277472	1	2	79.98	28.98
20652	2014-09-23 16:42:05	327472	277471	1	1	49.99	19.49
20653	2014-09-23 16:43:13	327475	277473	3	2	95.98	33.98
20654	2014-09-23 17:01:26	327486	277482	1	1	49.99	19.49
20655	2014-09-23 17:17:51	327499	277492	3	2	75.98	23.98
20656	2014-09-23 17:21:42	327503	277495	1	2	79.98	28.98
20657	2014-09-23 17:48:29	327531	277518	1	1	49.99	19.49
20658	2014-09-23 17:58:38	327536	277522	1	2	95.98	33.98
20659	2014-09-23 18:00:05	327554	277539	1	1	49.99	19.49
20660	2014-09-23 18:56:46	327583	277560	1	2	79.98	28.98
20661	2014-09-23 18:57:52	327582	277559	1	1	49.99	19.49
20662	2014-09-23 18:58:09	327578	231588	1	2	79.98	28.98
20663	2014-09-23 19:12:32	327596	277571	2	1	59.99	22.49

RECAP OF CAPTURING WEBSITE DATA



- User-related data (cookies)
- Traffic source data
- Technology data
- Pageview data

- Pageviews
- Other important events

- Purchases
- Subscriptions
- Lead signups
- Other business activity

NEW MESSAGE

April 6, 2014

From: **Molly Monterey** *(Website Manager)*

Subject: **Any recommended enhancements?**

Good morning!

Given your expertise and your knowledge of the website data we are already tracking in our SQL database, is there any additional data you recommend we add?

Thanks!

-Molly

Reply Forward

Helpful Hints

```
-- think about the website data topics we covered

-- is anything missing that you would want?
```

TEST YOUR SKILLS: **WRITING WEBSITE DATA**

NEW MESSAGE

April 6, 2014

From: **Molly Monterey** *(Website Manager)*

Subject: **Any recommended enhancements?**

Good morning!

Given your expertise and your knowledge of the website data we are already tracking in our SQL database, is there any additional data you recommend we add?

Thanks!

-Molly

Reply Forward

Solution

```
-- see video for solution
```

TEST YOUR SKILLS: **WRITING WEBSITE DATA**

NEW MESSAGE

April 15, 2014

From: **Molly Monterey** *(Website Manager)*

Subject: **Loading Website Pageview Data**

Hey!

I was able to get this February pageview data out of our web analytics tool. Would you be able to help me load it into the database so we can tie it to all of your other great data?

Thanks!

-Molly

16.website_pageviews_2014_Feb

Reply Forward

Helpful Hints

```
-- take a look at the pageview data structure

-- create a website_pageviews table

-- then upload the dataset
```

TEST YOUR SKILLS: **IMPORTING PAGEVIEW DATA**

NEW MESSAGE
April 15, 2014

From: **Molly Monterey** *(Website Manager)*

Subject: **Loading Website Pageview Data**

Hey!

I was able to get this February pageview data out of our web analytics tool. Would you be able to help me load it into the database so we can tie it to all of your other great data?

Thanks!

-Molly

16.website_pageviews_2014_Feb

Reply Forward

Solution Query

```
CREATE TABLE website_pageviews (
    website_pageview_id BIGINT,
    create_at DATETIME,
    website_session_id BIGINT,
    pageview_url VARCHAR(50),
    PRIMARY KEY (website_pageview_id)
)
;
```

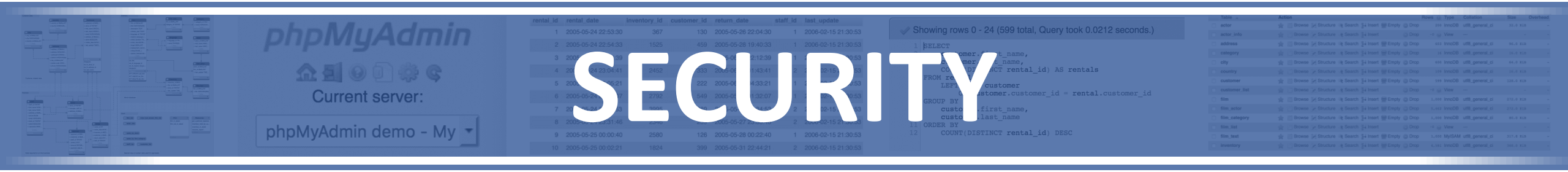
```
22    -- you should see 38,391 records if you've done this right
23 •  SELECT COUNT(*) AS total_records FROM website_pageviews;
```

100% 1:19

Result Grid Filter Rows: Search Export:

total_records
38391

TEST YOUR SKILLS: **IMPORTING PAGEVIEW DATA**



SECURITY

COMMON SECURITY THREATS & ATTACKS

1	**Weak Authentication**	*Attackers can take advantage of limited or weak authentication methods in place to break through your perimeter*
2	**Denial of Service**	*DoS attackers will attempt to flood or crash your system, rendering it useless for legitimate users*
3	**Privilege Escalation**	*Attackers will use this strategy to gain access to higher levels of your systems and information you don't want them to see*
4	**SQL Injection**	*Various endpoints of your system can create vulnerabilities for hackers who understand how your back-end works*
5	**Buffer Overflow**	*Similar to SQL injection, attackers can take advantage of holes in your application and make it do things it shouldn't*
6	**Ransomware**	*Malware attacks will render your data unusable unless you agree to pay a ransom*

WEAK AUTHENTICATION

Authentication refers to the processes you employ for users to prove to your system that they are who they say they are.

Attackers can gain access to your system by tricking your system into thinking they are one of your trusted users.

They may do this by taking advantage of:

- Weak passwords
- Passwords that do not change frequently
- Lack of a two-factor or multi-factor authentication mechanism

DENIAL OF SERVICE ATTACKS

With **Denial of Service (DoS) attacks**, the hackers will attempt to overwhelm your system with requests, which will render it useless for your legitimate users.

There are services you can employ that will filter out suspicious activity, and when a DoS attack is detected, there are a number of mechanisms for blocking the attack.

In one special case -- **a Distributed Denial of Service (DDoS)** -- attackers will coordinate the attack from multiple machines simultaneously, often distributed globally. This makes it harder to determine the attacking source and shut them off quickly.

THIS IS IMPORTANT!

A DoS or DDoS can sometimes be used to distract from and cover up additional attacks on your systems. In some cases, the intent is to extract data elsewhere. When you encounter a DoS, be on high alert across all systems.

PRIVILEGE ESCALATION ATTACKS

With **privilege escalation attacks**, hackers will attempt to gain access to increasingly secure levels of information by authenticating as a lower level user first, then climbing the ladder.

Once they have gained access to some of your systems, it can become easier for them to identify additional security holes and gain higher levels of access.

Some common methods for implementing these attacks:

- **Cookie-based** authentication
- **Form-based** authentication

With cookie-based authentication, hackers may spoof this to impersonate another user

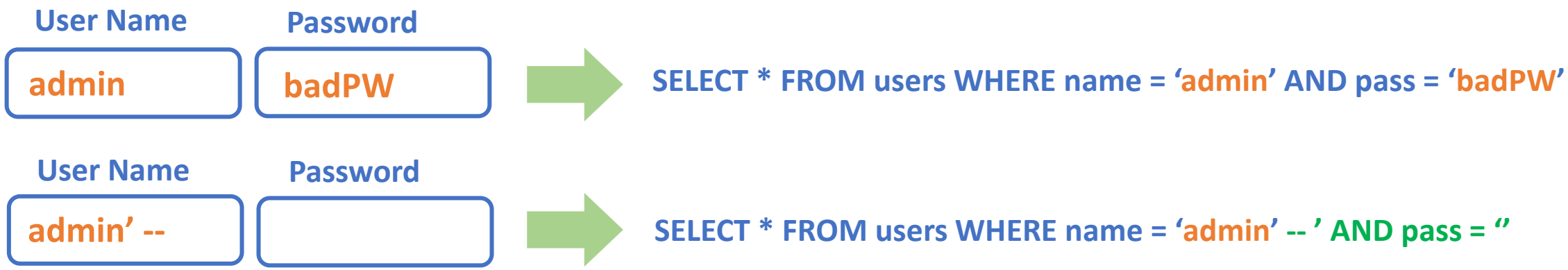
user_id = **666**

SQL INJECTION ATTACKS

With **SQL injection attacks**, hackers will attempt to make your application do something it's not supposed to do by editing SQL statements utilized on the back-end.

Common ways attackers might use SQL injection maliciously:

- **Gain information about your database** *(structure, version, etc.)*
- **Return results from your database that they shouldn't see**



BUFFER OVERFLOW VULNERABILITY

A **buffer** is a sequential section of memory which will contain anything from a string to an array of numbers.

A **buffer overflow** is when more data is put into the buffer than it can handle. The causes the data to overflow and get written elsewhere.

Attackers can deliberately overflow your application to crash it or to force it to do things you might not want it to do.

RANSOMWARE

Ransomware has become more prevalent in recent years. It represents an attractive opportunity for hackers and an expensive problem for businesses.

When a business is attacked by ransomware, systems will be rendered unusable and the attacker will demand payment *(typically in the form of cryptocurrency)*

Malware most often breaches systems by getting employees to download a file from their email, but social media attacks have been reported as well.

THIS IS IMPORTANT!

Preparing for and defending against ransomware attacks needs to be a team effort. The Database Administrator and CTO should prepare by having adequate and safe backups, and employees should be trained to avoid phishing.

SECURITY BEST PRACTICES

1 Practice Safe Data Storage	*Steer clear of storing personal information if you can. Anything sensitive that you do need to store should be encrypted.*
2 Limit Access	*Practice "minimum viable access". Don't grant more permissions than are needed for the job.*
3 Take Authentication Seriously	*Require strong passwords, require changing frequently, and enable multi-factor authentication.*
4 Dedicate Resources to Security	*Whether it is a team, a person, or some percentage of a person's time, someone in your organization should be focused on security.*
5 Have a Data Security Policy	*Come up with a formal security policy. Write it down. Train your employees on best practices.*
6 Backup, Log, Monitor & Audit	*Backup your data. Log changes & monitor access. Perform periodic audits & invite third parties to try and hack you.*

PRACTICE SAFE DATA STORAGE

Make every attempt to store as little **personally identifiable information** (PII) as possible

- *If you don't absolutely need it for your application, consider NOT storing it*

For sensitive data that you need to store *(credit cards, social security #, email address, phone number, passwords, etc.)*, make sure to **encrypt** it whenever possible

Make a list of the applications and people in your organization that have access to sensitive data, and where you are vulnerable. Is the list longer than it should be?

LIMIT ACCESS TO SYSTEMS AND DATA

- Make sure to practice "**minimum viable access**" in granting your permission levels
  - *Very few employees will need full access to systems and data. Grant only what they need.*

- When thinking about permission levels, consider whether employees need edit rights or if **read-only** access is sufficient for their job function

- Make sure that you have a plan for quickly **removing access** to systems when employees leave, or when you need to limit access on a case-by-case basis

TAKE AUTHENTICATION SERIOUSLY

- Require **strong passwords** for employees, and don't allow them create weak passwords
  - *Minimum 8 characters, containing a number, a special character, and both upper- and lower-case letters*

- Require employees to **change their password** once every 60 to 90 days
  - *They may hate it, but it will mitigate potential attacks*

- Leverage **two-factor** or **multi-factor** authentication methods, especially for employees with access to sensitive information

DEDICATE RESOURCES TO SECURITY

- Whether it is a whole team, a single person, or part of someone's time *(in very small companies)*, **someone in your organization should be responsible for security planning**
- Make security a part of someone's **job description** and **performance review**
- Give the responsible employee the **authority** to work with the rest of your organization to ensure that things are set up properly

HAVE A WRITTEN SECURITY POLICY

- Writing down your security policy forces you to create a **cohesive plan** and determine **specific action items** that you will be held accountable for completing

- **Share** your written security policy with all employees

- Make sure you are continually **reviewing and revising** your security policy as your business and your risks evolve over time

BACKUP, LOG & MONITOR YOUR DATA. PERFORM AUDITS

- **Back up** your data, especially the most important information; in the event of a breach, system malfunction, or malware, you will be happy to have another copy

- **Create logs** on admin systems and database activity so you can monitor activity

- **Conduct periodic reviews** and invite friendly third parties to try hacking your system to expose potential vulnerabilities

NEW MESSAGE

April 20, 2014

From: **Sally Blue** *(CEO)*

Subject: **We need a security plan!**

Hey there!

I know you've been learning a lot about security, and the board is giving me a lot of pressure on that topic.

Can you put together a comprehensive security plan for us? Include anything you think makes sense for today, and also as we scale up.

-Sally

Reply Forward

Result Preview

```
-- recommend you write out a formal plan

-- it can be a bulleted list, nothing fancy

-- writing it down will force you to think about it
```

TEST YOUR SKILLS: **DATABASE SECURITY**

NEW MESSAGE
April 20, 2014

From: **Sally Blue** *(CEO)*

Subject: **We need a security plan!**

Hey there!

I know you've been learning a lot about security, and the board is giving me a lot of pressure on that topic.

Can you put together a comprehensive security plan for us? Include anything you think makes sense for today, and also as we scale up.

-Sally

Reply Forward

Solution Query

```
-- see video for solution
```

TEST YOUR SKILLS: **DATABASE SECURITY**

INTRODUCING THE **FINAL COURSE PROJECT**

THE SITUATION

There have been some exciting developments for Maven Bear Builders. The company is going to start offering chat support on the website, and needs your help planning. The company has also been approached by potential acquirers, and you'll be asked to help with due diligence.

THE OBJECTIVE

Use SQL to:

- Create a plan for handling chat support, including the database infrastructure, EER diagrams explaining your plan, and reports to help management understand performance
- Provide support for questions relating to the potential acquisition, to help your CEO keep them interested and hopefully close the deal

INTRODUCING THE **FINAL PROJECT**

NEW MESSAGE
May 1, 2014

From: **Sally Bleu** *(CEO)*

Subject: **Exciting Developments!**

Good morning!

Two big updates for you…

1. We're adding chat support to the website, and need your help planning for the data structure.

2. We were approached by a larger company who is interested in acquiring us. Will need some help here too.

Let's do this!
-Sally

Reply Forward

YOUR OBJECTIVES:

- Update the database with the most recent data
- Create a plan for the company's service expansion to include chat support
- Help Sally with some support for asks related to the potential acquisition

FINAL COURSE PROJECT QUESTIONS

1

~ 0:37

Import the latest order_items and **order_item_refunds** data below into the database, and verify the order summary trigger you created previously still works *(if not, recreate it)*

17.order_items_2014_Mar
18.order_items_2014_Apr
19.order_item_refunds_2014_Mar
20.order_item_refunds_2014_Apr

2

~ 5:30

Import the website_sessions and **website_pageviews** data for March and April, provided below:

21.website_sessions_2014_Mar
22.website_sessions_2014_Apr
23.website_pageviews_2014_Mar
24.website_pageviews_2014_Apr

3

~ 9:14

The company is adding chat support to the website. You'll need to **design a database plan** to track which customers and sessions utilize chat, and which chat representatives serve each customer

4

~ 14:00

Based on your tracking plan for chat support, **create an EER diagram** that incorporates your new tables into the existing database schema *(including table relationships)*

FINAL COURSE PROJECT QUESTIONS

5

~ 20:40

Create the tables from your chat support tracking plan in the database, and include relationships to existing tables where applicable

6

~ 26:37

Using the new tables, **create a stored procedure** to allow the CEO to pull a count of chats handled by chat representative for a given time period, with a simple CALL statement which includes two dates

7

~ 31:18

Create two Views for the potential acquiring company; one detailing monthly order volume and revenue, the other showing monthly website traffic. Then **create a new User**, with access restricted to these Views

8

~ 37:00

The potential acquirer is commissioning a third-party security study, and your CEO wants to get in front of it. Provide her with a **list of your top data security threats** and recommendations for mitigating risk