

Information Technologies

Anselm Spoerri PhD (MIT)

SC&I @ Rutgers University

aspoerri@rutgers.edu

anselm.spoerri@gmail.com

Lecture 10 - Overview

Relational Databases

SQL = Structured Query Language

MySQL & MySQL Workbench

- Create MySQL Database on Server
- Grant Access to MySQL Workbench
- Create & Test “Connection” in MySQL Workbench
- Create Table and Perform SQL Queries in MySQL Workbench

Lectures – Week 10 Content

<http://comminfo.rutgers.edu/~aspoerri/Teaching/InfoTech/Lectures.html#week10>

Websites that use **Relational Databases**



amazon.com.



You Tube
Broadcast Yourself



eBay

Is the WWW a Relational Database?

Fairly sophisticated search available

- Crawler indexes pages on the Web
- Keyword-based search for pages

But, currently

- Data is mostly *unstructured* and *untyped*
- Can't modify the data
- Can't get summaries, complex combinations of data
- Few guarantees provided for freshness of data, consistency across data items, ...

The picture is changing

- New standards, e.g., XML, Semantic Web, etc., can provide richer models of data

How is Relational Database different from Spreadsheet?

Spreadsheet	Relational Database
<ul style="list-style-type: none">• Your data is of a manageable data size• There is no need for relationships between data• You are primarily creating calculations and statistics	<ul style="list-style-type: none">• You are working with large amounts of data• You need to create relationships between your data• You rely on external databases to analyze data

Relational Database – Basic Concepts

Relational Database

http://en.wikipedia.org/wiki/Relational_database

- Collection of data, **organized** to support access
- **Models** some aspects of reality

Components of Relational Database

Field = an “atomic” unit of data city

Record = a collection of related fields address

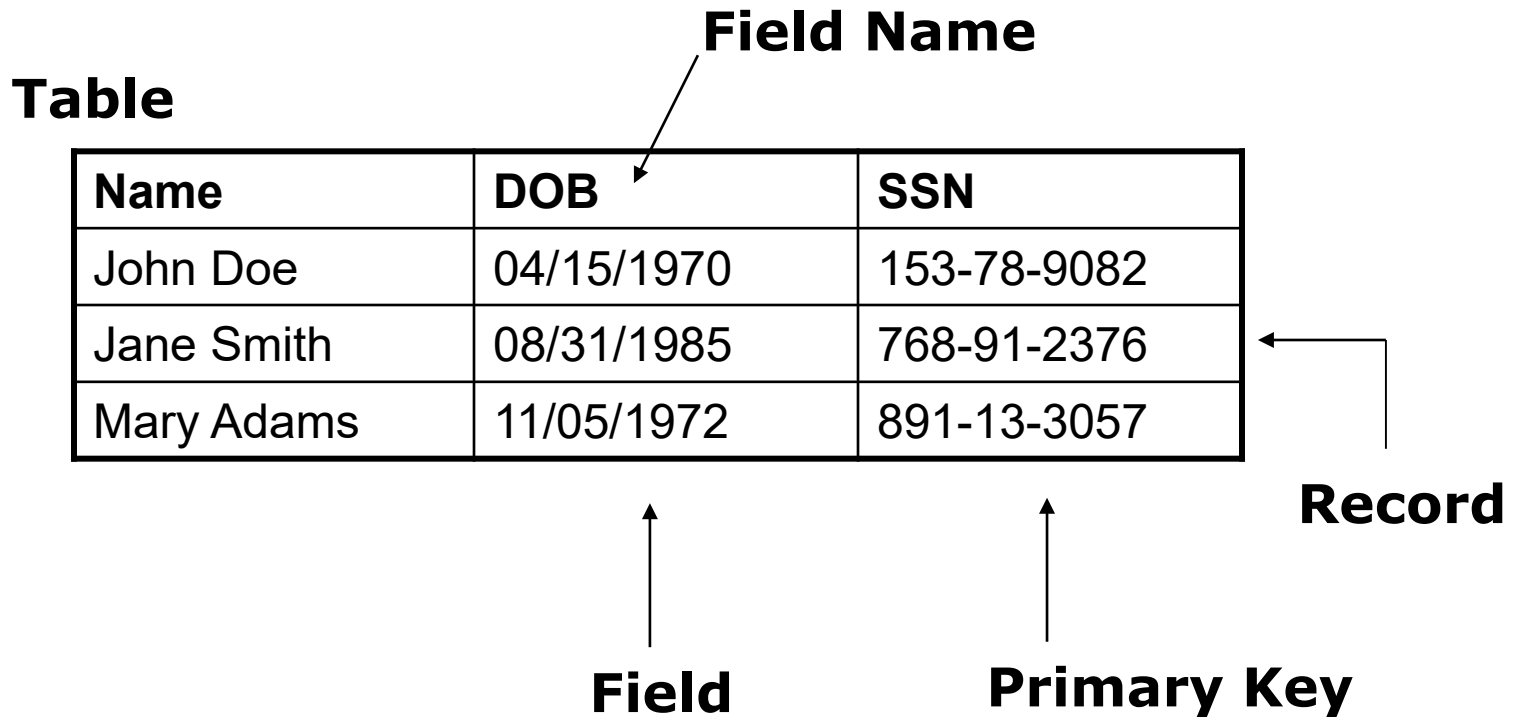
Table = a collection of related records addresses

- Each **record** is one **row** in the table
- Each **field** is one **column** in the table

Database = a collection of tables student data

Primary Key = **field** that *uniquely identifies* a record

Simple Example



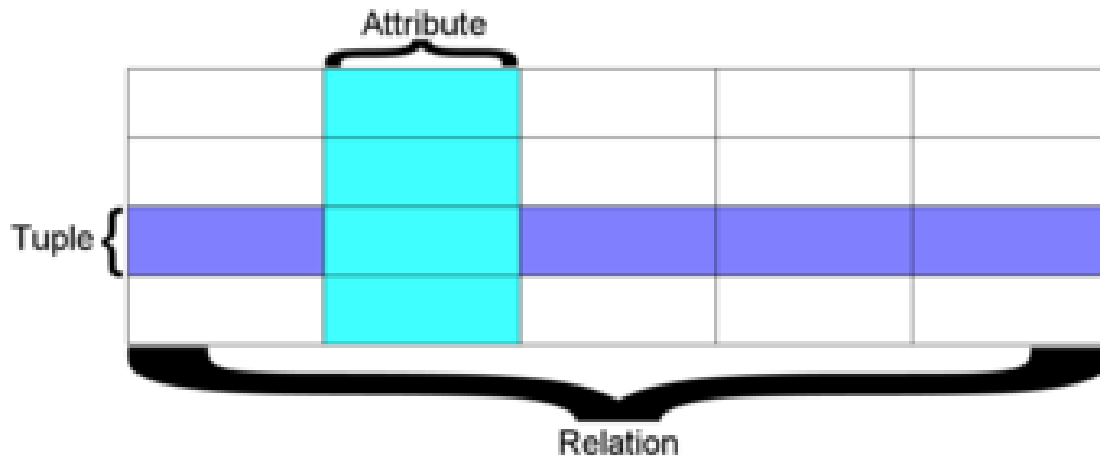
Why "Relational"?

Databases Model some Aspects of Reality

Relational Database groups data using Common Attributes found in data set

- The resulting "clumps" of organized data are much easier for people to understand
- The grouping uses the **relational model**

Relational Database – Terminology



A **relation** is defined as a set of tuples that have the same attributes.

A relation is usually described as a **table**, which is organized into **rows** and **columns**.

All the data referenced by an **attribute** are in the **same domain** and conform to the same constraints.

Registrar Example

Want to create Database to Collect Enrolled Student Data

What do we need to know (i.e., model)?

- Something about **students**
(e.g., first name, last name, email, department)
- Something about **courses**
(e.g., course ID, description, enrolled students, grades)
- Which students are in which courses

First Try

Put everything in a big table...

1	Arrows	John	EE	EE	lis550	Information Technology	90	jarrows@wam
1	Arrows	John	EE	Elec Engin	ee750	Communication	95	ja_2002@yahoo
2	Peters	Kathy	HIST	HIST	lis550	Informatino Technology	95	kpeters2@wam
2	Peters	Kathy	HIST	history	hist405	American History	80	kpeters2@wma
3	Smith	Chris	HIST	history	hist405	American History	90	smith2002@glue
4	Smith	John	CLIS	Info Sci	lis550	Information Technology	98	js03@wam

Discussion: Why is this a bad idea?

Need to **Normalize** the Data

➔ Attribute can only have one value

Goals of “Normalization”

http://en.wikipedia.org/wiki/Database_normalization

Remove Duplicates / Save Space

- Save each fact only once

More Rapid Updates

- Every fact only needs to be **updated once**

More Rapid Search

- **Finding it once** is good enough

Avoid Inconsistency

- Changing data once changes it everywhere

Second Try ... With Normalization in Mind

Student Table

Student ID	Last Name	First Name	Department ID	email
1	Arrows	John	EE	jarrows@wam
2	Peters	Kathy	HIST	kpeters2@wam
3	Smith	Chris	HIST	smith2002@glue
4	Smith	John	LIS	js03@wam

Department Table

Department ID	Department
EE	Electrical Engineering
HIST	History
LIS	Library and Info Science

Course Table

Course ID	Course Name
lis550	Information Technology
ee750	Communication
hist405	American History

Enrollment Table

Enroll ID	Student ID	Course ID
1	1	lis550
2	1	ee750
3	2	lis550
4	2	hist405
5	3	hist405
6	4	lis550

Approaches to Normalization and Keys

Simple Data Domains

- Start with “binary relationships” (pairs of related fields)
- Group Attributes together wherever possible
- Add keys where necessary

Complicated Data Domain

- Entity relationship modeling

“Primary Key” uniquely identifies a record

- Example: student ID in student table

“Foreign Key” is primary key in other table

- Does not have to be unique in the table where it is “foreign key”

Second Try ... With Normalization in Mind

Student Table

Foreign key

Student ID	Last Name	First Name	Department ID	email
1	Arrows	John	EE	jarrows@wam
2	Peters	Kathy	HIST	kpeters2@wam
3	Smith	Chris	HIST	smith2002@glue
4	Smith	John	LIS	js03@wam

Primary Key

Department Table

Department ID	Department
EE	Electrical Engineering
HIST	History
LIS	Library and Info Science

Primary Key

Course Table

Course ID	Course Name
lis550	Information Technology
ee750	Communication
hist405	American History

Primary Key

Enrollment Table

Enroll ID	Student ID	Course ID
1	1	lis550
2	1	ee750
3	2	lis550
4	2	hist405
5	3	hist405
6	4	lis550

Primary Key

Relational Database Operations

Joining Tables: JOIN

Choosing Columns: SELECT

- Based on their label

Choosing Rows: WHERE

- Based on their contents

These can be specified together

```
SELECT Student ID, Dept WHERE Dept = "History"
```


Relational Database Operation – **Join Tables**

Student ID	Last Name	First Name	Department ID	email
1	Arrows	John	EE	jarrows@wam
2	Peters	Kathy	HIST	kpeters2@wam
3	Smith	Chris	HIST	smith2002@glue
4	Smith	John	LIS	js03@wam

Department ID	Department
EE	Electrical Engineering
HIST	History
LIS	Library and Info Science

Student ID	Last Name	First Name	Dept ID	Department	email
1	Arrows	John	EE	Electrical Engineering	jarrows@wam
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue
4	Smith	John	LIS	Library and Info Science	js03@wam

Relational Database Operation – **SELECT**

Student ID	Last Name	First Name	Dept ID	Department	email
1	Arrows	John	EE	Electrical Engineering	jarrows@wam
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue
4	Smith	John	LIS	Library and Info Science	js03@wam



SELECT Student ID, Department

Student ID	Department
1	Electrical Engineering
2	History
3	History
4	Library and Info Science

Relational Database Operation – **WHERE**

Student ID	Last Name	First Name	Dept ID	Department	email
1	Arrows	John	EE	Electrical Engineering	jarrows@wam
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue
4	Smith	John	LIS	Library and Info Scienc	js03@wam

WHERE Department ID = "HIST"

Student ID	Last Name	First Name	Department ID	Department	email
2	Peters	Kathy	HIST	History	kpeters2@wam
3	Smith	Chris	HIST	History	smith2002@glue

Database Integrity and Integrity Constraints

Example: Registrar Database

Database must be Internally Consistent

- All enrolled students must have entry in student table
- All courses must have a name
- ...

These Conditions must be true at any time

- Specified when the database is **designed**
- Checked when the database is **modified**

Relational Database Management System

- **Ensures Integrity Constraints** are always kept
- Database contents need to reflect the real world
- Helps **avoid data entry errors**

Database Design Considerations

Field Size Property

- Set Field Size ...
- Always anticipate current field size **may need to be larger**

Validation Rules

- Used to **avoid data entry errors** by restricting what can be entered
- **Validation text** needs to be used to **provide explanation** of the type of **allowed data** in a field

Store Data in its Smallest Part for Greater Flexibility

- Instead of one field for an address, use many
- Instead of one field for a name, two or three

Use Multiple Tables to Reduce Redundancy

- The process is also referred to as normalization

SQL – Querying Relational Database

SQL = Structured Query Language

<http://en.wikipedia.org/wiki/SQL>

Database computer language designed for
Managing & Accessing Data in Relational Database

Queries

- Ask question about data
- Receive answer back by **returning subset of table data.**
- Use JOIN, SELECT, WHERE
- **Need semicolon ;** at the end of query
- SQL commands and keywords are **case-insensitive**

rutgers-sci.domains – Create MySQL Database

Log into **rutgers-sci.domains** using your NetID and Password

In Dashboard: **Databases** – click on **MySQL Database Wizard**

Welcome to SC&I Course Apps Manage Your Account

RUTGERS
School of Communication
and Information

Home Request Form

EMAIL

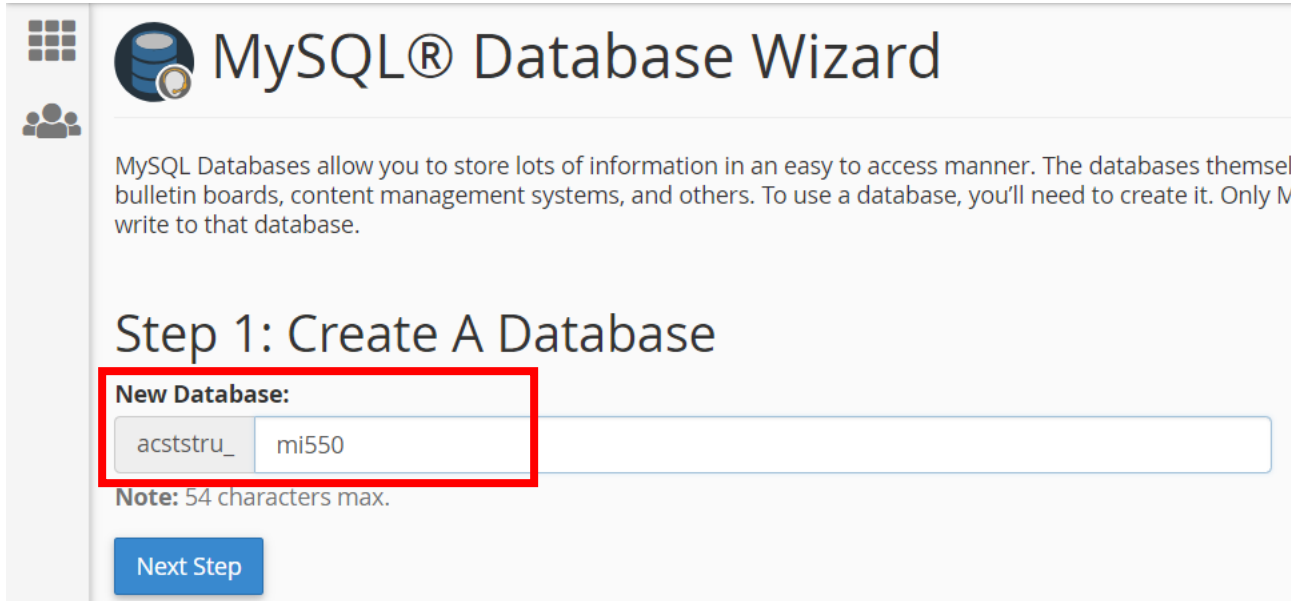
Email Routing Spam Filters Encryption

DATABASES

phpMyAdmin MySQL® Databases **MySQL® Database Wizard** Remote MySQL®

rutgers-sci.domains – Create MySQL Database

Step 1: **New Database** enter "**mi550**"



MySQL® Database Wizard

MySQL Databases allow you to store lots of information in an easy to access manner. The databases themselves bulletin boards, content management systems, and others. To use a database, you'll need to create it. Only M write to that database.

Step 1: Create A Database

New Database:

acststru_ mi550

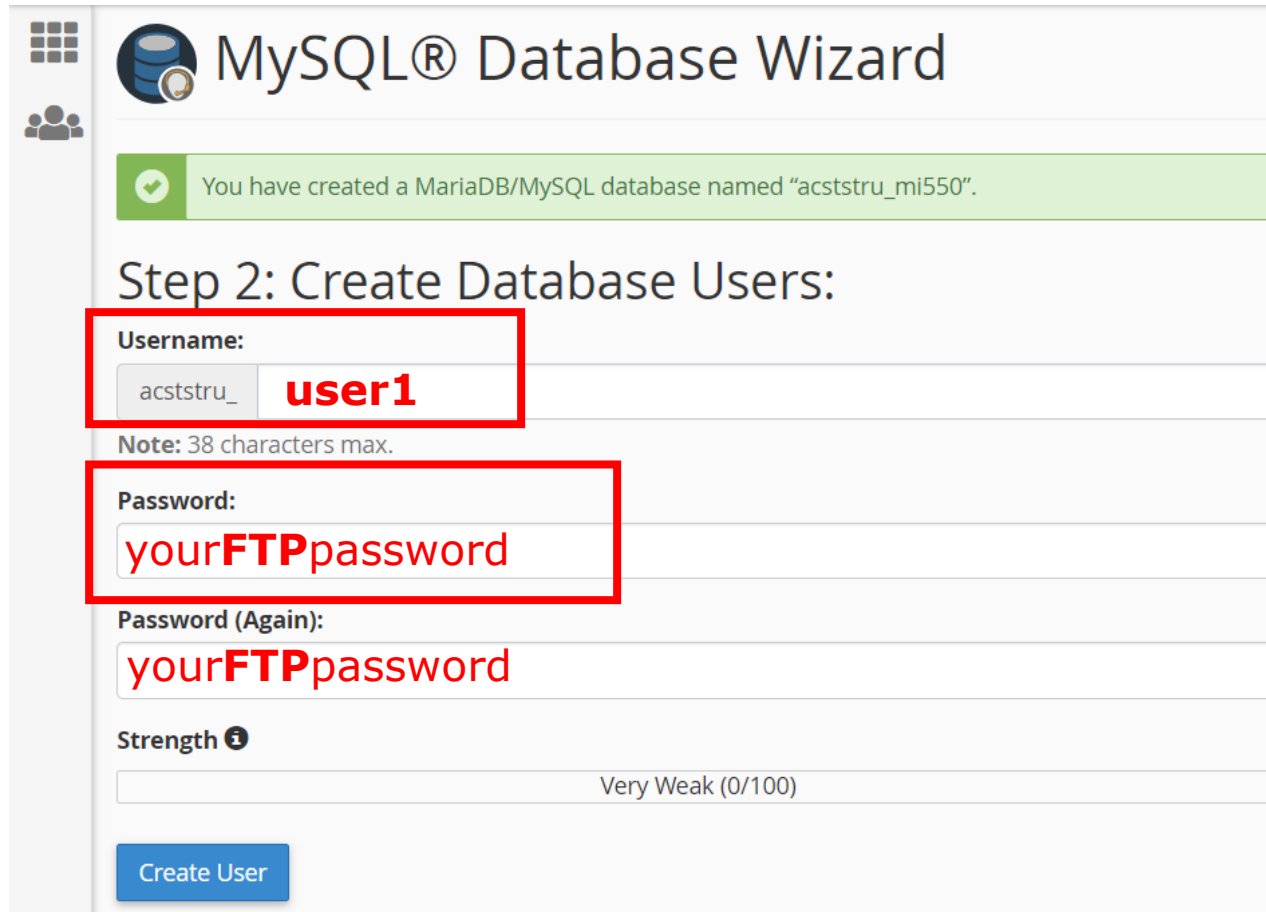
Note: 54 characters max.

Next Step

→ Database name = **yourFTPusername_mi550**

rutgers-sci.domains – Create MySQL Database

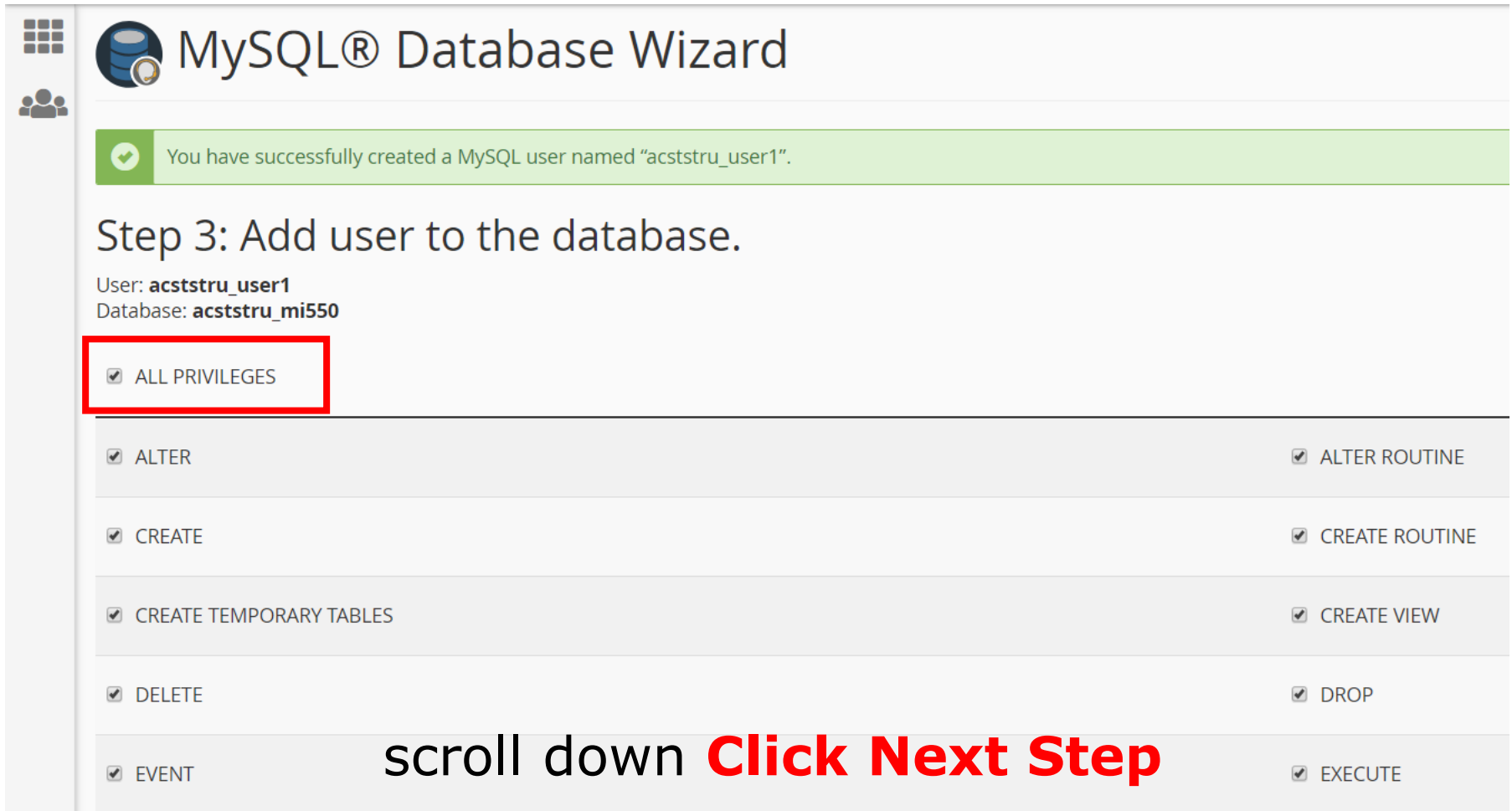
Step 2: **Create** enter "**user**" and Password = your**FTP**password




The screenshot shows the MySQL Database Wizard interface. At the top, it says "MySQL® Database Wizard". Below that, a green notification bar states: "You have created a MariaDB/MySQL database named 'acststru_mi550'." The main heading is "Step 2: Create Database Users:". There are three input fields: "Username:" with the value "acststru_user1", "Password:" with the value "yourFTPpassword", and "Password (Again):" with the value "yourFTPpassword". A "Note: 38 characters max." is displayed below the username field. A "Strength" indicator shows "Very Weak (0/100)". A blue "Create User" button is located at the bottom left.

rutgers-sci.domains – Create MySQL Database

Step 3: **Select ALL PRIVILEGES** checkbox



 MySQL® Database Wizard

 You have successfully created a MySQL user named "acststru_user1".

Step 3: Add user to the database.

User: **acststru_user1**
Database: **acststru_mi550**

ALL PRIVILEGES

<input checked="" type="checkbox"/> ALTER	<input checked="" type="checkbox"/> ALTER ROUTINE
<input checked="" type="checkbox"/> CREATE	<input checked="" type="checkbox"/> CREATE ROUTINE
<input checked="" type="checkbox"/> CREATE TEMPORARY TABLES	<input checked="" type="checkbox"/> CREATE VIEW
<input checked="" type="checkbox"/> DELETE	<input checked="" type="checkbox"/> DROP
<input checked="" type="checkbox"/> EVENT	<input checked="" type="checkbox"/> EXECUTE

scroll down **Click Next Step**

MySQL Username & Password

**Name of Your MySQL database =
yourFTPusername_mi550**

Example: **acststru_mi550**

MySQL username = yourFTPusername

or

yourFTPusername_user1 (when created)

MySQL password = yourFTPpassword

Please **contact help@comminfo.rutgers.edu**
if you have any **technical problems** accessing your server
account or mysql database.

MySQL Workbench – Installation

MySQL Workbench <http://www.mysql.com/products/workbench/>

- **Documentation:** <https://dev.mysql.com/doc/workbench/en/>
- **Download:** Windows | Mac <https://dev.mysql.com/downloads/workbench/>

Generally Available (GA) Releases

MySQL Workbench 8.0.13

Select Operating System:
Microsoft Windows

Looking for previous GA versions?

Recommended Download:

MySQL Installer for Windows

All MySQL Products. For All Windows Platforms. In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI [Go to Download Page >](#)

Other Downloads:

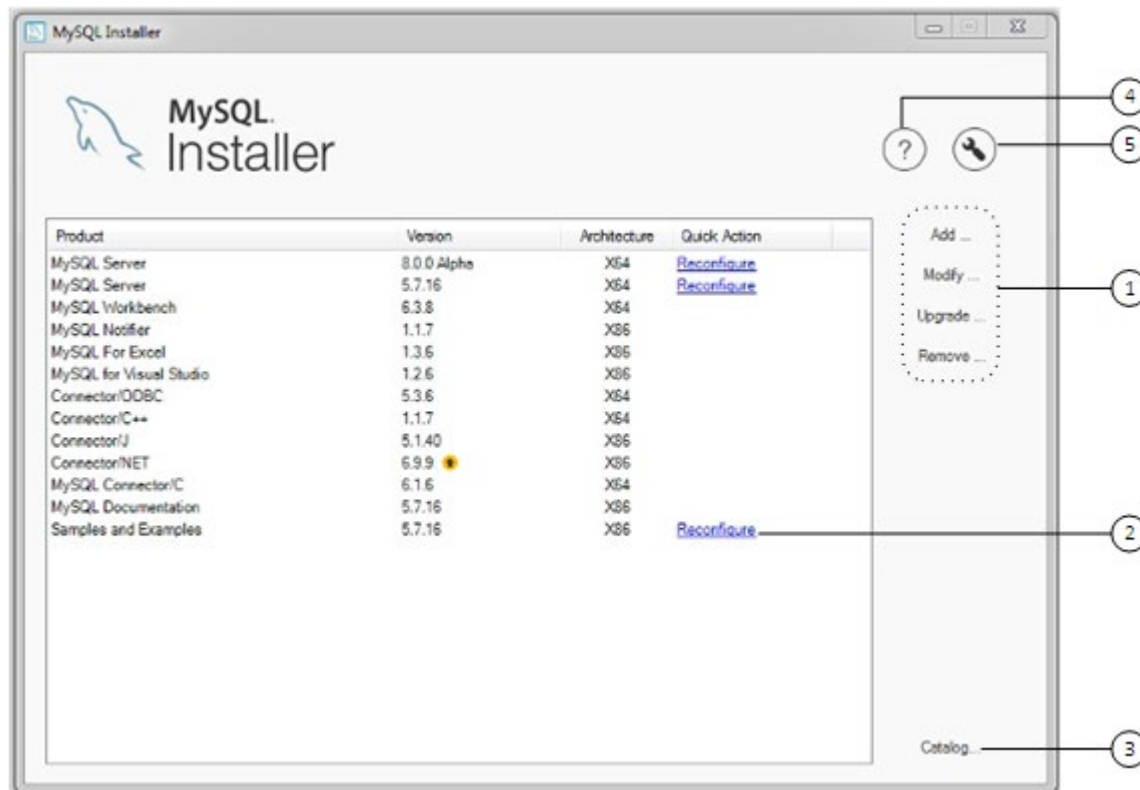
Windows (x86, 64-bit), MSI Installer (mysql-workbench-community-8.0.13-winx64.msi)	8.0.13	33.6M	Download
MD5: 298f374a2a032bf5148fc4909a2dcab1 Signature			

MySQL Installer – Client or Custom

Client only: Only install most recent MySQL client applications, such as **MySQL Workbench**, and **MySQL connectors**.

Custom You can install MySQL Workbench, instead of installing all client applications for Windows.

<https://dev.mysql.com/doc/refman/8.0/en/mysql-installer.html>



Remote MySQL – Grant Access to MySQL Workbench



Dashboard – Databases: click on **Remote MySQL**

- 1 **Host** text box, enter **"%"** (stands for all IP addresses allowed)
- 2 **Click Add Host**

A screenshot of the "Remote MySQL®" "Add Access Host" form. The form has a light gray background. At the top left, there is a grid icon and a user icon. The main heading is "Remote MySQL®". Below it, there is a paragraph of text: "Add a specific domain name to allow visitors to connect to your MySQL databases. Applications like bulletin information, read the [documentation](#)." The form title is "Add Access Host". There are two input fields: "Host (% wildcard is allowed)" and "Comment (optional)". The "Host" field contains the character "%". A red rectangular box highlights the "Host" label and the input field. A red circle highlights the "Add Host" button. Another red circle highlights a black circular button to the right of the "Host" input field.

MySQL Workbench – Create New Connection

New Connection

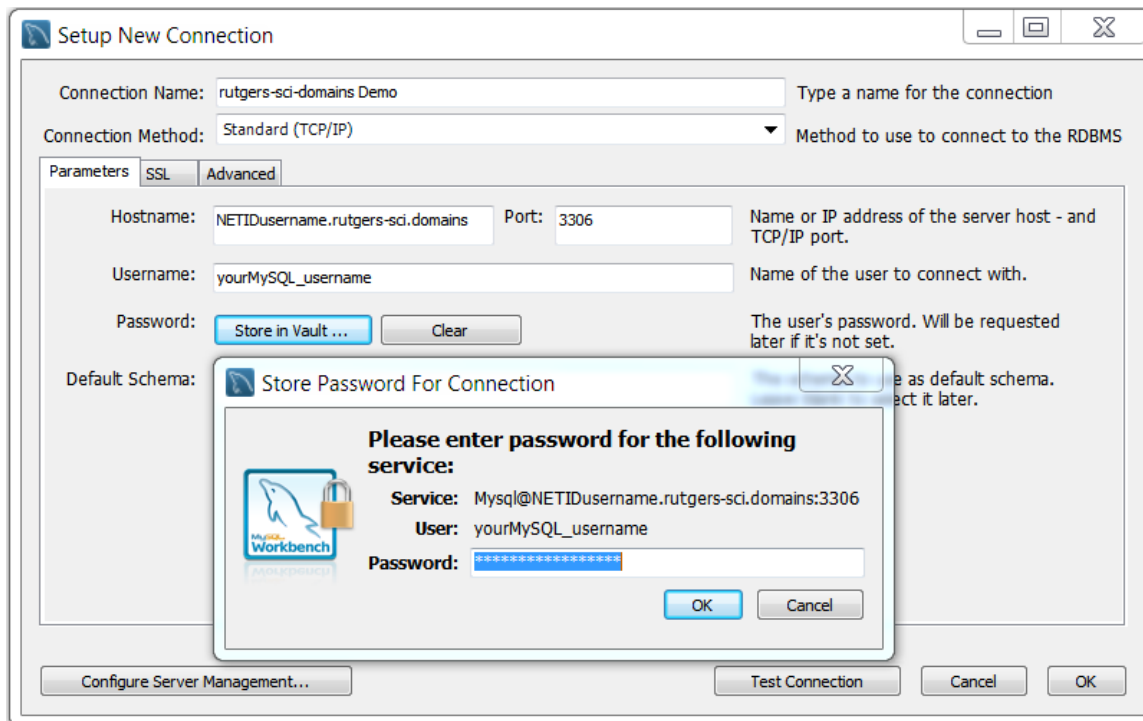
Connection Method: **Standard TCP / IP**

Hostname: your**NetID**username.rutgers-sci.domains

Port: **3306**

Username: your**MySQL**username (= yourFTPusername or yourFTPusername_user1)

Password: your**MySQL**password (= yourFTPpassword)



SQL Demo – Create Table

DatabaseToUse = **yourFTPusername_mi550** created by you

USE DatabaseToUse; // tells SQL which database to use

Note: if DatabaseToUse **contains – hyphens** then will **trigger SQL error**

You can use the grave mark ` to escape names that contain reserved lexical symbols

MySQL Workbench:

Double-click icon of database to use in left-hand panel below “Schemas” to tell SQL which database to use.

```
CREATE TABLE classics (  
  author VARCHAR(128),  
  title VARCHAR(128),  
  type VARCHAR(16),  
  year CHAR(4),  
  id INT UNSIGNED NOT NULL AUTO_INCREMENT KEY)  
ENGINE MyISAM;
```

DESCRIBE classics;

SQL Demo – Add Data and Add Column

```
INSERT INTO classics (author, title, type, year) VALUES('Mark Twain','The Adventures of Tom Sawyer','Fiction','1876');INSERT INTO classics(author, title, type, year) VALUES('Jane Austen','Pride and Prejudice','Fiction','1811');INSERT INTO classics(author, title, type, year) VALUES('Charles Darwin','The Origin of Species','Non-Fiction','1856');INSERT INTO classics(author, title, type, year) VALUES('Charles Dickens','The Old Curiosity Shop','Fiction','1841');INSERT INTO classics(author, title, type, year) VALUES('William Shakespeare','Romeo and Juliet','Play','1594');
```

```
DESCRIBE classics;
```

```
ALTER TABLE classics ADD isbn CHAR(13);
```

```
UPDATE classics SET isbn='9781598184891' WHERE year='1876';
```

```
UPDATE classics SET isbn='9780582506206' WHERE year='1811';
```

```
UPDATE classics SET isbn='9780517123201' WHERE year='1856';
```

```
UPDATE classics SET isbn='9780099533474' WHERE year='1841';
```

```
UPDATE classics SET isbn='9780192814968' WHERE year='1594';
```

```
DESCRIBE classics;
```

SQL Demo – Querying MySQL Database

SELECT *something* FROM *tablename*;

SELECT author FROM classics;

SELECT DISTINCT author FROM classics;

SELECT author, title FROM classics;

**SELECT author, title FROM classics WHERE
author="Mark Twain";**

**SELECT author, title FROM classics WHERE
isbn="9781598184891";**

<http://comminfo.rutgers.edu/~aspoerri/Teaching/InfoTech/Lectures/Lec10/Steps/MySQLdemo.txt>