WEEK 8

MORE SQL – THE SYSTEM CATALOG, AKA THE METADATA

STUDENT OBJECTIVES

- Upon completion of this video, you should be able to:
  - Describe how a DBMS is structured
  - List at least 4 tables in the system catalog
  - List at least 3 useful queries that you could use the system catalog to help you answer
  - Display the tables in a database in 2 different ways.

SYSTEM TABLES

QUESTION: What do you do if you create a table and can't remember what type a particular column is?

ANSWER:

SHOW TABLES;

DESCRIBE TABLE;

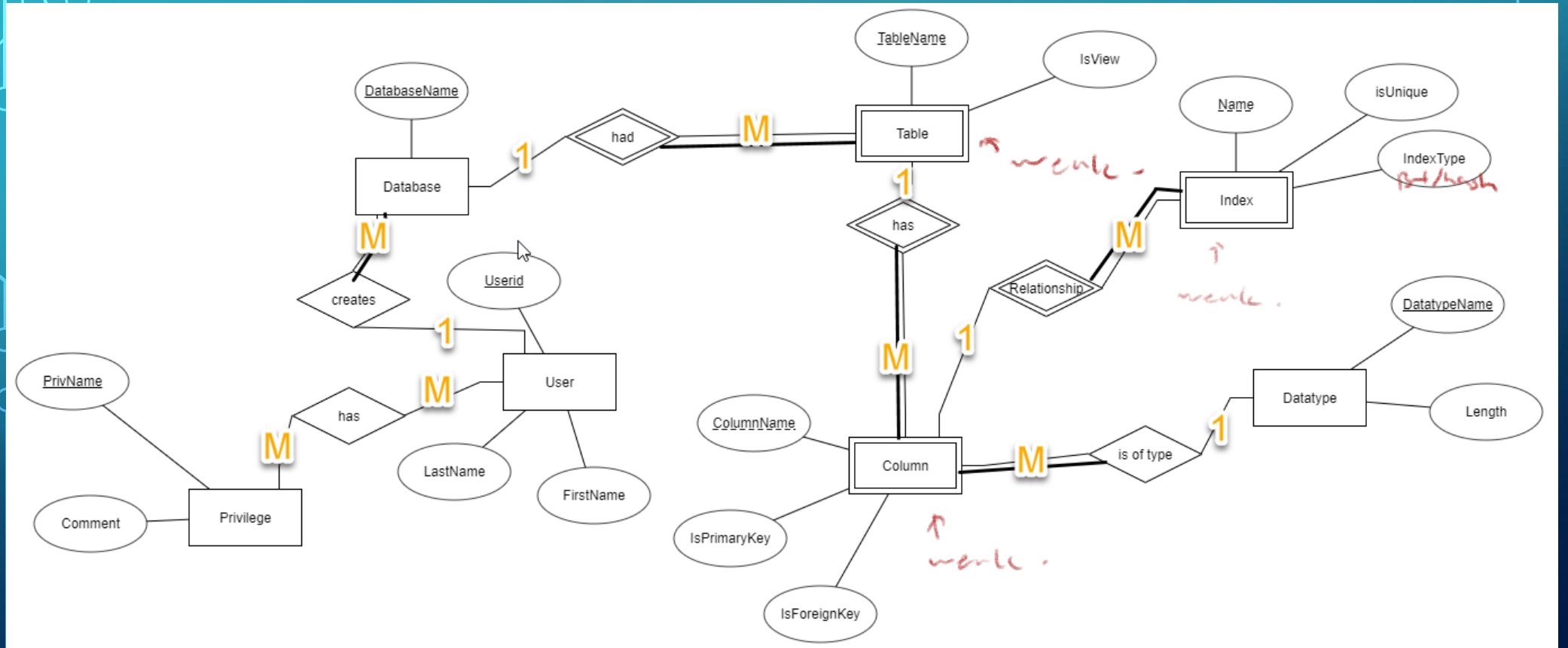
QUESTION: What are some of the *things* (components) in a relational database:

Databases, Tables, Columns, DataTypes, Views;

QUESTION: What is another word for a *thing* in an ER diagram?

Entity

QUESTION: Draw a very simple ER diagram of a relational database:



QUESTION: Map your ER diagram to a relational model and fill in some of the tables with a few records:

Databases

DatabaseName	CreatorID*
vetoffice	lreid
assign2	lreid
Information_schema	mysql

TABLES

DatabaseName*	Table_Name	Autoincrement
vetoffice	pet	No
vetoffice	owner	Yes
Information_schema	TABLES	Yes

COLUMNS

DatabaseName*	Table_Name*	Column_Name	DataType*
vetoffice	pet	petName	Varchar(40)
vetoffice	pet	petID	INT
vetoffice	pet	species	Varchar(40)
Information_schema	TABLES	Table_Name	Varchar(40)
Information_schema	COLUMNS	Table_Name	Varchar(40)
Information_schema	COLUMNS	Column_Name	Varchar(40)

QUESTION: How do you think MySql represents your database?

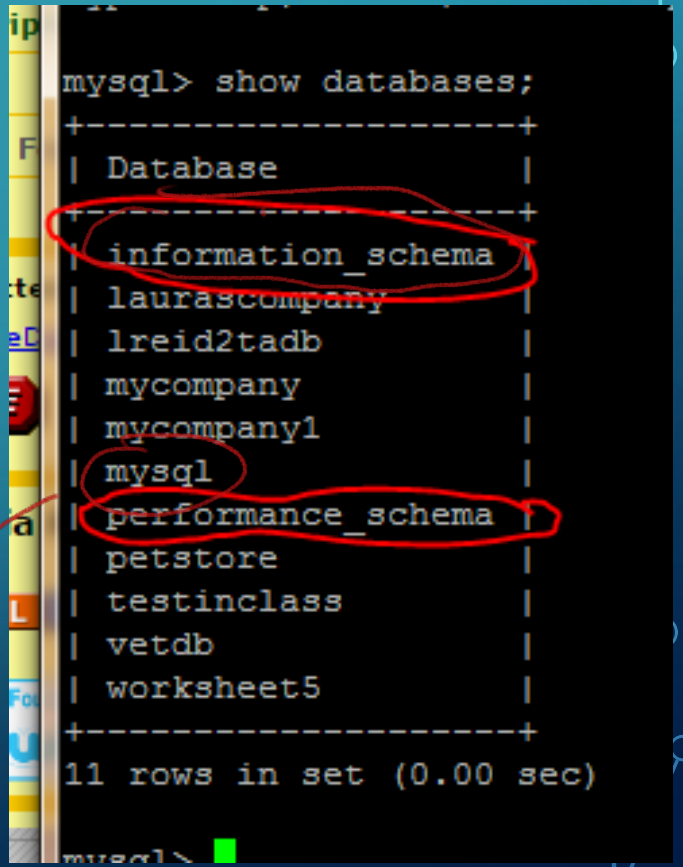
- MySQL represents your databases as a bunch of tables, just like you represent your database!
- These tables are called System Tables or System Catalog or Data Dictionary or Metadata or a mini database which describes your database.
- System Catalog keeps track of all the table names, attribute names, attribute domains, descriptions of constraints, etc...
- The Systems tables are hidden from you slightly but you can see them if you really want to ☺

But you cannot modify it.

VIEWING THE MYSQL SYSTEM CATALOG

- Notice the 2 databases circled.
- You did NOT create those
- They can be very useful, as you will see.
- You can only read from these, you can NOT do any inserts, updates or delete operations
- Need the *use* command →

  USE information_schema;



TRY THESE SQL COMMANDS IN MYSQL ON YOUR VIRTUAL MACHINE:

SHOW DATABASES;

OR

SELECT SCHEMA_NAME FROM information_schema.SCHEMATA;

OR

USE information_schema;

SELECT SCHEMA_NAME FROM SCHEMATA;

SELECT * FROM SCHEMATA;

- QUESTION: so what does the SCHEMATA table hold?
- ANSWER: all our databases (a row for each database)

- QUESTION: What is the SCHEMA_NAME?
- ANSWER: the UNIQUE name of each database (the primary key for this table!)

NOW TRY THIS…

SHOW TABLES;

OR

SELECT table_name FROM information_schema.tables;

OR

USE information_schema;

SELECT table_name FROM tables;

*SELECT * FROM tables;* ***(watch what happens with this command)***

QUESTION: HOW WOULD YOU ONLY SHOW THE TABLE NAMES IN A PARTICULAR DATABASE?

To figure this out, do this command first:

SELECT * FROM information_schema.TABLES;

Then consider how you would modify this command to show just the tables in the ***vetoffice*** database?

```
SELECT table_name FROM information_schema.TABLES
 WHERE table_schema = "vetoffice";
```

TYPE OF INFORMATION YOU CAN GET FROM THE SYSTEM CATALOG:

- This command:

```
SELECT table_name FROM
  information_schema.TABLES WHERE
  table_schema = 'information_schema';
```

returns this:

```
+---------------------------------------+
| TABLE_NAME                            |
+---------------------------------------+
| CHARACTER_SETS                        |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS                               |
| COLUMN_PRIVILEGES                     |
| ENGINES                               |
| EVENTS                                |
| FILES                                 |
| GLOBAL_STATUS                         |
| GLOBAL_VARIABLES                      |
| KEY_COLUMN_USAGE                      |
| PARTITIONS                            |
| PLUGINS                               |
| PROCESSLIST                           |
| PROFILING                             |
| REFERENTIAL_CONSTRAINTS               |
| ROUTINES                              |
| SCHEMATA                              |
| SCHEMA_PRIVILEGES                     |
| SESSION_STATUS                        |
| SESSION_VARIABLES                     |
| STATISTICS                            |
| TABLES                                |
| TABLESPACES                           |
| TABLE_CONSTRAINTS                     |
| TABLE_PRIVILEGES                      |
| TRIGGERS                              |
| USER_PRIVILEGES                       |
| VIEWS                                 |
| INNODB_BUFFER_PAGE                    |
```

- All the information schema views are created when a database is created with the CREATE DATABASE command
- The catalog views and tables CANNOT be explicitly dropped or created
- The catalog views and tables are updated as you perform SQL commands.

Question: Which of the following 2 commands do you think will update the catalog tables/view?

Both!

INSERT INTO pet VALUES ("dog","Scruffy",22);

ALTER TABLE pet ADD COLUMN weight INT;

- How could we check?

*SELECT * FROM information_schema.columns WHERE table_name = 'pet';*

WHY ARE SYSTEM/CATALOG TABLES USEFUL?

- What if you want to see which tables are really big (more than a given number of rows?)

SELECT CONCAT(table_schema, '.' ,table_name) as table_name, table_rows FROM information_schema.tables WHERE table_rows > 1000 AND table_schema not in('information_schema','mysql','performance_schema') ORDER BY table_rows desc;

```
+-----------------------------------+------------+
| table_name                        | table_rows |
+-----------------------------------+------------+
| customers.orders                  | 2007       |
| customers.contact_info            | 1245       |
| customers.rewards_points          | 2147       |
| business_contacts.company_info    | 1340       |
| business_contacts.phone_numbers   | 1712       |
| sonar.project_measures            | 178618     |
| sonar.resource_index              | 110328     |
| sonar.rule_failures               | 40793      |
+-----------------------------------+------------+
```

OTHER THINGS YOU COULD DO

- Write a query to list all the tables without a primary key
  - ***SELECT t.table_name FROM tables t LEFT JOIN TABLE_CONSTRAINTS tc ON t.table_schema = tc.table_schema AND t.table_name = tc.table_name AND constraint_type='PRIMARY KEY' WHERE tc.constraint_name IS NULL AND t.table_type='BASE TABLE';***
- Find the top 5 largest tables in a database
  - *SELECT table_schema, table_name, data_length FROM tables ORDER BY data_length DESC LIMIT 5;*