

# MySQL

Not *your* SQL, understand?

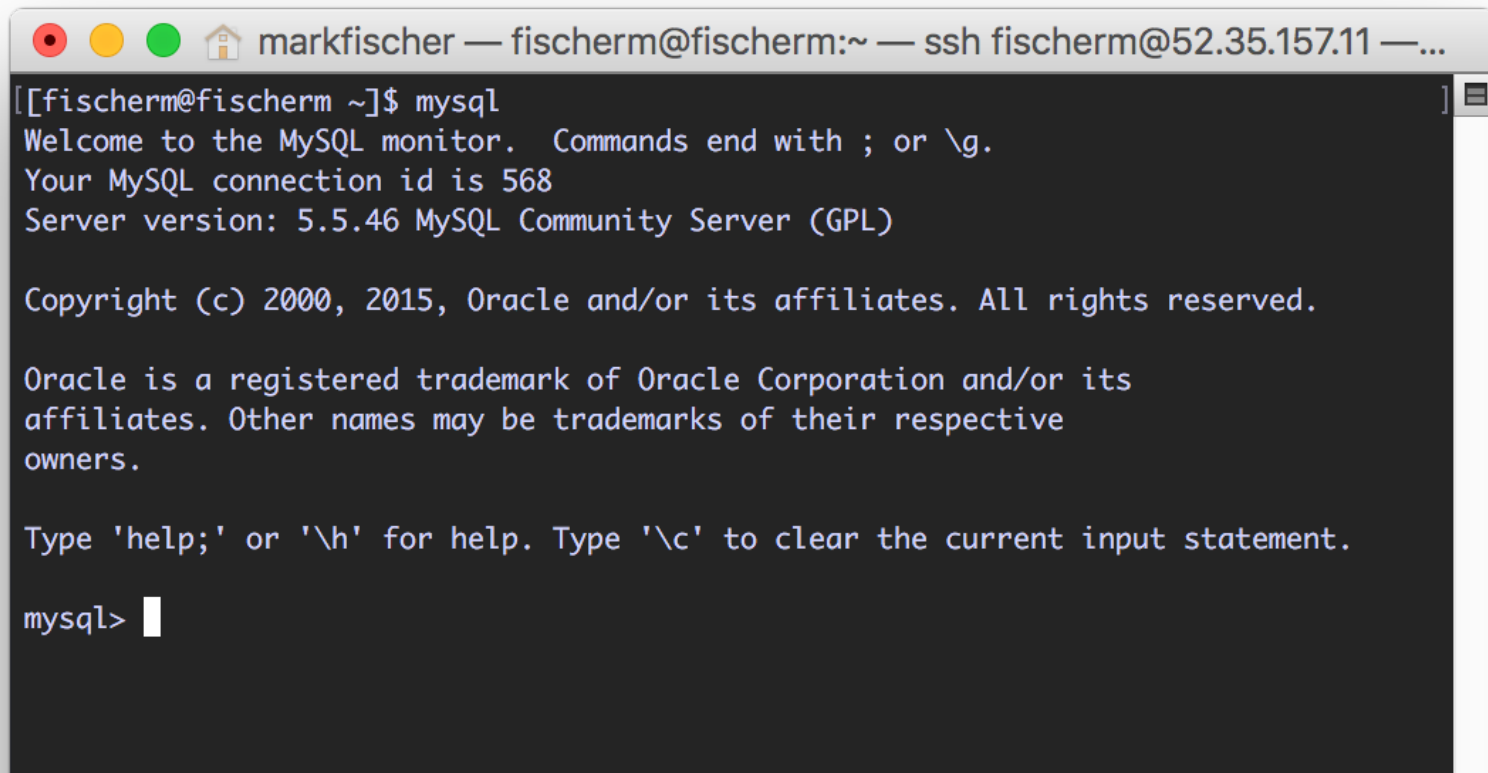
# MySQL

- Relational Database Management System
  - RDBMS
- Stores stuff in Tables
- Tables have named columns
- Tables have multiple rows with the same columns for each row
- Tables can be related to each other

# Connecting

- AWS VM
- From your command line:

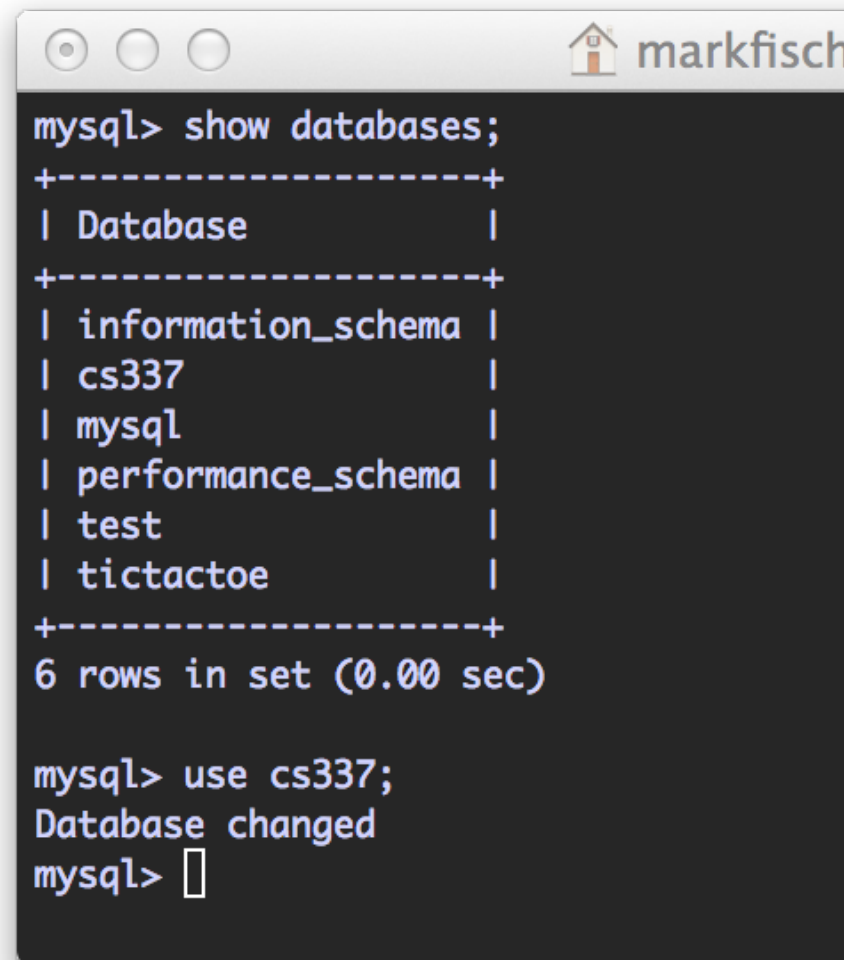
```
$ mysql
```

A terminal window with a dark background and light text. The window title bar shows 'markfischer — fischerm@fischerm:~ — ssh fischerm@52.35.157.11 —...'. The terminal content shows the execution of the 'mysql' command, resulting in a MySQL monitor prompt. The output includes a welcome message, connection ID (568), server version (5.5.46 MySQL Community Server (GPL)), and copyright information. The prompt 'mysql>' is visible at the bottom with a cursor.

```
markfischer — fischerm@fischerm:~ — ssh fischerm@52.35.157.11 —...  
[[fischerm@fischerm ~]$ mysql  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 568  
Server version: 5.5.46 MySQL Community Server (GPL)  
  
Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql> |
```

# Databases

- `show databases;`
- Lists all the databases on this server
- `use <database>;`
- Select a database to send commands to



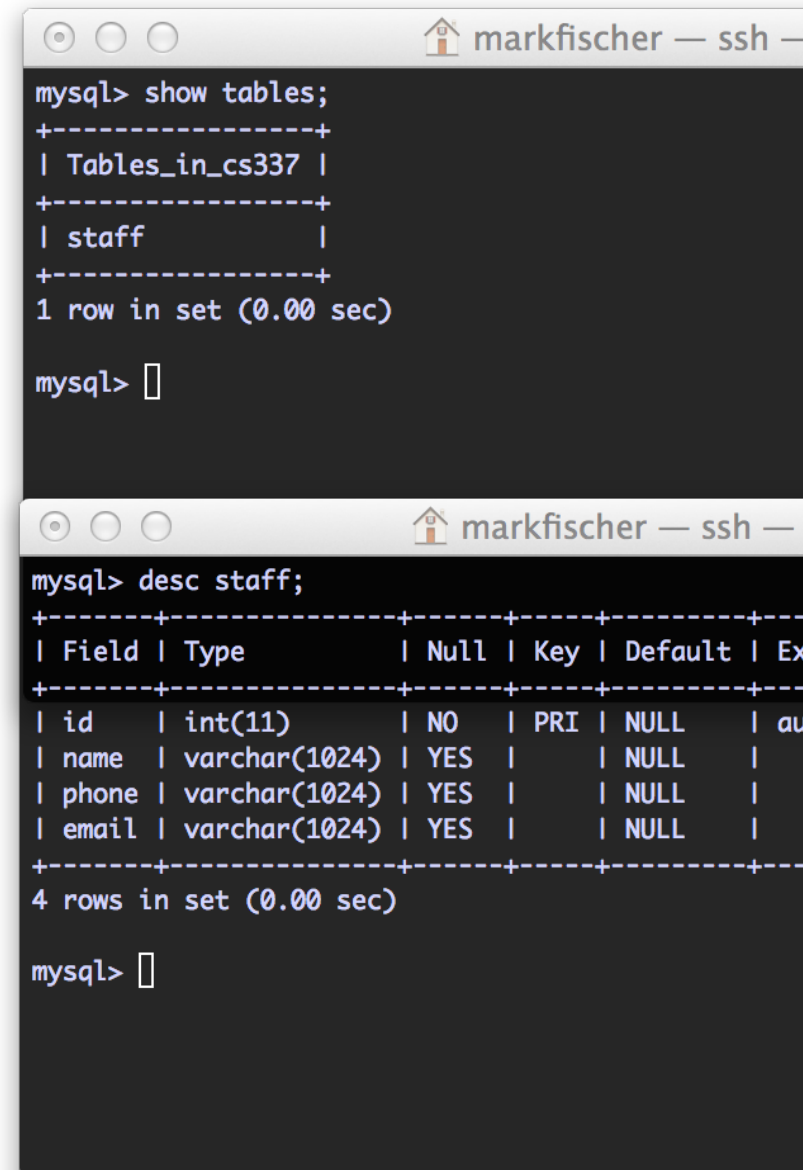
A terminal window with a title bar containing three window control buttons and the text "markfisch". The terminal displays the following MySQL commands and output:

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| cs337 |
| mysql |
| performance_schema |
| test |
| tictactoe |
+-----+
6 rows in set (0.00 sec)

mysql> use cs337;
Database changed
mysql> 
```

# Looking At Tables

- `show tables;`
- Lists all tables in the database
- `describe <tablename>;`
- Print out the column structure of the given table



The image shows two terminal windows. The top window displays the command `mysql> show tables;` and its output, which lists two tables: `Tables_in_cs337` and `staff`. The bottom window displays the command `mysql> desc staff;` and its output, which shows the column structure of the `staff` table.

```
mysql> show tables;
+-----+
| Tables_in_cs337 |
+-----+
| staff           |
+-----+
1 row in set (0.00 sec)

mysql>

mysql> desc staff;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Ex  |
+-----+-----+-----+-----+-----+-----+
| id    | int(11)       | NO   | PRI | NULL    | au  |
| name  | varchar(1024) | YES  |     | NULL    |    |
| phone | varchar(1024) | YES  |     | NULL    |    |
| email | varchar(1024) | YES  |     | NULL    |    |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

# SQL

- SQL - **S**tructured **Q**uery **L**anguage
- An english like syntax to interact with a databases
- Basic Verbs initiate Commands
  - **SELECT**
  - **INSERT**
  - **UPDATE**
  - **DELETE**

# CREATE TABLE

- Make a new table to hold stuff
- Think about the columns you want to have in your table
- Data Modeling

```
CREATE TABLE `staff` (  
  `id` int(11) NOT NULL auto_increment,  
  `name` varchar(1024) default NULL,  
  `phone` varchar(1024) default NULL,  
  `email` varchar(1024) default NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

# MySQL Datatypes

<http://dev.mysql.com/doc/en/data-types.html>

- Several ways to hold a string
  - CHAR and VARCHAR
  - Also BLOB and TEXT
- Numbers
  - INT, SMALLINT, BIGINT etc
  - DECIMAL, NUMERIC, FLOAT, DOUBLE, BIT
- Dates & Times
  - DATE, TIME, TIMESTAMP, DATETIME



# CRUD

- You'll hear people mention CRUD in connection with databases
  - **C**reate
  - **R**etrieve
  - **U**ppdate
  - **D**elete

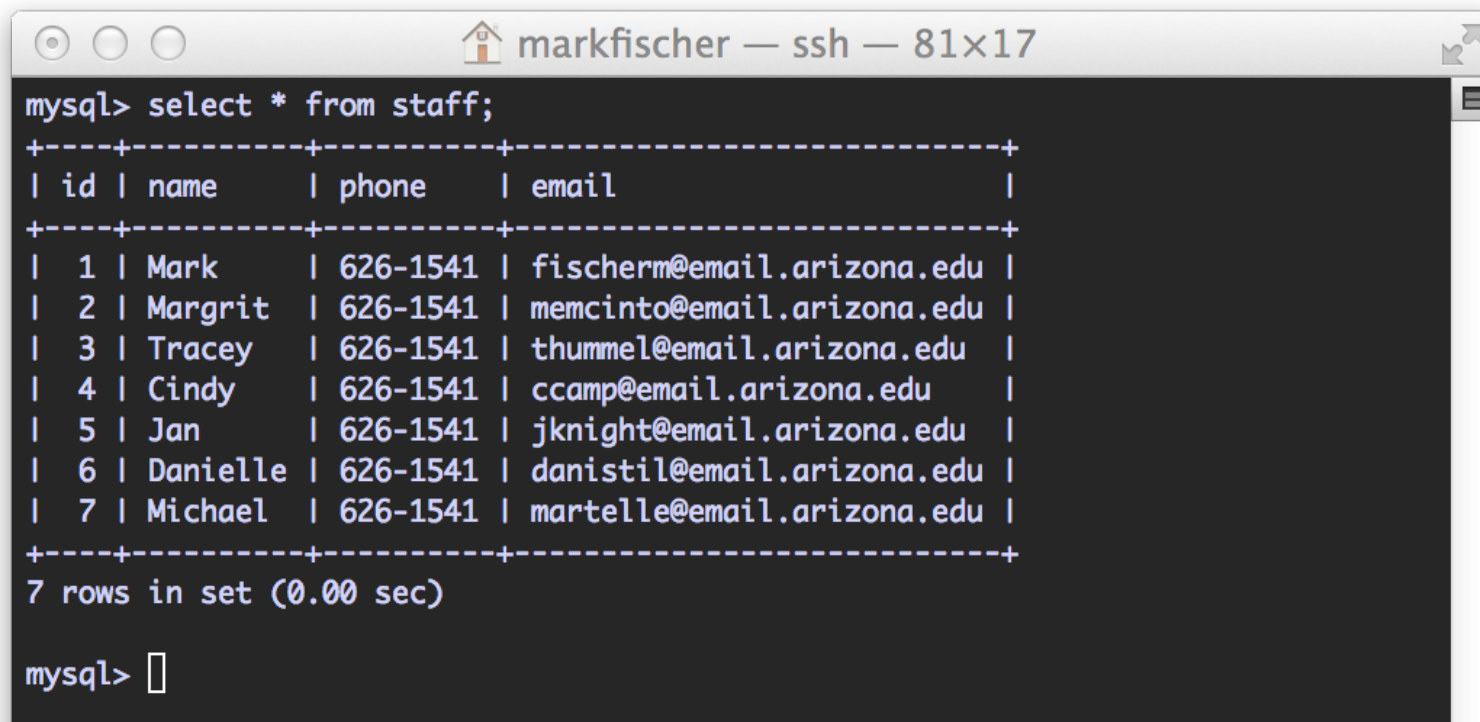
# SQL

CRUD	SQL Verb
Create	INSERT
Retrieve	SELECT
Update	UPDATE
Delete	DELETE

# select

- Getting data out of tables

```
SELECT <fields> FROM <tables> [WHERE <conditions>];
```



A terminal window titled "markfischer — ssh — 81x17" showing a MySQL session. The user enters the command "mysql> select \* from staff;". The output is a table with 7 rows and 4 columns: id, name, phone, and email. The data is as follows:

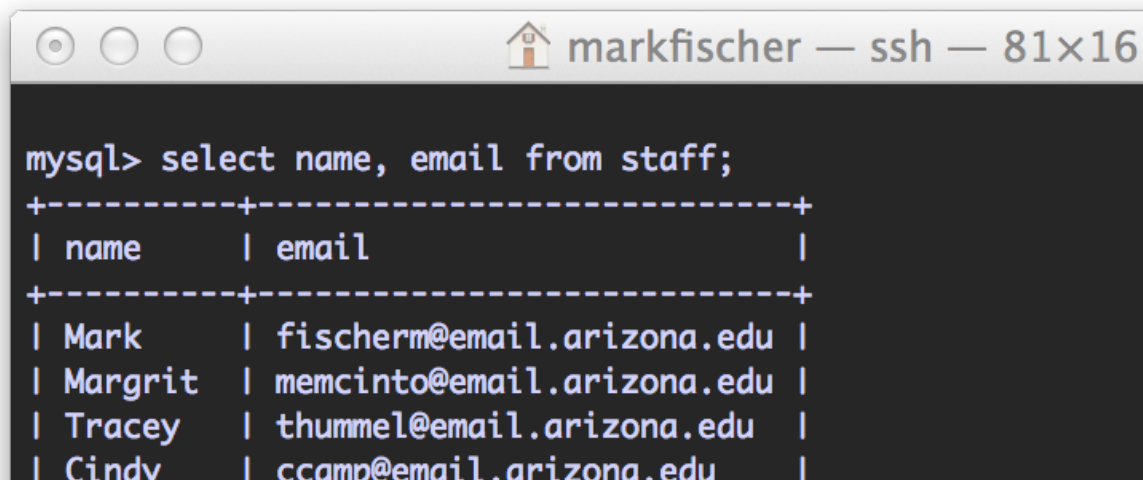
id	name	phone	email
1	Mark	626-1541	fischerm@email.arizona.edu
2	Margrit	626-1541	memcinto@email.arizona.edu
3	Tracey	626-1541	thummel@email.arizona.edu
4	Cindy	626-1541	ccamp@email.arizona.edu
5	Jan	626-1541	jknight@email.arizona.edu
6	Danielle	626-1541	danistil@email.arizona.edu
7	Michael	626-1541	martelle@email.arizona.edu

Below the table, the terminal shows "7 rows in set (0.00 sec)" and the prompt "mysql> " with a cursor.

# select

- SQL is case in-sensitive
- These all work the same
- The Asterisk '\*' means “All the fields in the tables”
- Can select just specific fields by specifying which ones

```
select * from staff;  
SELECT * FROM staff;  
Select * From Staff;
```



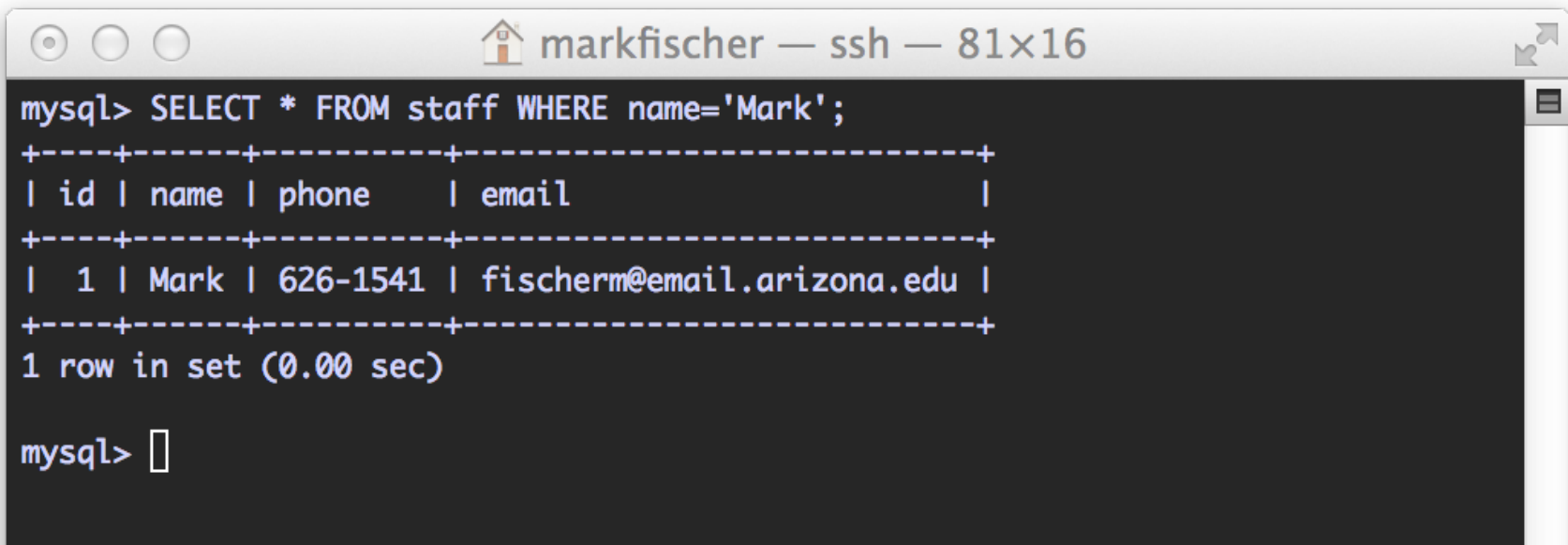
A terminal window titled "markfischer — ssh — 81x16" showing a MySQL query and its output. The query is "mysql> select name, email from staff;". The output is a table with two columns: "name" and "email". The rows are: Mark (fischerm@email.arizona.edu), Margrit (memcinto@email.arizona.edu), Tracey (thummel@email.arizona.edu), and Cindy (ccamp@email.arizona.edu).

```
mysql> select name, email from staff;  
+-----+-----+  
| name   | email                               |  
+-----+-----+  
| Mark   | fischerm@email.arizona.edu         |  
| Margrit | memcinto@email.arizona.edu        |  
| Tracey | thummel@email.arizona.edu         |  
| Cindy  | ccamp@email.arizona.edu           |
```

# selecting specific things

- The WHERE clause for a SELECT statement allows us to limit the rows selected from a set of tables

```
SELECT * FROM staff WHERE name='Mark';
```



A terminal window titled "markfischer — ssh — 81x16" showing a MySQL command and its output. The command is "mysql> SELECT \* FROM staff WHERE name='Mark';". The output is a table with columns "id", "name", "phone", and "email". The table contains one row with the values "1", "Mark", "626-1541", and "fischerm@email.arizona.edu". Below the table, it says "1 row in set (0.00 sec)". The prompt "mysql>" is followed by a cursor.

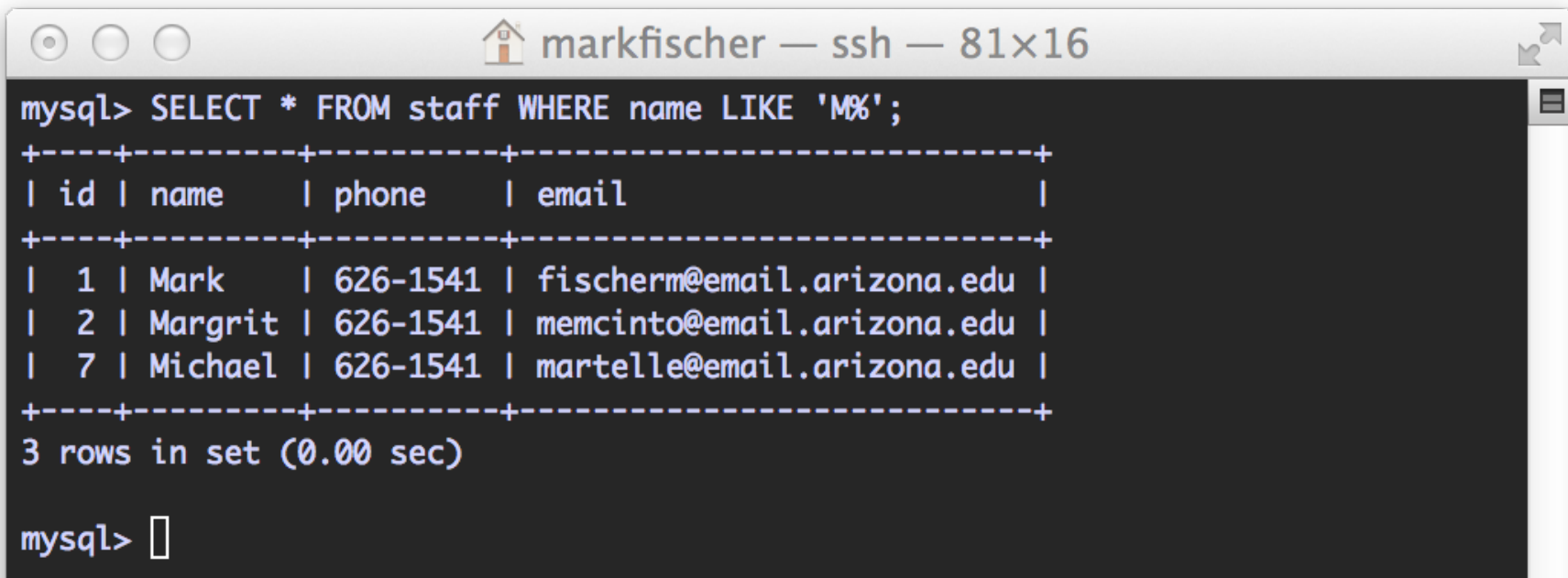
```
mysql> SELECT * FROM staff WHERE name='Mark';
+----+-----+-----+-----+
| id | name | phone   | email                               |
+----+-----+-----+-----+
|  1 | Mark | 626-1541 | fischerm@email.arizona.edu         |
+----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> █
```

# selecting specific things

- Doesn't have to be an *exact* match – LIKE
- % is our wildcard match character for strings in SQL

```
SELECT * FROM staff WHERE name LIKE 'M%';
```



A terminal window titled "markfischer — ssh — 81x16" showing a MySQL query and its results. The query is "mysql> SELECT \* FROM staff WHERE name LIKE 'M%';". The output is a table with 4 columns: id, name, phone, and email. The results are 3 rows: Mark (id 1), Margrit (id 2), and Michael (id 7). The terminal also shows "3 rows in set (0.00 sec)" and a prompt "mysql> " with a cursor.

```
mysql> SELECT * FROM staff WHERE name LIKE 'M%';
+----+-----+-----+-----+
| id | name  | phone | email                               |
+----+-----+-----+-----+
| 1  | Mark  | 626-1541 | fischerm@email.arizona.edu |
| 2  | Margrit | 626-1541 | memcinto@email.arizona.edu |
| 7  | Michael | 626-1541 | martelle@email.arizona.edu |
+----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> 
```

# insert

- Adding new rows to a table
- Values must match positions with their field names
- Values must be correct for the datatype of the field
- Strings must be surrounded by single quotes – 'some string'

```
INSERT INTO <table>  
(field1, field2, ...) VALUES (value1, value2, ...);
```

```
mysql> select * from staff;
```

```
+-----+-----+-----+-----+
| id | name      | phone    | email                                     |
+-----+-----+-----+-----+
| 1  | Mark      | 626-1541 | fischerm@email.arizona.edu             |
| 2  | Margrit   | 626-1541 | memcinto@email.arizona.edu             |
| 3  | Tracey    | 626-1541 | thummel@email.arizona.edu              |
| 4  | Cindy     | 626-1541 | ccamp@email.arizona.edu                 |
| 5  | Jan       | 626-1541 | jknight@email.arizona.edu               |
| 6  | Danielle  | 626-1541 | danistil@email.arizona.edu              |
| 7  | Michael   | 626-1541 | martelle@email.arizona.edu              |
+-----+-----+-----+-----+
```

```
7 rows in set (0.00 sec)
```

```
mysql> INSERT INTO staff (name, phone, email) VALUES ('Adam', '621-1541', 'adam@email.arizona.edu');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> select * from staff;
```

```
+-----+-----+-----+-----+
| id | name      | phone    | email                                     |
+-----+-----+-----+-----+
| 1  | Mark      | 626-1541 | fischerm@email.arizona.edu             |
| 2  | Margrit   | 626-1541 | memcinto@email.arizona.edu             |
| 3  | Tracey    | 626-1541 | thummel@email.arizona.edu              |
| 4  | Cindy     | 626-1541 | ccamp@email.arizona.edu                 |
| 5  | Jan       | 626-1541 | jknight@email.arizona.edu               |
| 6  | Danielle  | 626-1541 | danistil@email.arizona.edu              |
| 7  | Michael   | 626-1541 | martelle@email.arizona.edu              |
| 8  | Adam      | 621-1541 | adam@email.arizona.edu                  |
+-----+-----+-----+-----+
```

```
8 rows in set (0.00 sec)
```

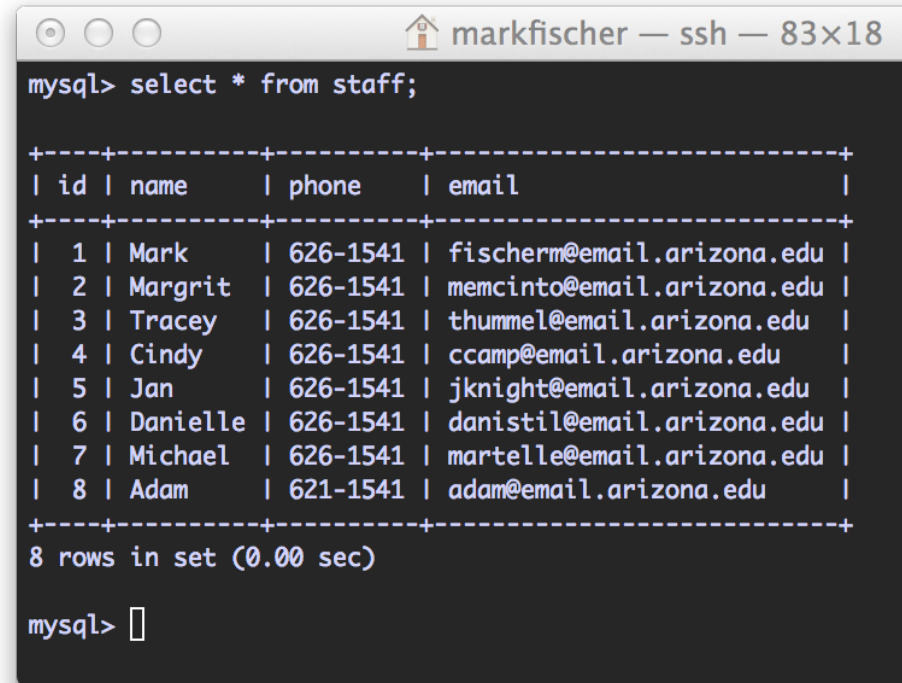
```
mysql> █
```



# insert

```
INSERT INTO staff
(name, phone, email) VALUES
('Adam', '621-1541', 'adam@email.arizona.edu');
```

- Why didn't we specify the `id` field?
- Where does the `8` come from?



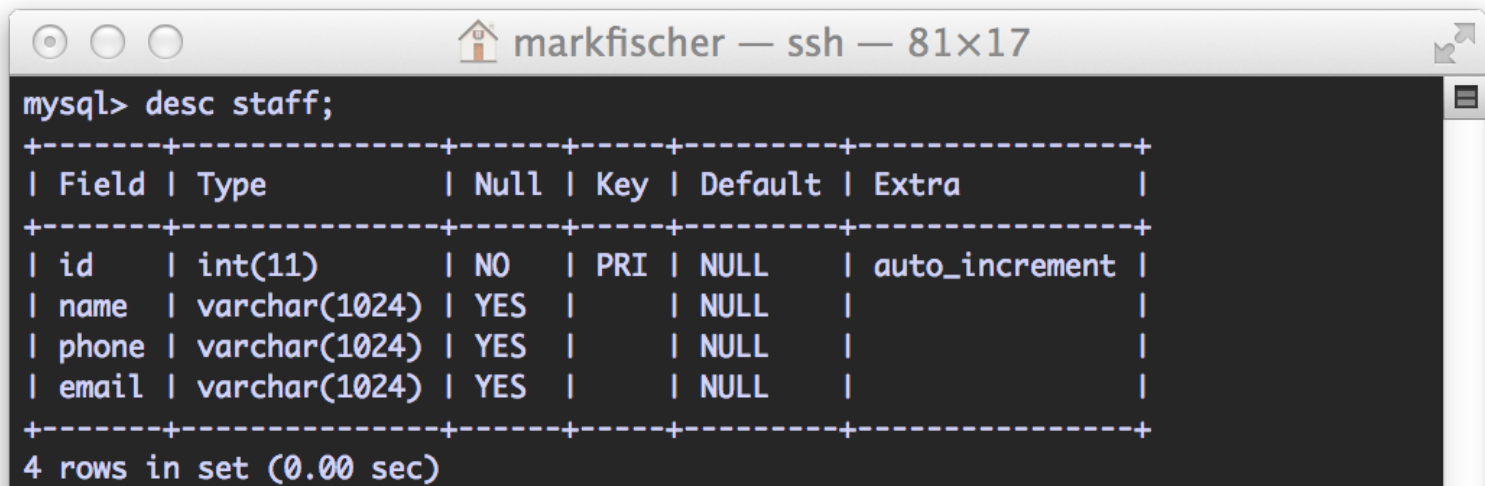
A terminal window titled 'markfischer — ssh — 83x18' showing a MySQL query and its results. The query is 'mysql> select \* from staff;'. The output is a table with 8 rows and 4 columns: id, name, phone, and email. The rows are: 1 | Mark | 626-1541 | fischerm@email.arizona.edu, 2 | Margrit | 626-1541 | memcinto@email.arizona.edu, 3 | Tracey | 626-1541 | thummel@email.arizona.edu, 4 | Cindy | 626-1541 | ccamp@email.arizona.edu, 5 | Jan | 626-1541 | jknight@email.arizona.edu, 6 | Danielle | 626-1541 | danistil@email.arizona.edu, 7 | Michael | 626-1541 | martelle@email.arizona.edu, and 8 | Adam | 621-1541 | adam@email.arizona.edu. Below the table, it says '8 rows in set (0.00 sec)'. The prompt 'mysql>' is followed by a cursor.

```
mysql> select * from staff;
+----+-----+-----+-----+
| id | name  | phone | email                               |
+----+-----+-----+-----+
| 1  | Mark  | 626-1541 | fischerm@email.arizona.edu |
| 2  | Margrit | 626-1541 | memcinto@email.arizona.edu |
| 3  | Tracey | 626-1541 | thummel@email.arizona.edu |
| 4  | Cindy  | 626-1541 | ccamp@email.arizona.edu |
| 5  | Jan    | 626-1541 | jknight@email.arizona.edu |
| 6  | Danielle | 626-1541 | danistil@email.arizona.edu |
| 7  | Michael | 626-1541 | martelle@email.arizona.edu |
| 8  | Adam   | 621-1541 | adam@email.arizona.edu |
+----+-----+-----+-----+
8 rows in set (0.00 sec)

mysql> █
```

# AUTO INCREMENT

- When defining a table, you can specify a PRIMARY KEY field be **AUTO INCREMENT**
- This does pretty much what it sounds like
- Anytime a new row is inserted into the table, MySQL will automatically assign a new value, incrementing an internal counter



```
markfischer — ssh — 81x17
mysql> desc staff;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id    | int(11)       | NO   | PRI | NULL    | auto_increment |
| name  | varchar(1024) | YES  |     | NULL    |                |
| phone | varchar(1024) | YES  |     | NULL    |                |
| email | varchar(1024) | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

# update

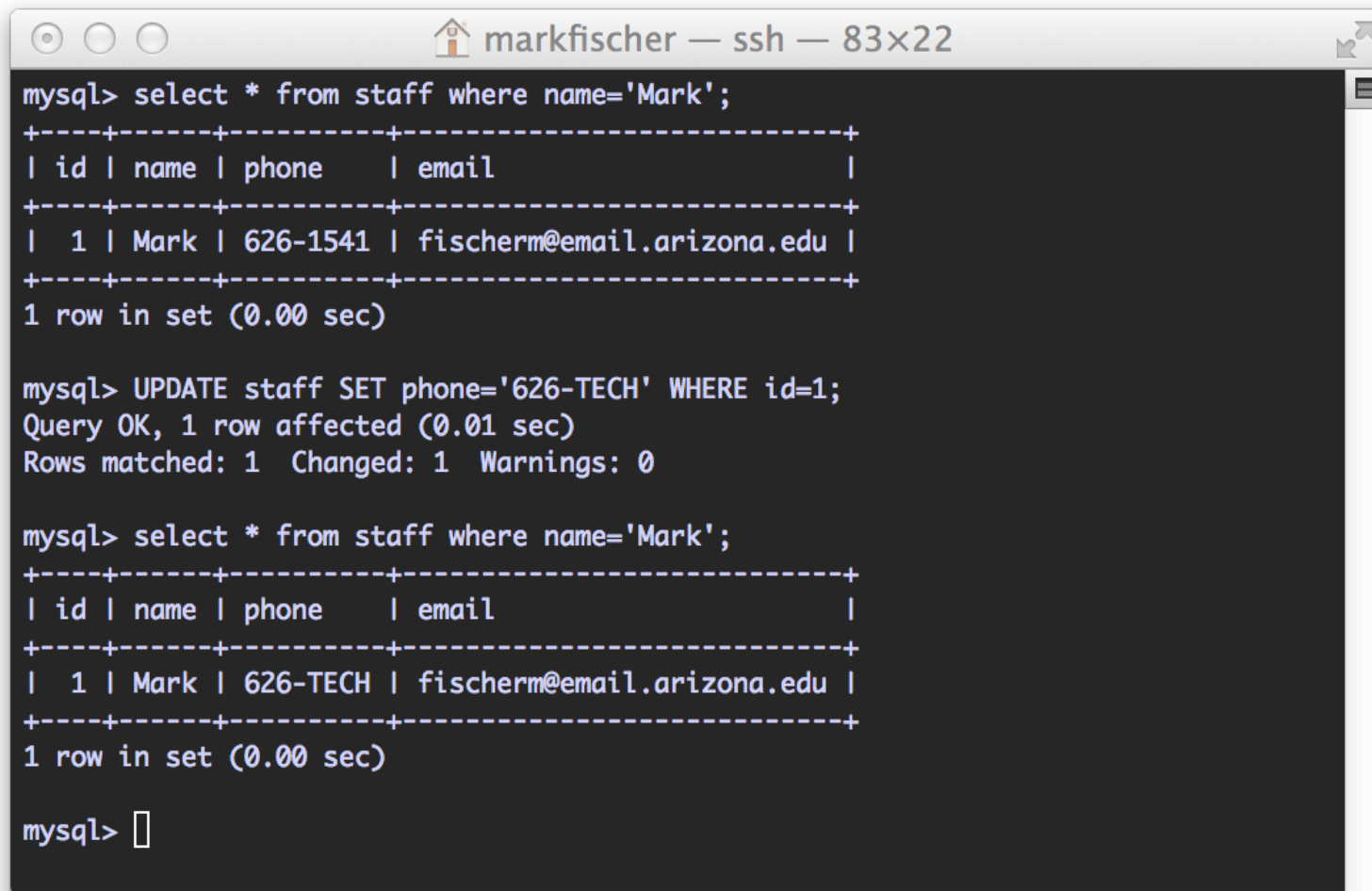
- Change a value for a field or set of fields.

```
UPDATE <table> SET field1=value1, field2=value2  
WHERE [conditions];
```

- **WATCH OUT!**
- If you don't specify any conditions, you will update **EVERY ROW!**

# update

```
UPDATE staff SET phone='626-TECH' WHERE id=1;
```



```
markfischer — ssh — 83x22
mysql> select * from staff where name='Mark';
+----+-----+-----+-----+
| id | name | phone   | email                               |
+----+-----+-----+-----+
|  1 | Mark | 626-1541 | fischerm@email.arizona.edu         |
+----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> UPDATE staff SET phone='626-TECH' WHERE id=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from staff where name='Mark';
+----+-----+-----+-----+
| id | name | phone   | email                               |
+----+-----+-----+-----+
|  1 | Mark | 626-TECH | fischerm@email.arizona.edu         |
+----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> █
```

# delete

- Deletes rows from a table

```
DELETE FROM <table> WHERE [conditions];
```

- **WATCH OUT!**
- If you don't specify any conditions, you will **DELETE EVERY ROW!**

```
DELETE FROM staff WHERE id=8;
```

```
markfischer — ssh — 83x33
mysql> select * from staff;
+----+-----+-----+-----+
| id | name   | phone | email                               |
+----+-----+-----+-----+
| 1  | Mark   | 626-TECH | fischerm@email.arizona.edu |
| 2  | Margrit | 626-1541 | memcinto@email.arizona.edu |
| 3  | Tracey | 626-1541 | thummel@email.arizona.edu |
| 4  | Cindy  | 626-1541 | ccamp@email.arizona.edu |
| 5  | Jan    | 626-1541 | jknight@email.arizona.edu |
| 6  | Danielle | 626-1541 | danistil@email.arizona.edu |
| 7  | Michael | 626-1541 | martelle@email.arizona.edu |
| 8  | Adam   | 621-1541 | adam@email.arizona.edu |
+----+-----+-----+-----+
8 rows in set (0.00 sec)

mysql> DELETE FROM staff WHERE id=8;
Query OK, 1 row affected (0.00 sec)

mysql> select * from staff;
+----+-----+-----+-----+
| id | name   | phone | email                               |
+----+-----+-----+-----+
| 1  | Mark   | 626-TECH | fischerm@email.arizona.edu |
| 2  | Margrit | 626-1541 | memcinto@email.arizona.edu |
| 3  | Tracey | 626-1541 | thummel@email.arizona.edu |
| 4  | Cindy  | 626-1541 | ccamp@email.arizona.edu |
| 5  | Jan    | 626-1541 | jknight@email.arizona.edu |
| 6  | Danielle | 626-1541 | danistil@email.arizona.edu |
| 7  | Michael | 626-1541 | martelle@email.arizona.edu |
+----+-----+-----+-----+
7 rows in set (0.00 sec)

mysql> 
```

HI, THIS IS  
YOUR SON'S SCHOOL.  
WE'RE HAVING SOME  
COMPUTER TROUBLE.



OH, DEAR - DID HE  
BREAK SOMETHING?  
IN A WAY- )



DID YOU REALLY  
NAME YOUR SON  
Robert'); DROP  
TABLE Students;-- ?



OH. YES. LITTLE  
BOBBY TABLES,  
WE CALL HIM.

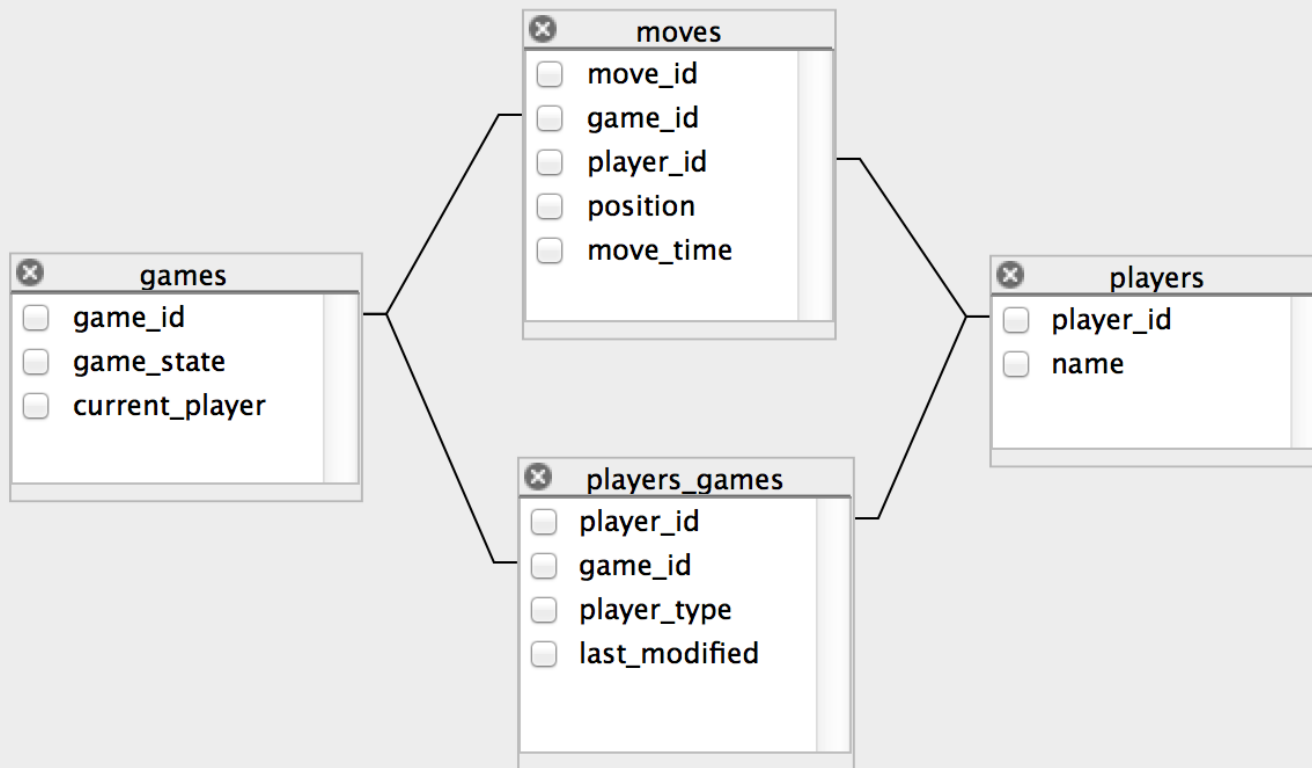
WELL, WE'VE LOST THIS  
YEAR'S STUDENT RECORDS.  
I HOPE YOU'RE HAPPY.



AND I HOPE  
YOU'VE LEARNED  
TO SANITIZE YOUR  
DATABASE INPUTS.

# Joins

- The Relational part of RDBMS

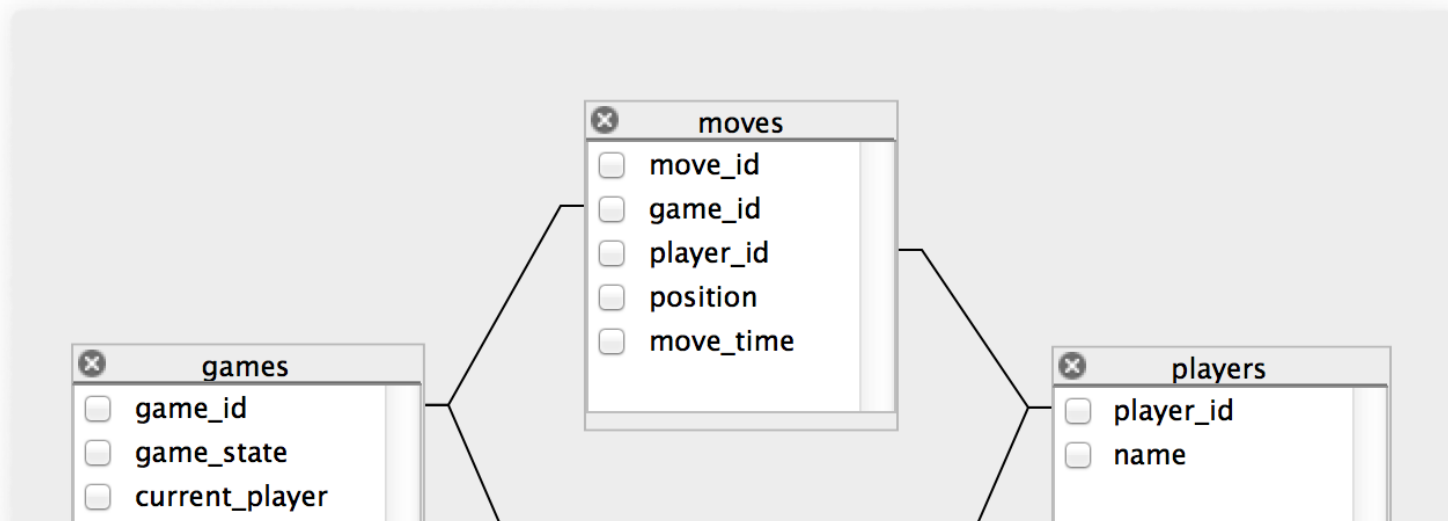




# Joins

- You can SELECT from multiple tables in a single query

```
SELECT games.game_state,  
       games.game_id,  
       players_games.player_id  
FROM players_games INNER JOIN games  
ON players_games.game_id = games.game_id;
```



# Joins

- When specifying fields to select from multiple tables, you prefix the field name by the table name
- `tablename.fieldname`

```
SELECT  
  games.game_state,  
  games.game_id,  
  players_games.player_id  
  ...
```

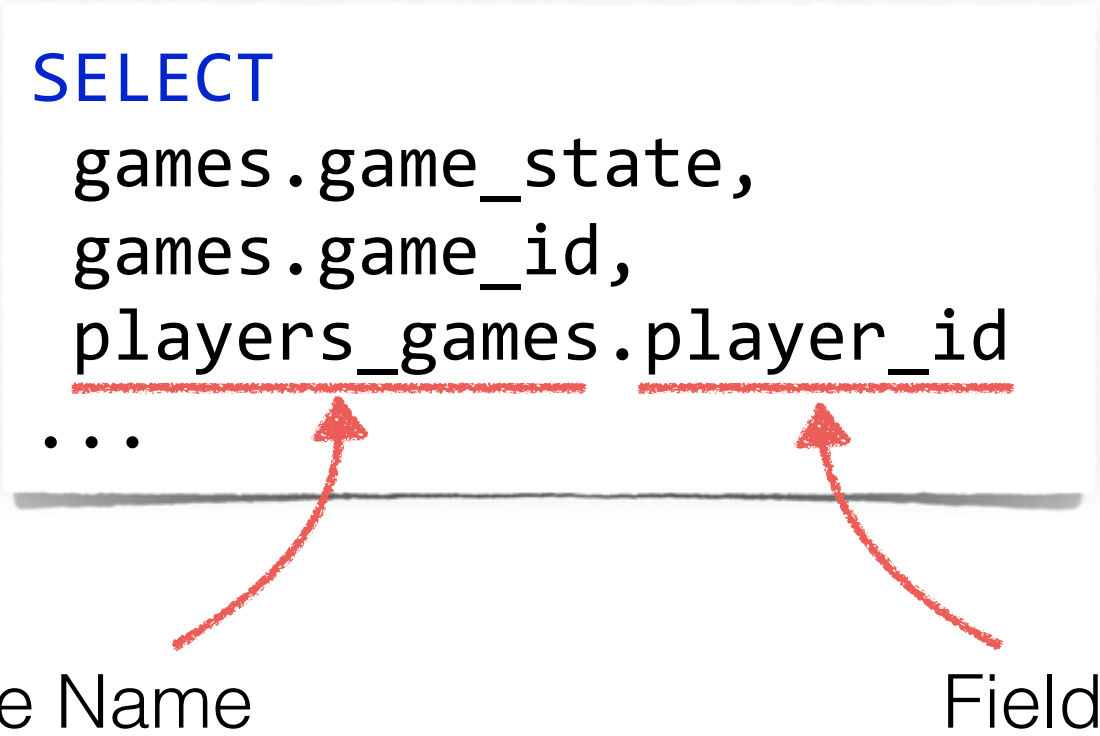


Table Name

Field Name

```
SELECT games.game_state,  
       games.game_id,  
       players_games.player_id  
FROM players_games INNER JOIN games  
ON players_games.game_id = games.game_id;
```

```
php — fischer@workbench:~ — ssh — 66x22  
mysql> SELECT games.game_state,  
-> games.game_id,  
-> players_games.player_id  
-> FROM players_games INNER JOIN games  
-> ON players_games.game_id = games.game_id;  
+-----+-----+-----+  
| game_state | game_id | player_id |  
+-----+-----+-----+  
| playing   | 60      | 57fd375464bab393065734b8d3e4cf1d |  
| playing   | 60      | d29c3e8b83e01f68e97458182e3d4039 |  
| ended     | 61      | d29c3e8b83e01f68e97458182e3d4039 |  
| ended     | 62      | d9dd132068c07304cab89c4659c80c0d |  
| ended     | 63      | 32d6df3d5e59f71a927ed5bea1a6c4bb |  
| ended     | 63      | d9dd132068c07304cab89c4659c80c0d |  
| open      | 64      | d9dd132068c07304cab89c4659c80c0d |  
+-----+-----+-----+  
7 rows in set (0.00 sec)  
  
mysql> █
```

# Lots Of Other Stuff

- Lots of built-in functions
  - ABS, AVG, POW, RAND, SYSDATE, VARIANCE
- Standard Operators
  - + - / \* = > etc
- Stored Procedures
  - Write your code directly in the database, then make SQL calls to the functions
- Can store JSON natively now
- Transactions

# Great!

Now go do all that from PHP!