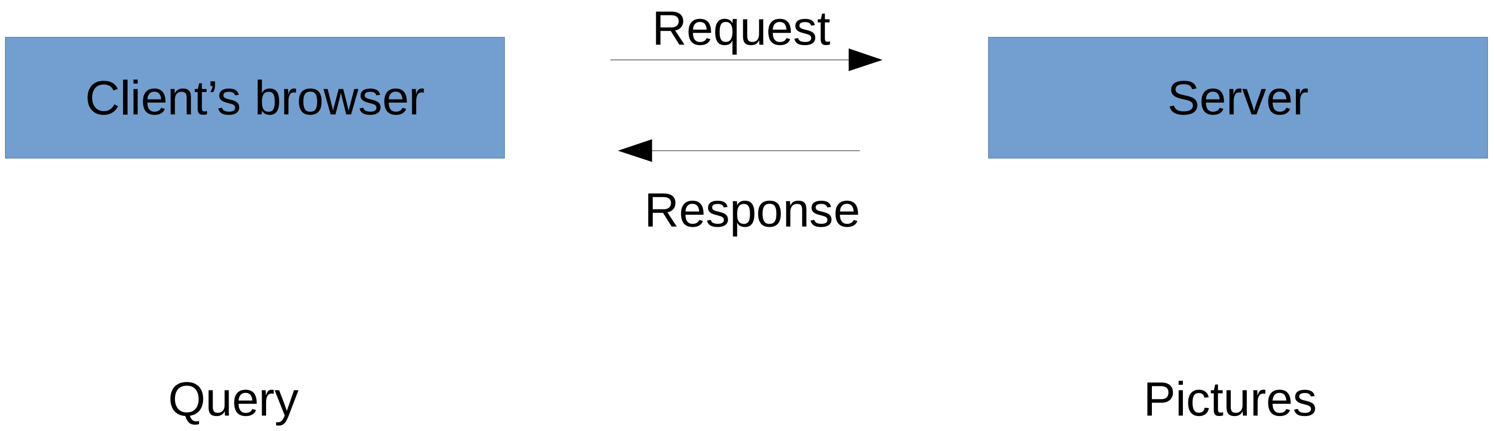
Database management (RDBMS)

What have we done so far?

- Basic shell commands
- Running as root from a virtual box
- Opening and communicating with remote computers using sockets

- What is the point of all this?
    - We want to set up web services of our own
    - We need to understand what the communication protocols that underpin these services can and cannot do

Architecture of a web service



Request

Client’s browser

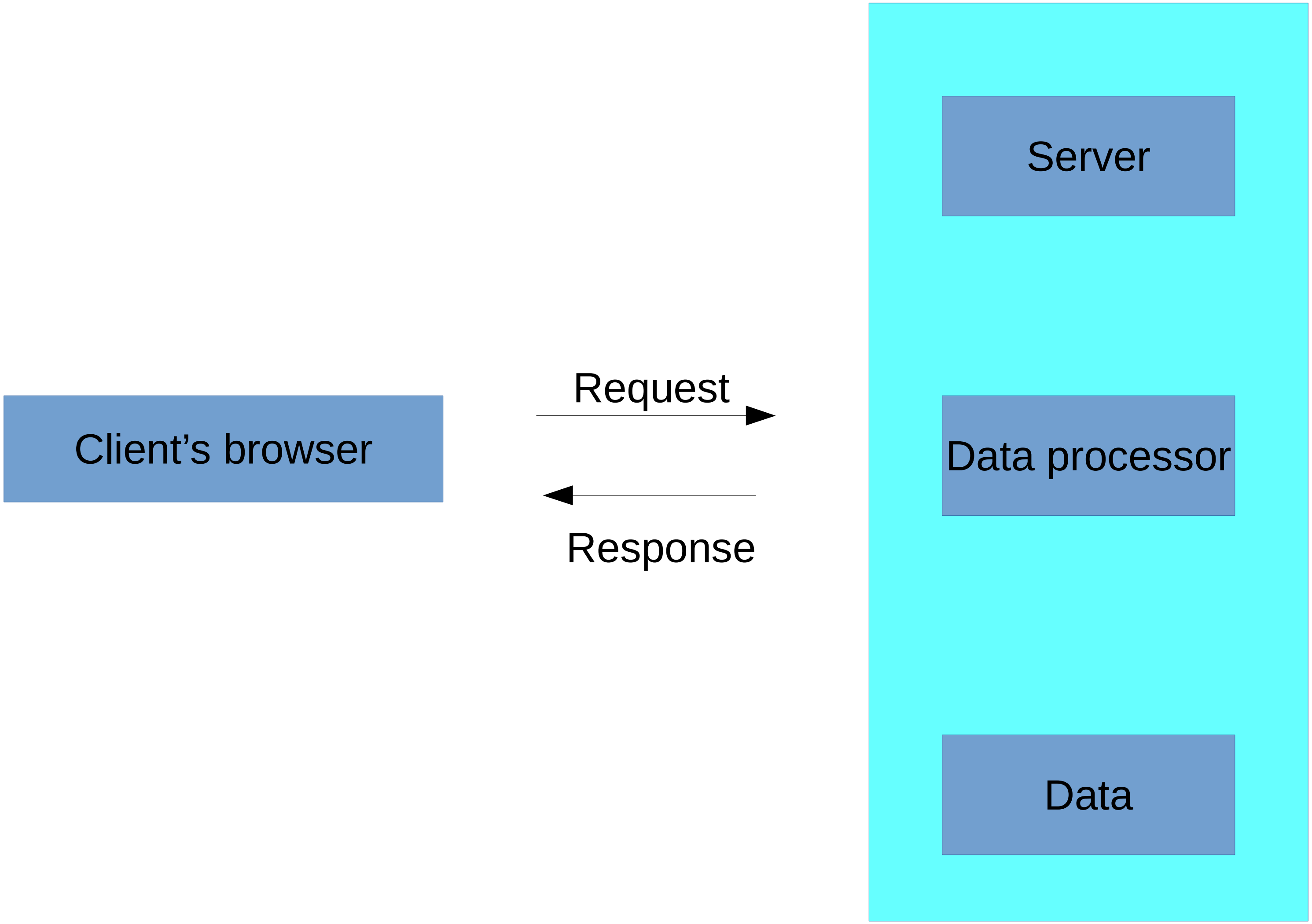
Server

Response

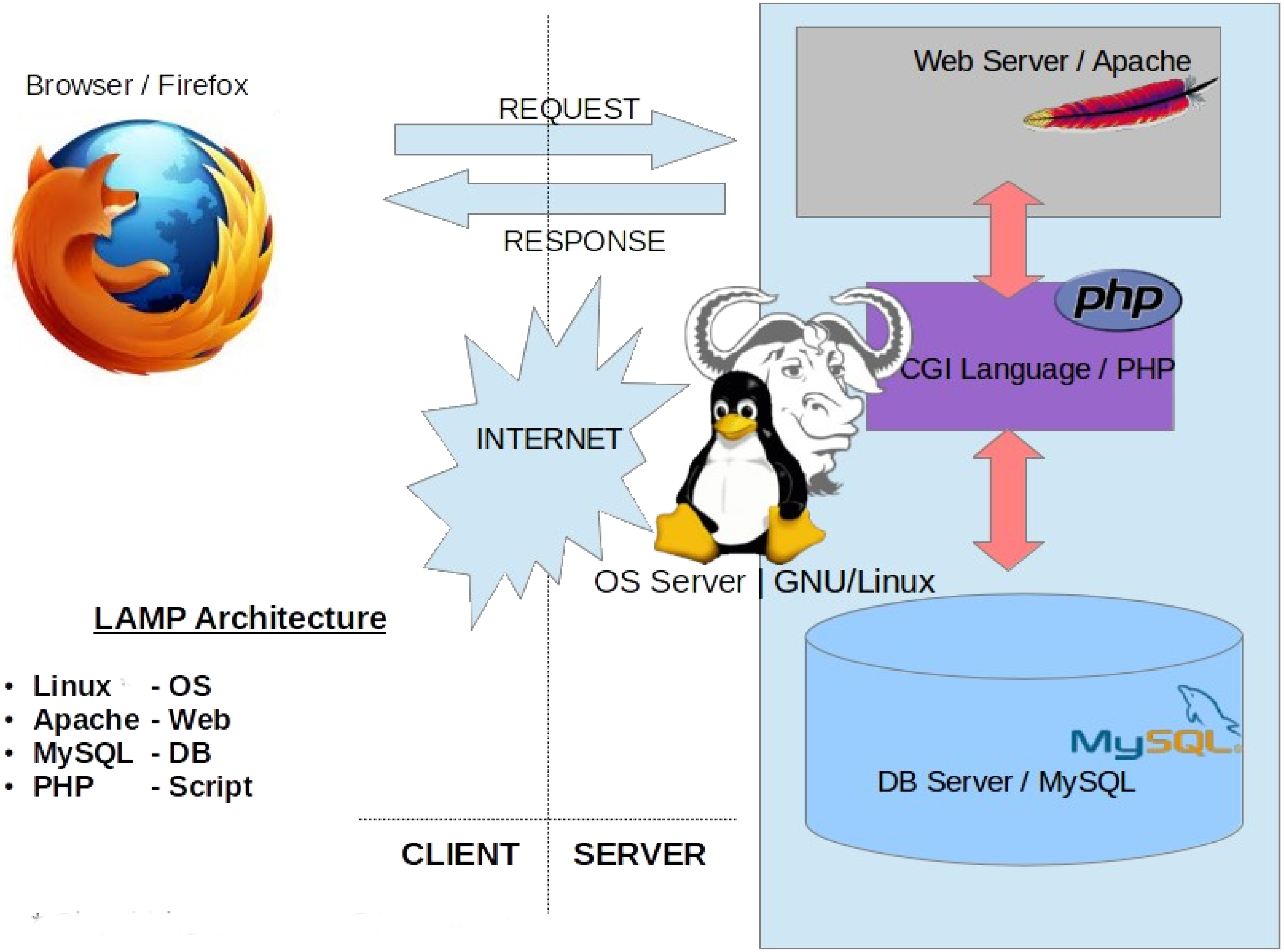
Query

Pictures

Architecture of a web service



LAMP stack



Browser / Firefox

REQUEST

RESPONSE

INTERNET

Web Server / Apache

CGI Language / PHP

OS Server | GNU/Linux

DB Server / MySQL

LAMP Architecture

- **Linux - OS**
- **Apache - Web**
- **MySQL - DB**
- **PHP - Script**

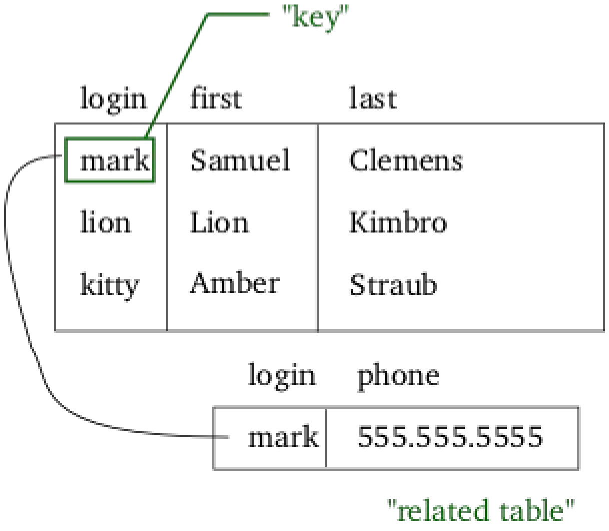
CLIENT **SERVER**

Apache server running on Linux

- Listens for connections on ports you specify
  - Where do we specify?
  - In apache2.conf file
- Makes publicly viewable content you put in the var/www/ folder of your Linux box
  - Be sure permissions are set correctly (755/644)
- We’ve already seen how to set this up, even inside a virtual container

What’s the point of a database?

- Why can’t we just serve data from a csv file?
- Because then we’d have to load the whole file into memory for every operation
  - doesn’t scale well with data volume
- Relational database management systems (RDBMS) are a popular variant
  - Relational := related records
  - Related via keys

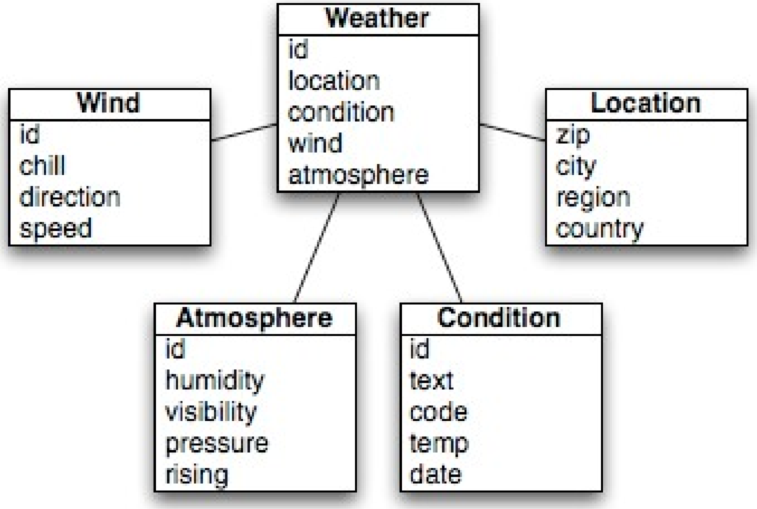


login	first	last
mark	Samuel	Clemens
lion	Lion	Kimbro
kitty	Amber	Straub

login	phone
mark	555.555.5555

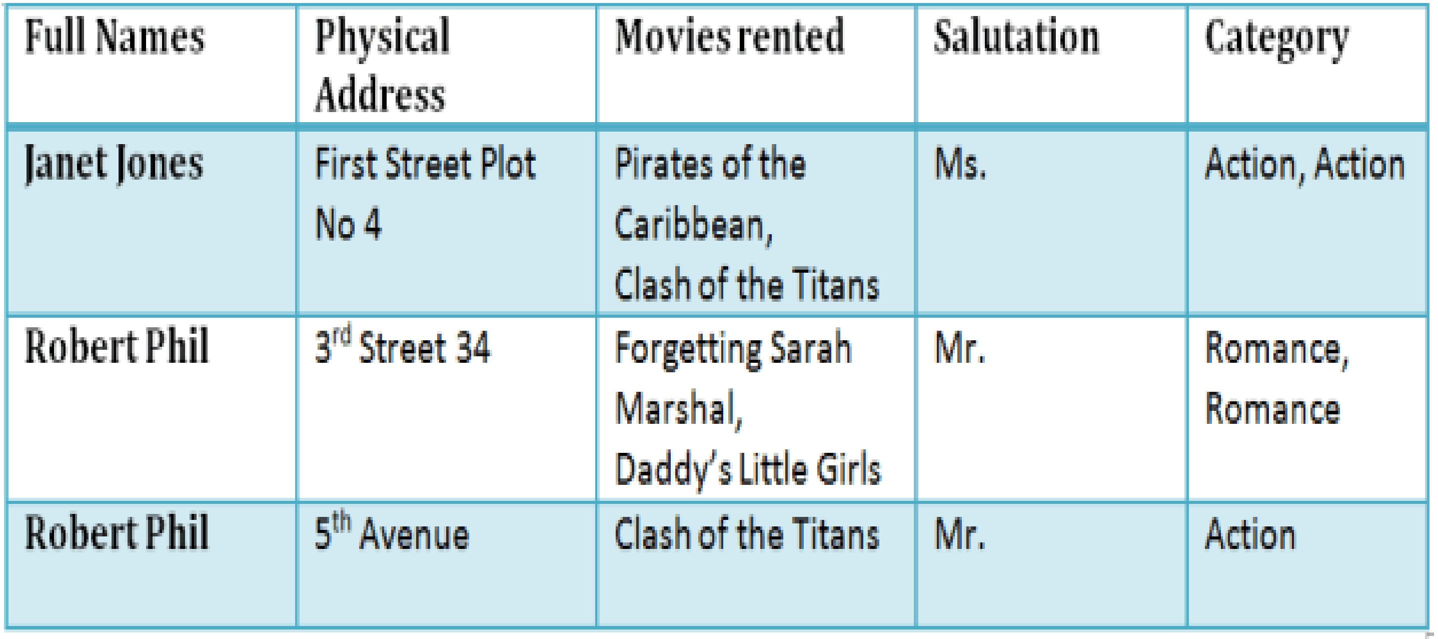
RDBMS = data modeling

- Learning to work with RDBMS
  - Connection etc. (5%)
  - CRUD operations (10% effort)
  - Data modeling (85% effort)
- What makes for an effective data model?



Data normalization

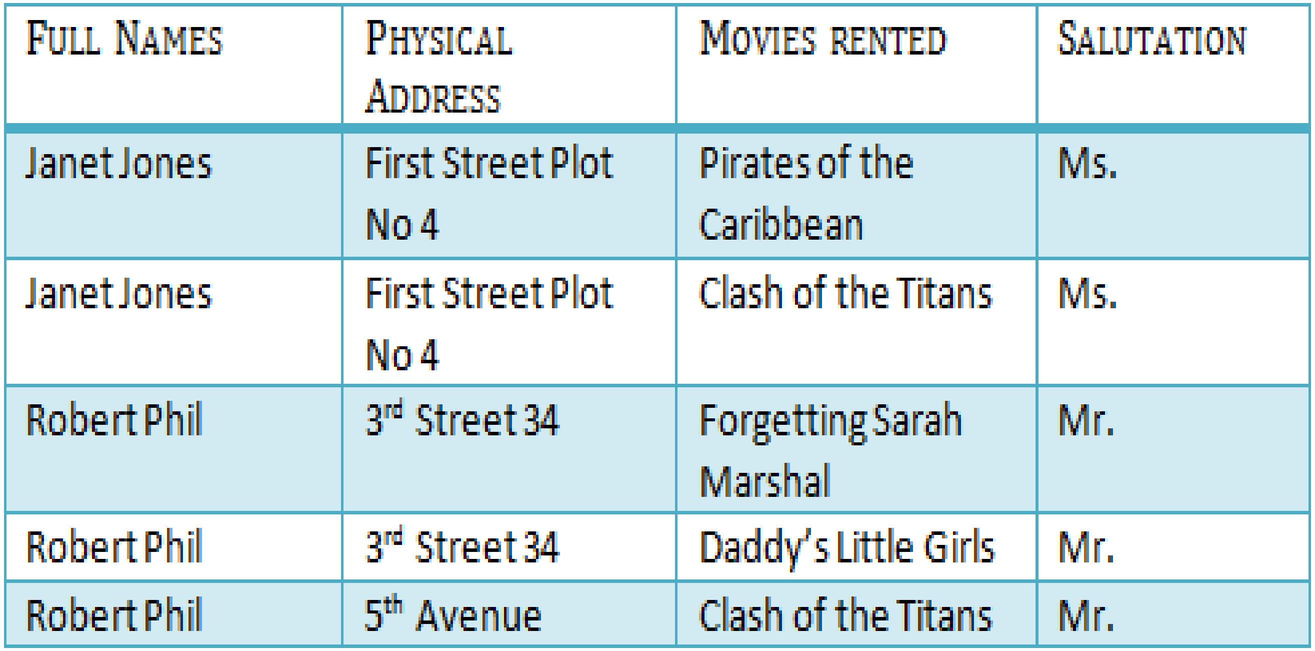
- How would you make queries on this database?



Full Names	Physical Address	Movies rented	Salutation	Category
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.	Action, Action
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.	Romance, Romance
Robert Phil	5th Avenue	Clash of the Titans	Mr.	Action

First normal form

- Every cell should have one value only
- Every record (row) should be unique

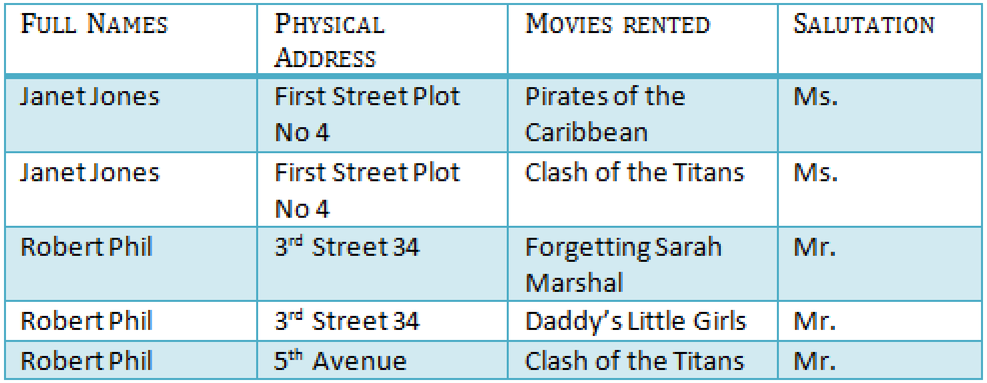


Full Names	Physical Address	Movies rented	Salutation
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

Second normal form

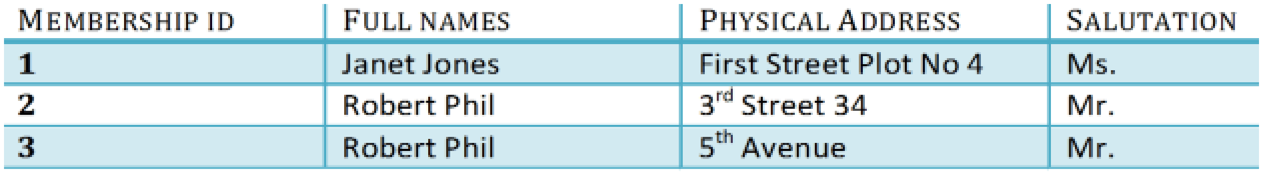
- Primary key
  - Must be unique and non-null
  - Must auto-increment with records
  - Cannot be changed
- Second normal form desiderata
  - Be in 1st NF
  - Single column primary key

Second normal form

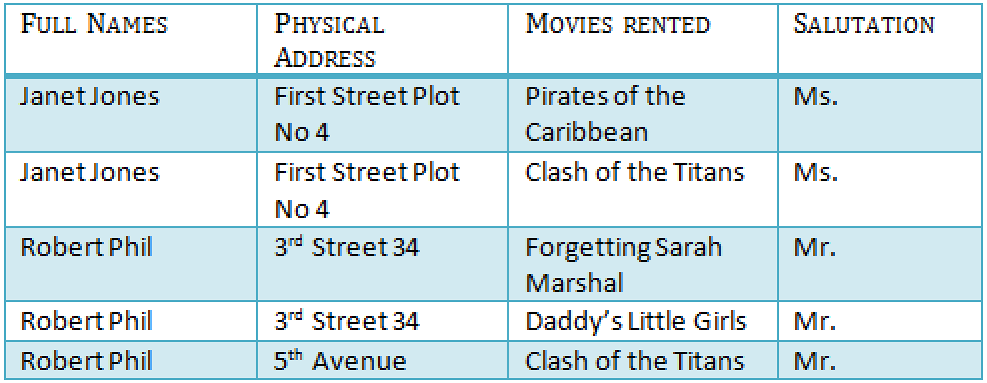


Full Names	Physical Address	Movies rented	Salutation
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

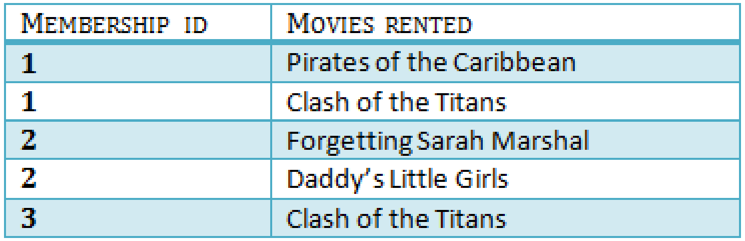
Second normal form



MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.



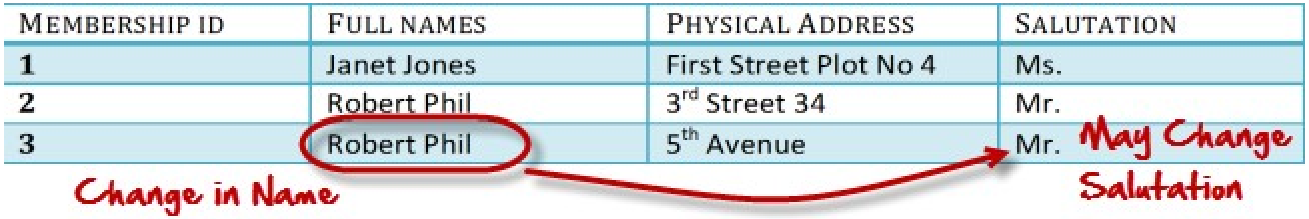
FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.



MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Third normal form

- Be in 2NF
- Remove transitive functional dependencies
- Transitive functional dependencies

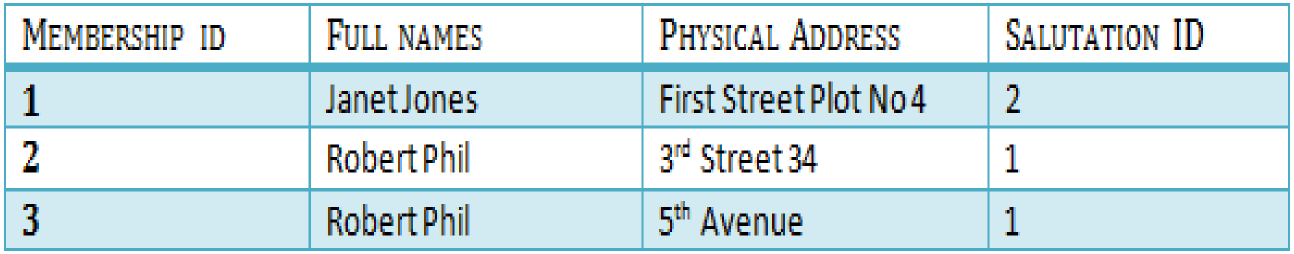


MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

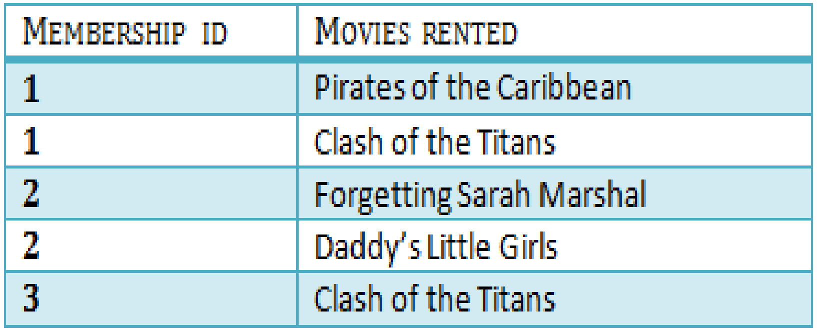
Change in Name

May Change Salutation

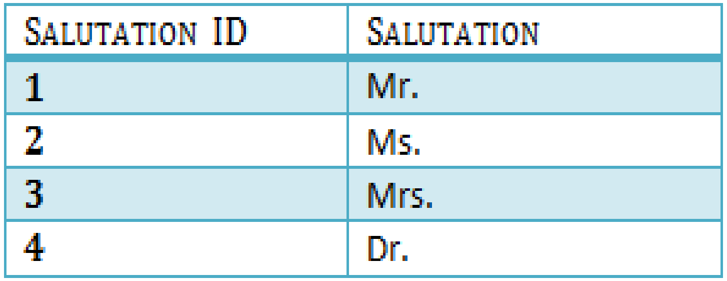
Third normal form



Membership ID	Full names	Physical Address	Salutation ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1



Membership ID	Movies rented
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans



Salutation ID	Salutation
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

Database desiderata

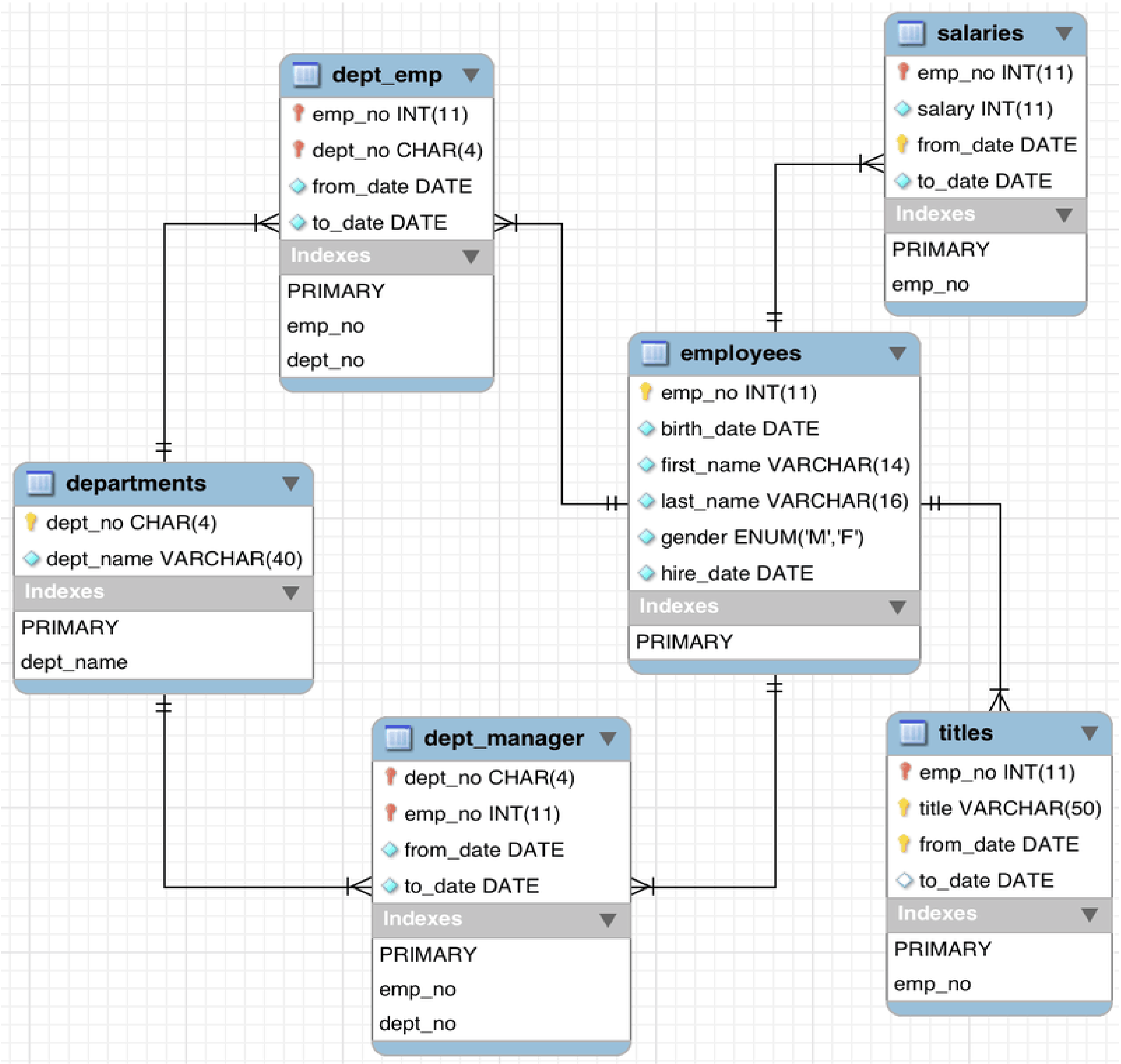
- Databases must pass the ACID test
- Atomicity = in a transaction with two or more pieces of information, either all are committed or none are
- Consistency = a transaction either creates a new valid state, or changes nothing
- Isolation = an uncommitted transaction must stay in isolation from other transactions
- Durability = committed data is saved by the system so its available even in the event of a failure

An employee management system

- An employee
  - Belongs to one of several departments
  - Has a certain salary
  - Holds a certain title
  - All of these things change over time
  - How will you model this database?

Get the actual database from https://dev.mysql.com/doc/employee/en/

Data model



dept_emp
emp_no INT(11)
dept_no CHAR(4)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no
dept_no

salaries
emp_no INT(11)
salary INT(11)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no

employees
emp_no INT(11)
birth_date DATE
first_name VARCHAR(14)
last_name VARCHAR(16)
gender ENUM('M','F')
hire_date DATE
Indexes
PRIMARY

departments
dept_no CHAR(4)
dept_name VARCHAR(40)
Indexes
PRIMARY
dept_name

dept_manager
dept_no CHAR(4)
emp_no INT(11)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no
dept_no

titles
emp_no INT(11)
title VARCHAR(50)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no

Working with databases in mySQL

- Once you've downloaded the database, install it to mysql
  - Install mySQL if you don't have it on your machine
- Practice CRUD operations from the mysql shell
  - CRUD = create, read, update, delete
  - Basic syntax: [action] [target records] FROM [table] .... [conditions/constraints];

Practicing SQL

- Typical actions = select, update, delete, insert into
  - e.g. select * from employees;
- Other actions = alter table, drop table, create index, drop index
  - e.g. alter table employees drop column last_name;
- Typical conditions = where, order by, group by
  - e.g. delete from employees where emp_no = 342 ;

Adding constraints during table creation

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    UNIQUE (ID)
);
```

Common constraints: not null, unique, primary key, foreign key, check, default, index

SQL with PHP

- A PHP script can be included anywhere within an HTML document within the tags <?php   ?>
- Have to use a .php extension for that file
- Syntax of PHP very much like C
- But it has a bunch of functions and (super) global variables that simplify server-side programming
- Most common server-side scripting language

Web form data entry with PHP

```
<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_REQUEST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>
```

PHP+SQL

- Can use PHP to
  - Connect to specific databases on your machine

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$db = “emp_db”
// Create connection
$conn = new mysqli($servername, $username, $password, $db);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

PHP+SQL

- Can use PHP to
  - Execute SQL queries on the database tables

```
$sql = "SELECT id, firstname, lastname FROM employees";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
    }
} else {
    echo "0 results";
}
```

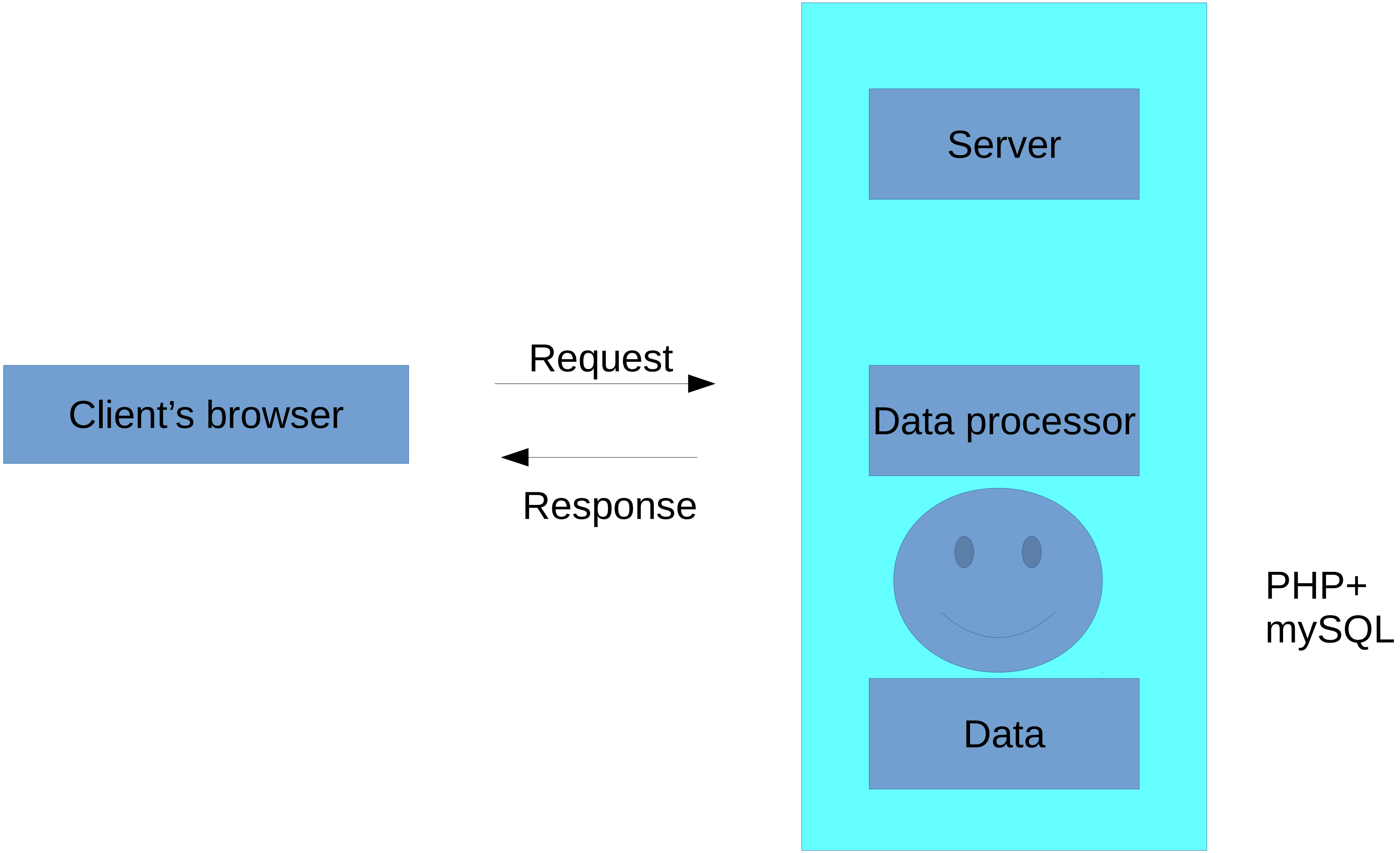
Displaying results to the web

- PHP variables can be concatenated easily with HTML strings to generate content dynamically

```
<?php
$txt1 = "SQL access ";
$txt2 = <data from SQL database>;

echo "<h2>" . $txt1 . "</h2>";
echo "Data from db" . $txt2 . "<br>";
?>
```

Architecture of a web service



In lab next week

- Learn how to connect to dbs on mysql shell and perform CRUD operations on them
- Learn how to do the same thing using PHP
- I've posted a link to a nice tutorial for both on the website
- Build a web interface for the employees database such that
  - I can query for employees by ID, by last name or by department
  - I can identify which departments are the largest, by employee count
  - I can display people within departments ordered by tenure with the company
  - I can see the gender ratio of employees in any department
  - I can see the gender pay ratio in any department
    - Gender pay ratio = the ratio of female to male salary for the same job title
- Think about a different data set you could model for developing your own web app for the coming week