Exercises

1. **Thesarus:** A web service called thesaurus.php has the following behavior:

   Word look up query: GET request
   Returns a plaintext, space delimited list of synonyms or antonyms for the requested word

Parameter Name	Value
q	The word to look up
mode	"syn" – find synonyms for this word "ant" – find antonyms for this word

   Add related word pair query: POST request
   Adds a relationship between two words, either as synonyms or antonyms

Parameter Name	Value
q1	One word of the relation
q2	Second word of the relation
relationship	"syn" – these words are synonyms "ant" – these words are antonyms

   a) Implement a search form for this thesaurus web service. The starter HTML is written below. When a user submits the form, you should stop the form submission event and make an AJAX request to thesaurus.php with the appropriate parameters to find a synonym or antonym for the requested word. Upon a successful request, you should display the definition in the paragraph element with the id of "results."

```
<form id="search_form">
   <fieldset>
      Search for: <input type="text" id="word" />
      <select id="type">
         <option>synonyms</option>
         <option>antonyms</option>
      </select>
      <input type="submit" value="Search" />
   </fieldset>
</form>
<p id="results"></p>
```

b) Implement a way to add synonyms or antonyms to the service. The starter HTML is below. Similarly to part a, you should stop the form submission event and make an AJAX request to thesaurus.php with the appropriate parameters to add the synonym or antonym pair. Upon a successful request, you should display "<word 1> and <word 2> added" in the paragraph element with the id of "results."

```
<form id="search_form">
   <fieldset>
      <input type="text" id="word1" /> and <input type="text" id="word2" />
      are
      <select id="type">
         <option>synonyms</option>
         <option>antonyms</option>
      </select>
      <input type="submit" value="Add Relationship" />
   </fieldset>
</form>
<p id="results"></p>
```

2. **World Database:** There is a world database with the following tables:

   Countries(code, name, continent, surface_area, population, life_expectancy, gnp, …)
   Cities(id, name, country_code, district, population)
   CountriesLanguages(country_code, language, official, percentage)

   a) Write a MySQL query that will grab all the districts of Japan with more than 2,500,000 people residing in it. You should use the Cities table in the world database for this problem. Your end result should list three districts:
      - Tokyo-to
      - Kanagawa
      - Osaka

      Remember that Japan's country code is JPN.

   b) Create a MySQL query that will grab the top 5 most populous English-speaking nations. You may need to join some or all of the three tables provided: Cities, Countries, and CountriesLanguages. The result of the query should be:
      - United States
      - Japan
      - United Kingdom
      - South Africa
      - Canada

      Remember that these results are listed in descending order, with the United States being the first and Canada being the fifth most populous English-speaking country.

   c)

3. **Simpsons Database:** There is a database for Springfield Elementary School with the following tables:

   Courses(id, name, teacher_id)

Grades(student_id, course_id, grade)
Students(id, name, email, password)
Teachers(id, name)

a) Write a SQL query to return the all the courses offered listed with the name of the teacher who teaches the course.

b) Write a SQL query to return the names of all teachers who have taught a course where at least 2 students received a B- or better in the course. This query should return the following results:
   - Krabappel
   - Hoover

   Remember that the value of an A is *less* than a B- because A is before B alphabetically, even though A is logically a "greater" grade than B-.

c)

4. **IMDB Database:** Our IMDB database has the following tables:

actors(id, first_name, last_name, gender)
directors(id, first_name, last_name)
directors_genres(director_id, genre, prob)
movies(id, name, year, rank)
movies_directors(director_id, movie_id)
roles(actor_id, movie_id, role)
movies_genres(movie_id, genre)

a) Write a query that lists the names of themovies whose title contains the word "space" in descending order by rank.

b) Write a query that lists the female actors who appeared in a movie during the 90s (1990-1999) that was rated higher than 8.5.

c) Write a query that lists all actors who was in a movie rated lower than 3.0 two or more times. List the name of the actor, the movie and each rating, ordered ascending by the actors' last name then first name.

d) Write a query that lists all actors who have been in two or more movies of different genres. List their name, movie and their respective genres.