Exercise/Task	Checklist
Exercise 1 — Tasks Table: Movies Id Title Director Year Length_minutes 1 Toy Story John Lasseter 1995 81 2 A Bug's Life John Lasseter 1998 95 3 Toy Story 2 John Lasseter 1999 93 4 Monsters, Inc. Pete Docter 2001 92 5 Finding Nemo Andrew Stanton 2003 107 6 The Incredibles Brad Bird 2004 116 7 Cars John Lasseter 2006 117 8 Ratatouille Brad Bird 2007 115 9 WALL-E Andrew Stanton 2008 104 10 Up Pete Docter 2009 101	

Task	Syntax	Result
Find the **title** of each film ✓		Table: Movies Id Title Director Year Length_minutes 1 Toy Story John Lasseter 1995 81 2 A Bug's Life John Lasseter 1998 95 3 Toy Story 2 John Lasseter 1999 93 4 Monsters, Inc. Pete Docter 2001 92 5 Finding Nemo Andrew Stanton 2003 107 6 The Incredibles Brad Bird 2004 116 7 Cars John Lasseter 2006 117 8 Ratatouille Brad Bird 2007 115 9 WALL-E Andrew Stanton 2008 104 10 Up Pete Docter 2009 101
Find the **director** of each film✓		
Find the **title** and **director**of each film ✓		
Find the **title** and **year** of each film ✓		
Find **all** the information about each film		

Exercise 2 — Tasks

Task	Syntax	Result

	Find the movie with a row **id** of 6 ✓		Table: Movies Title | Year Toy Story | 1995 A Bug's Life | 1998 Toy Story 2 | 1999 Monsters, Inc. | 2001 Finding Nemo | 2003 `SELECT title, year FROM movies where id <= 5;`
	Find the movies released in the **year**s between 2000 and 2010 ✓		
	Find the movies **not** released in the **year**s between 2000 and 2010 ✓		
	Find the first 5 Pixar movies and their release **year** ✓		
Exercise 3 — Tasks			

Task	Syntax	Result
Find all the Toy Story movies ✓		Table: Movies Id | Title | Director | Year | Length_minutes 1 | Toy Story | John Lasseter | 1995 | 81 2 | A Bug's Life | John Lasseter | 1998 | 95 3 | Toy Story 2 | John Lasseter | 1999 | 93 4 | Monsters, Inc. | Pete Docter | 2001 | 92 5 | Finding Nemo | Andrew Stanton | 2003 | 107 6 | The Incredibles | Brad Bird | 2004 | 116 7 | Cars | John Lasseter | 2006 | 117 8 | Ratatouille | Brad Bird | 2007 | 115 9 | WALL-E | Andrew Stanton | 2008 | 104 10 | Up | Pete Docter | 2009 | 101 `SELECT * FROM movies WHERE title NOT LIKE "John Lasseter";`

	Find all the movies directed by John Lasseter ✓		
	Find all the movies (and director) not directed by John Lasseter ✓		
	Find all the WALL-* movies ✓		

Exercise 4 — Tasks

Task	Syntax	Result
List all directors of Pixar movies (alphabetically), without duplicates ✓	SELECT DISTINCT director FROM movies ORDER BY director ASC;	
List the last four Pixar movies released (ordered from most recent to least) ✓	SELECT title FROM movies ORDER BY year DESC LIMIT 4 ;	Table: Movies Title Monsters University Brave Cars 2 Toy Story 3 SELECT title FROM movies ORDER BY year DESC LIMIT 4 ;

<table>
<tr>
<td></td>
<td>
<table>
<tr>
<td>List the first five Pixar movies sorted alphabetically ✓</td>
<td>SELECT title FROM movies
ORDER BY title ASC limit 5;</td>
<td>Table: Movies
Title
A Bug's Life
Brave
Cars
Cars 2
Finding Nemo
 
</td>
</tr>
<tr>
<td></td>
<td>SELECT title FROM movies
ORDER BY title ASC limit 5
OFFSET 5;</td>
<td>Table: Movies
Title
Monsters University
Monsters, Inc.
Ratatouille
The Incredibles
Toy Story
 
</td>
</tr>
</table>
</td>
</tr>
<tr>
<td>Exercise 4</td>
<td>
<table>
<tr><th>Task</th><th>Syntax</th><th>Result</th></tr>
<tr><td>List all directors of Pixar movies (alphabetically), without duplicates ✓</td><td></td><td></td></tr>
<tr><td>List the last four Pixar movies released (ordered from most recent to least) ✓</td><td></td><td></td></tr>
<tr><td>List the first five Pixar movies sorted alphabetically ✓</td><td></td><td></td></tr>
</table>
</td>
</tr>
</table>

	List the **next** five Pixar movies sorted alphabetically ✓

Exercise 5

SELECT query
SELECT column, another_column, … FROM mytable WHERE *condition(s)* ORDER BY column ASC/DESC LIMIT num_limit OFFSET num_offset;

Table: North_american_cities

City	Country	Population	Latitude	Longitude
Guadalajara	Mexico	1500800	20.659699	-103.349609
Toronto	Canada	2795060	43.653226	-79.383184
Houston	United States	2195914	29.760427	-95.369803
New York	United States	8405837	40.712784	-74.005941
Philadelphia	United States	1553165	39.952584	-75.165222
Havana	Cuba	2106146	23.05407	-82.345189
Mexico City	Mexico	8555500	19.432608	-99.133208

Review 1 — Tasks

1. List all the Canadian cities and their populations ✓
2. Order all the cities in the United States by their latitude from n south ✓
3. List all the cities west of Chicago, ordered from west to east ✓
4. List the two largest cities in Mexico (by population) ✓
5. List the third and fourth largest cities (by population) in the Un and their population ✓

Stuck? Read this task's Solution.
Solve all tasks to continue to the next lesson.

Continue ›

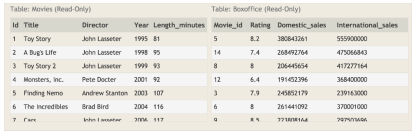
Task	Syntax	Result
1. List all the Canadian cities and their populations ✓	SELECT * FROM north_american_cities;	Table: North_american_cities City: Toronto, Population: 2795060 City: Montreal, Population: 1717767
1. Order all the cities in the United States by their latitude from north to south ✓	SELECT * FROM north_american_cities WHERE Country = "United States" ORDER BY latitude DESC;	Table: North_american_cities Chicago, United States, 2718782, 41.878114, -87.629798 New York, United States, 8405837, 40.712784, -74.005941 Philadelphia, United States, 1553165, 39.952584, -75.165222 Los Angeles, United States, 3884307, 34.052234, -118.243685 Phoenix, United States, 1513367, 33.448377, -112.074037 Houston, United States, 2195914, 29.760427, -95.369803
1. List all the cities west of Chicago, ordered from west to east ✓	SELECT * FROM north_american_cities WHERE Longitude <= "-87.629798" ORDER BY longitude ASC;	Table: North_american_cities Los Angeles, United States, 3884307, 34.052234, -118.243685 Phoenix, United States, 1513367, 33.448377, -112.074037 Guadalajara, Mexico, 1500800, 20.659699, -103.349609 Mexico City, Mexico, 8555500, 19.432608, -99.133208 Ecatepec de Morelos, Mexico, 1742000, 19.601841, -99.050674 Houston, United States, 2195914, 29.760427, -95.369803 Chicago, United States, 2718782, 41.878114, -87.629798

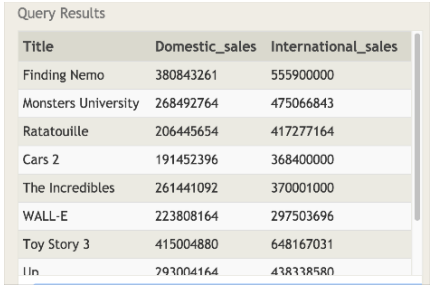
	1. List the two largest cities in Mexico (by population) ✓	SELECT city, population FROM north_american_cities WHERE Country = "Mexico" ORDER BY population DESC LIMIT 2;	Table: North_american_cities City: Mexico City, Ecatepec de Morelos Population: 8555500, 1742000
	1. List the third and fourth largest cities (by population) in the United States and their population ✓	SELECT city, population FROM north_american_cities WHERE Country = "United States" ORDER BY population DESC LIMIT 2 OFFSET 2;	Table: North_american_cities City: Chicago, Houston Population: 2718782, 2195914
Exercise 6 SELECT column, another_table_column, … FROM mytable **INNER JOIN another_table** **ON mytable.id = another_table.id** WHERE *condition(s)* ORDER BY column, … ASC/DESC			

Task	Syntax	Result

LIMIT num_limit OFFSET num_offset;

PsuedoCode: SELECT column and another_table_column FROM mytable INNER JOIN another_table ON (matching) mytable.id to another_table.id...





Query Results

Title	Domestic_sales	International_sales
Finding Nemo	380843261	555900000
Monsters University	268492764	475066843
Ratatouille	206445654	417277164
Cars 2	191452396	368400000
The Incredibles	261441092	370001000
WALL-E	223808164	297503696
Toy Story 3	415004880	648167031
Up	293004164	438338580

1. Find the domestic and international sales for each movie ✓

```
SELECT Title, Domestic_sales, International_Sales
FROM Movies
INNER JOIN Boxoffice
ON Movies.id = Boxoffice.movie_id;
```

Query Results

Title	Domestic_sales	International_sales
Finding Nemo	380843261	555900000
Monsters University	268492764	475066843
Ratatouille	206445654	417277164
Cars 2	191452396	368400000
Toy Story 2	245852179	239163000
The Incredibles	261441092	370001000
WALL-E	223808164	297503696
Toy Story 3	415004880	648167031
Toy Story	191796233	170162503
Cars	244082982	217900167

1. Show the sales numbers for each movie that did better internationally rather than domestically ✓

```
SELECT Title, Domestic_Sales, International_Sales
FROM Movies
INNER JOIN Boxoffice
   ON Movies.id = Boxoffice.Movie_id
WHERE  International_Sales > Domestic_Sales;
```

Query Results

Title	Domestic_sales	International_sales
Finding Nemo	380843261	555900000
Monsters University	268492764	475066843
Ratatouille	206445654	417277164
Cars 2	191452396	368400000
The Incredibles	261441092	370001000
WALL-E	223808164	297503696
Toy Story 3	415004880	648167031
Up	293004164	438338580

1. List all the movies by their ratings in descending order ✓

```
SELECT Title, Rating
FROM Movies
INNER JOIN Boxoffice
   ON Movies.id = Boxoffice.Movie_id
ORDER BY Rating DESC;
```

Query Results

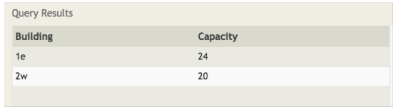
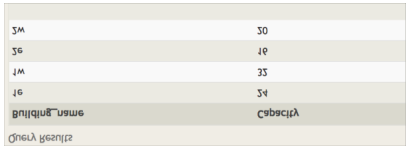
Title	Rating
WALL-E	8.5
Toy Story 3	8.4
Toy Story	8.3
Up	8.3
Finding Nemo	8.2
Monsters, Inc.	8.1
Ratatouille	8

Task	Syntax	Result

Exercise 7 - LEFT/RI

Select query with LEFT/RIGHT/FULL JOINs on multiple tables

SELECT column, another_column, …
FROM mytable
INNER/LEFT/RIGHT/FULL JOIN another_table
ON mytable.id = another_table.matching_id
WHERE *condition(s)*
ORDER BY column, … ASC/DESC
LIMIT num_limit OFFSET num_offset;

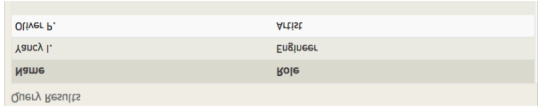
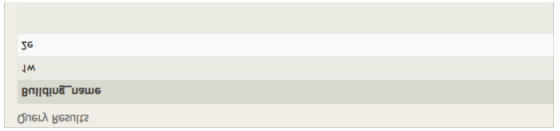
Task	Syntax	Result
1. Find the list of all buildings that have employees ✓	SELECT DISTINCT Building, Capacity FROM Employees LEFT JOIN Buildings ON Employees.Building = Buildings.Building_name;	
Find the list of all buildings and their capacity ✓	SELECT Building_Name, Capacity FROM Buildings; OR SELECT * FROM buildings;	
List all buildings and the distinct employee roles in	SELECT DISTINCT building_name, role FROM buildings	

	each building (including empty buildings) ✓	LEFT JOIN employees ON building_name = building; ~~SELECT Buildings.Building_name, DISTINCT Employees.Role FROM Buildings LEFT JOIN Employees ON Buildings.Building_name = Employees.Building;~~	

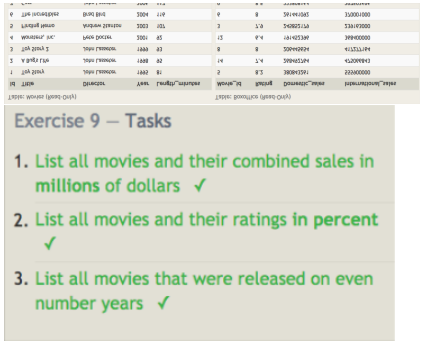
Exercise 8

Select query with constraints on NULL values

SELECT column, another_column, … FROM mytable **WHERE column IS/IS NOT NULL** AND/OR *another_condition* AND/OR …;

Task	Syntax	Result
1. Find the name and role of all employees who have **not** been assigned to a building ✓	SELECT Name, Role FROM employees WHERE building IS NULL;	
1. Find the names of the buildings that hold no employees ✓	SELECT Building_name FROM Buildings LEFT JOIN Employees ON Building_name = Building WHERE Building IS NULL;	

Exercise 9



	Syntax	Result
9-1	SELECT Title, (domestic_sales + International_sales)/1000000 AS Combined_Sales FROM movies INNER JOIN Boxoffice ON Movies.id = Boxoffice.movie_id;	
9-2	SELECT Title, ROUND (Rating * 10) AS "Rating %" FROM movies INNER JOIN Boxoffice ON Movies.id = Boxoffice.movie_id;	
9-3	SELECT Title, Year FROM movies INNER JOIN Boxoffice ON Movies.id = Boxoffice.movie_id WHERE (year % 2) = 0;	

Exercise 10 - Aggregate Functions

Select query with aggregate functions over all rows

Question	Syntax	Result
10-1	SELECT MAX(years_employed) as Max_years_employed	

SELECT
AGG_FUNC(*column_or_expression*)
AS aggregate_description, …
FROM mytable
WHERE *constraint_expression*;

Exercise 10 – Tasks

1. Find the longest time that an employee has been at the studio ✓
2. For each role, find the average number of years employed by employees in that role ✓
3. Find the total number of employee years worked in each building ✓

	FROM employees; ~~SELECT *~~ ~~FROM Employees~~ ~~WHERE~~ ~~MAX(years_employed)~~ ~~GROUP BY~~ ~~years_employed;~~	
10-2	SELECT Role, AVG(Years_employed) FROM employees GROUP BY role;	
10-3	SELECT building, SUM(years_employed) FROM employees GROUP BY building;	

Exercise 11 – Aggregates (Pt. 2)

Exercise 11 – Tasks

1. Find the number of Artists in the studio (without a **HAVING** clause) ✓
2. Find the number of Employees of each role in the studio ✓
3. Find the total number of years employed by all Engineers ✓

	Syntax	Result
11-1	SELECT COUNT(Role) FROM employees WHERE Role = "Artist";	

	11-2	SELECT role, count(role) AS Count FROM employees GROUP BY role;	
	11-3	SELECT role, sum(Years_employed) AS Years FROM employees WHERE role = "Engineer";	

Exercise 12

Exercise	Syntax	Result
12-1	SELECT COUNT(title), director FROM movies Group BY director;	
12-2	Correct Answer: SELECT director, SUM(domestic_sales + international_sales) as Cumulative_sales_from_all_movies FROM movies INNER JOIN boxoffice ON movies.id = boxoffice.movie_id GROUP BY director; _____	Query Results Director | Cumulative_sales_from_all_movies Andrew Stanton | 1458055121 Brad Bird | 1255164910 Brenda Chapman | 538983207 Dan Scanlon | 743559607 John Lasseter | 2232208025 Lee Unkrich | 1063171911 Pete Docter | 1294159000

My Answers

```
SELECT director,
sum(domestic_sales) AS Domestic,
sum(international_sales) AS
International
FROM movies
INNER JOIN Boxoffice
   ON Movies.id =
Boxoffice.movie_id
GROUP BY Director;
```

OR

```
SELECT Director, (domestic_sales +
International_sales) AS
Combined_Sales
FROM movies
INNER JOIN Boxoffice
   ON Movies.id =
Boxoffice.movie_id
GROUP BY director;
```

CORRECTED ANSWER 2

```
SELECT Director,
SUM(domestic_sales +
International_sales) AS
Combined_Sales
FROM movies
INNER JOIN Boxoffice
   ON Movies.id =
Boxoffice.movie_id
GROUP BY director;
```

_____

My Answers

Query Results

Director	Domestic	International
Andrew Stanton	604651425	853403696
Brad Bird	467886746	787278164
Brenda Chapman	237283207	301700000
Dan Scanlon	268492764	475066843
John Lasseter	1035982355	1196225670
Lee Unkrich	415004880	648167031
Pete Docter	582920420	711238580

OR

Exercise 13 Exercise 13 – Tasks 1. Add the studio's new production, **Toy Story 4** to the list of movies (you can use any director) ✓ 2. Toy Story 4 has been released to critical acclaim! It had a rating of **8.7**, and made **340 million domestically** and **270 million internationally**. Add the record to the **`BoxOffice`** table. ✓	

Exercise	Syntax	Result
13-1	INSERT INTO Movies (title, director) VALUES ("Toy Story 4", "John Lasseter");	Query Results Id | Title | Director | Year | Length_minutes 1 | Toy Story | John Lasseter | 1995 | 81 2 | A Bug's Life | John Lasseter | 1998 | 95 3 | Toy Story 2 | John Lasseter | 1999 | 93 15 | Toy Story 4 | John Lasseter 16 | Toy Story 4 | John Lasseter
13-2	INSERT INTO boxoffice VALUES (4, 8.7, 340000000, 270000000); _____ **My Answer** INSERT INTO Boxoffice (rating, Domestic_sales, International_sales) VALUES (8.7, 340000000,270000000) INNER JOIN Movies	

		ON movies.id = boxoffice.movie_id WHERE Movies.title = "Toy Story 4"; **Corrected** INSERT INTO Boxoffice ~~(rating, Domestic_sales, International_sales)~~ VALUES (8.7, 340000000,270000000) ~~INNER JOIN Movies ON movies.id = boxoffice.movie_id~~ ~~WHERE Movies.title = "Toy Story 4"~~;	

Exercise 14 - UPDATE			
		Syntax	Result
	14-1	UPDATE Movies SET director = "John Lasseter"	

		WHERE Title = "A Bug's Life";	
	14-2	UPDATE Movies SET Year = "1999" WHERE Title = "Toy Story 2";	
	14-3	UPDATE Movies SET Title = "Toy Story 3", Director = "Lee Unkrich" WHERE Title = "Toy Story 3";	
Exercise 15- DELETE Delete statement with condition DELETE FROM mytable WHERE condition;			
		Syntax	Result
	15-1	DELETE FROM movies WHERE Year < 2005;	
	15-2	DELETE FROM movies WHERE Director = "Andrew Stanton";	

Exercise 16 – CREATE TABLE

Create table statement w/ optional table constraint and default value
CREATE TABLE IF NOT EXISTS mytable (column *DataType TableConstraint* DEFAULT *default_value*, another_column *DataType TableConstraint* DEFAULT *default_value*, …);

Exercise 16 – Tasks

1. Create a new table named **`Database`** with the following columns:
   - **`Name`** A string (text) describing the name of the database
   - **`Version`** A number (floating point) of the latest version of this database
   - **`Download_count`** An integer count of the number of times this database was downloaded

Stuck? Read this task's Solution.

This table has no constraints. ✓

Solve all tasks to continue to the next lesson.

Exercise	Syntax	Result
16-1	CREATE TABLE database (Name TEXT, Version FLOAT, Download_count)	

Exercise 17 - Altering Tables - Adding Columns

Altering table to add new column(s)
ALTER TABLE mytable ADD column *DataType OptionalTableConstraint* DEFAULT default_value;

Altering table to remove column(s)
ALTER TABLE mytable DROP column_to_be_deleted;

Altering table name
ALTER TABLE mytable RENAME TO new_table_name;

	Syntax	Result
17-1	ALTER TABLE movies ADD Aspect_ratio FLOAT;	
	ALTER TABLE movies ADD Language TEXT DEFAULT "English";	

Exercise 18 – Dropping Tables

Drop table statement
DROP TABLE IF EXISTS mytable;

	Syntax	Result
18-1	DROP TABLE IF EXISTS Movies	
18-2	DROP TABLE IF EXISTS BoxOffice;	

Hacker Rank Normalization 1

The following unnormalized table named **PRODUCT** is transformed to first normal form (**1NF**) by splitting it into two tables which have **X** and **Y** rows (such that **X** <**Y**) respectively. Both the tables have **Z** columns.

```
*Product-ID*    *Colors*    *Price*
1               Red,Green   15.0
2               Blue        18.0
3               Yellow,Pink 2.5
```

What are the values of **X**, **Y**, **Z**? Enter these integers, each on a new line, in the text-box below. Do not leave any leading or trailing spaces.

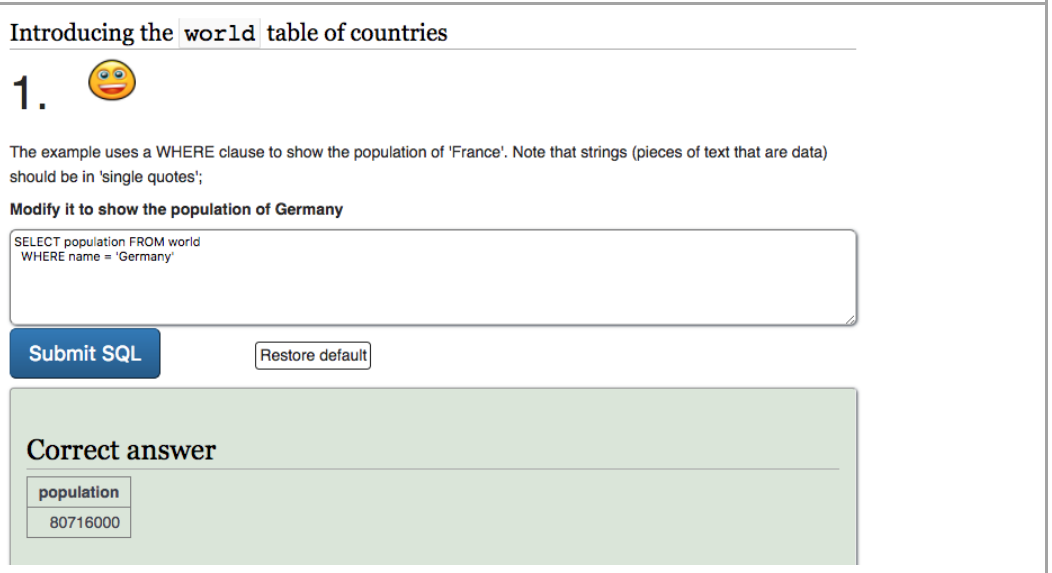
Current Buffer (saved locally, editable)

```
1  *Product-ID* *Colors*
2  1               Red
3  1               Green
4  2               Blue
5  3               Yellow
6  3               Pink
7
8  *Product-ID* *Price*
9  1               15.0
10 1               15.0
11 2               18.0
12 3               2.5
13 3               2.5
```

Hacker Rank Normalization 2

	Syntax	Result

SQLZOO 0-1 SELECT	

Exercise	Syntax	Result
SELECT Basics 1		

2

Scandinavia

2.

Checking a list The word **IN** allows us to check if an item is in a list. The example shows the name and population for the countries 'Brazil', 'Russia', 'India' and 'China'.

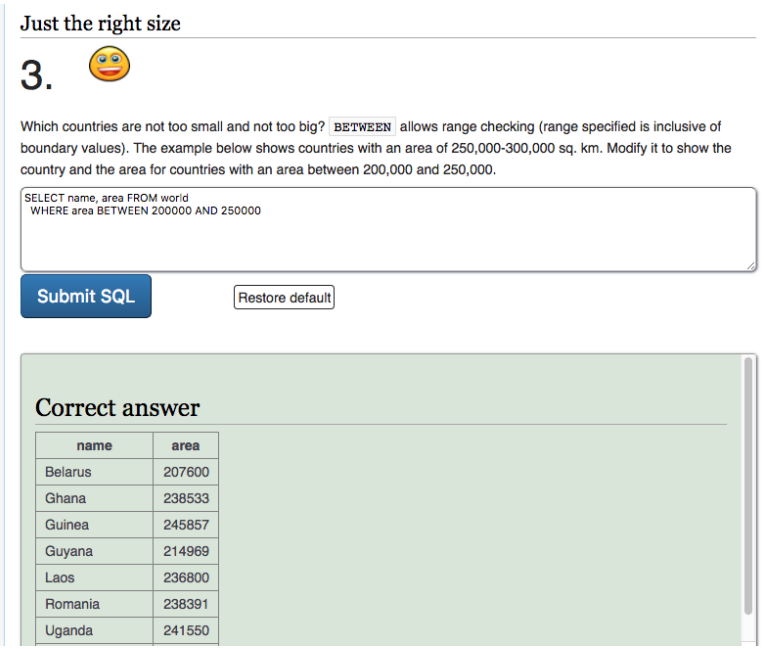
Show the name and the population for 'Sweden', 'Norway' and 'Denmark'.

```
SELECT name, population FROM world
  WHERE name IN ('Sweden', 'Norway', 'Denmark');
```

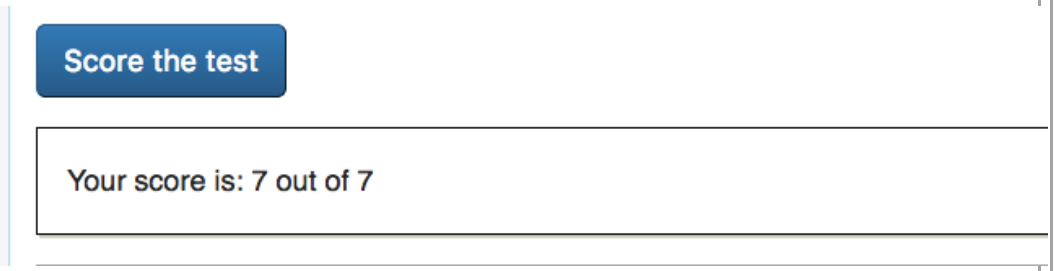
Submit SQL Restore default

Correct answer

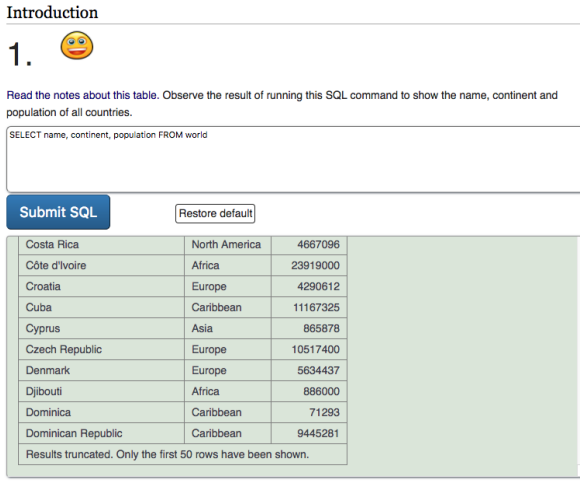
name	population
Denmark	5634437
Norway	5124383
Sweden	9675885


Just the right size
3.
Which countries are not too small and not too big? BETWEEN allows range checking (range specified is inclusive of boundary values). The example below shows countries with an area of 250,000-300,000 sq. km. Modify it to show the country and the area for countries with an area between 200,000 and 250,000.
SELECT name, area FROM world
WHERE area BETWEEN 200000 AND 250000
Submit SQL
Restore default
Correct answer
name area
Belarus 207600
Ghana 238533
Guinea 245857
Guyana 214969
Laos 236800
Romania 238391
Uganda 241550

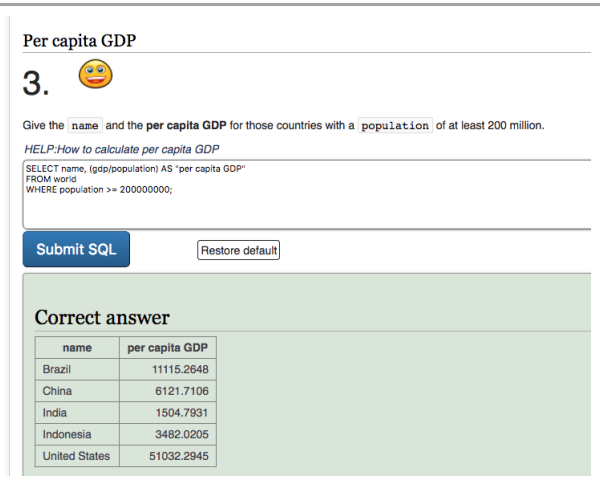
SELECT QUIZ


Score the test
Your score is: 7 out of 7

SELECT from WORLD Tutorial

Large Countries	Syntax	Result
1		
2		

3



Per capita GDP

3.

Give the `name` and the **per capita GDP** for those countries with a `population` of at least 200 million.

HELP:How to calculate per capita GDP

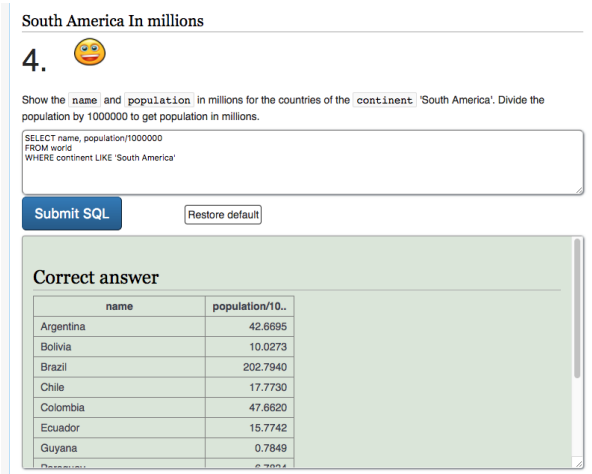
```
SELECT name, (gdp/population) AS "per capita GDP"
FROM world
WHERE population >= 200000000;
```

Submit SQL Restore default

Correct answer

name	per capita GDP
Brazil	11115.2648
China	6121.7106
India	1504.7931
Indonesia	3482.0205
United States	51032.2945

4



South America In millions

4.

Show the `name` and `population` in millions for the countries of the `continent` 'South America'. Divide the population by 1000000 to get population in millions.

```
SELECT name, population/1000000
FROM world
WHERE continent LIKE 'South America'
```

Submit SQL Restore default

Correct answer

name	population/10..
Argentina	42.6695
Bolivia	10.0273
Brazil	202.7940
Chile	17.7730
Colombia	47.6620
Ecuador	15.7742
Guyana	0.7849

	5		France, Germany, Italy 5. Show the `name` and `population` for France, Germany, Italy SELECT name, population FROM world WHERE name LIKE 'France' OR name LIKE 'Germany' OR name LIKE 'Italy'; Submit SQL Restore default Correct answer name | population France | 65906000 Germany | 80716000 Italy | 60782668
	BBC Quiz		Score the test Your score is: 7 out of 7