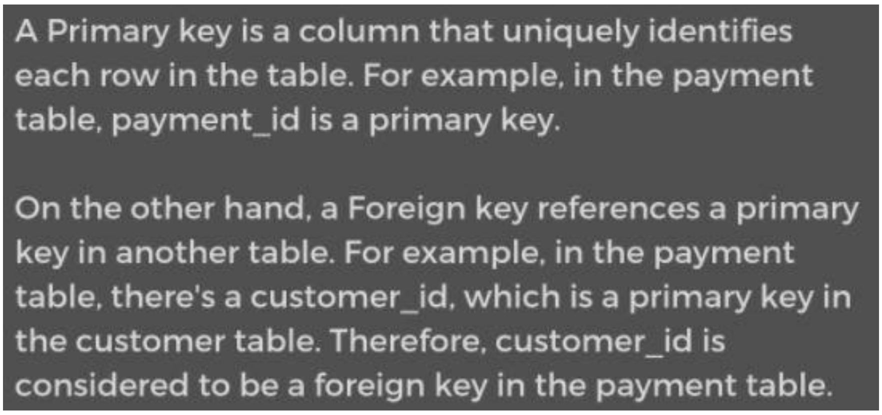
# <<<<<<SQL ORAL INTERVIEW QUESTIONS>>>>>>

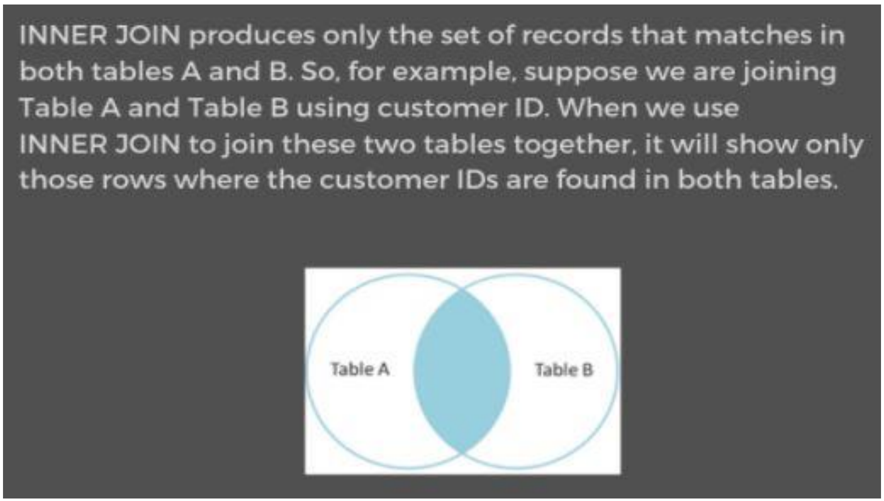
1. **What is a Primary Key and a Foreign key? You may illustrate using an example?**



A Primary key is a column that uniquely identifies each row in the table. For example, in the payment table, payment_id is a primary key.

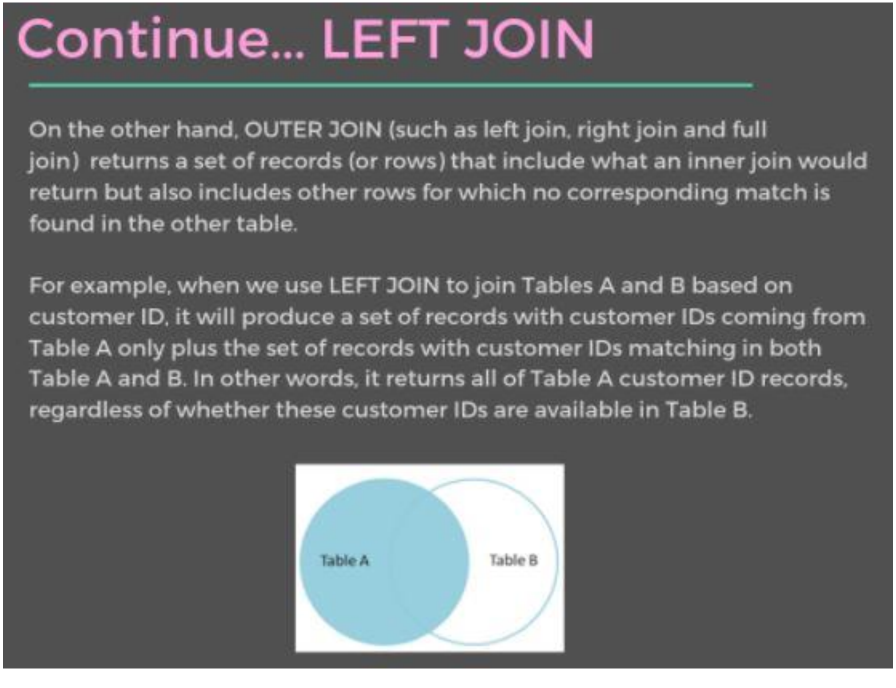
On the other hand, a Foreign key references a primary key in another table. For example, in the payment table, there's a customer_id, which is a primary key in the customer table. Therefore, customer_id is considered to be a foreign key in the payment table.

2. **Tell me in non-technical terms the difference between an inner join & outer join (for example, left or right join)?**



INNER JOIN produces only the set of records that matches in both tables A and B. So, for example, suppose we are joining Table A and Table B using customer ID. When we use INNER JOIN to join these two tables together, it will show only those rows where the customer IDs are found in both tables.

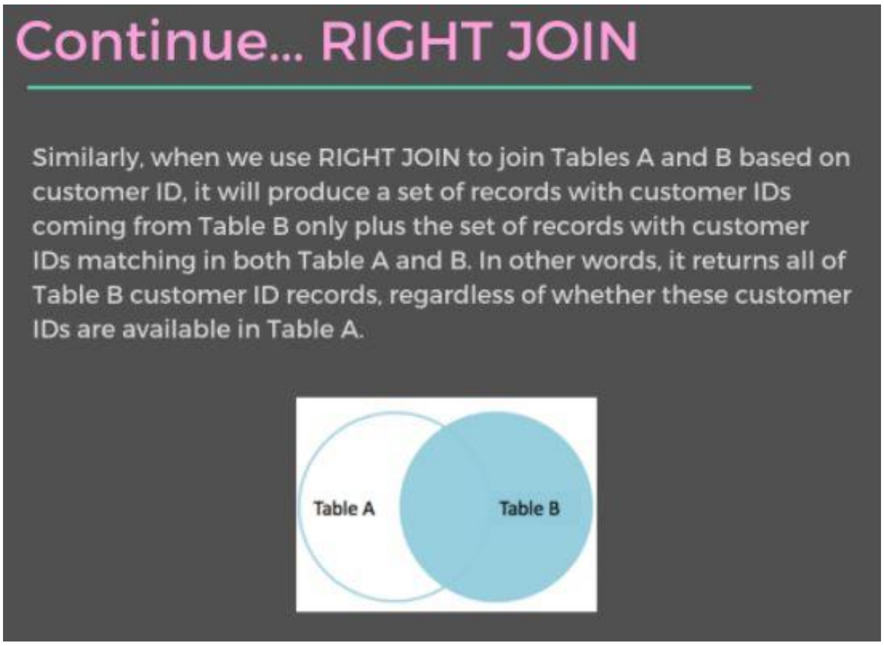
Table A
Table B



## Continue... LEFT JOIN

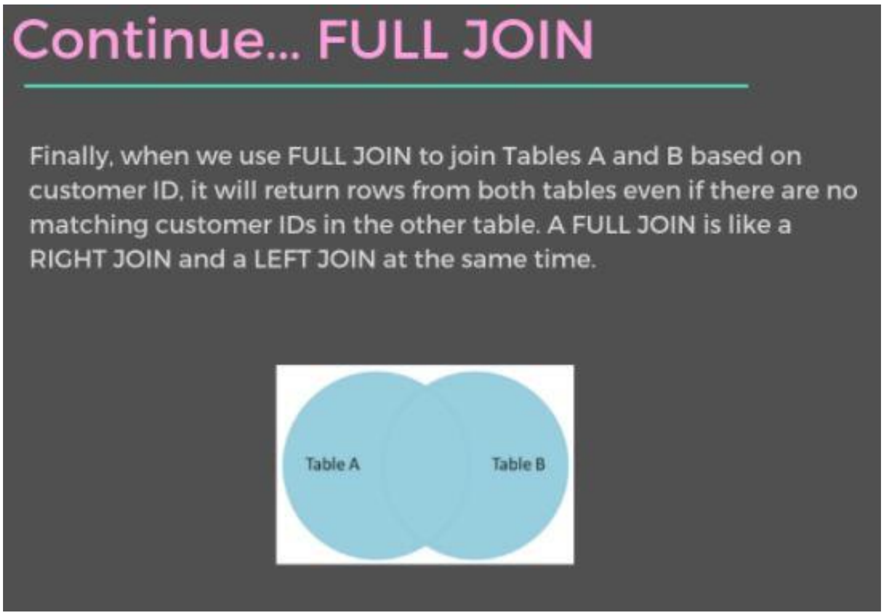
On the other hand, OUTER JOIN (such as left join, right join and full join) returns a set of records (or rows) that include what an inner join would return but also includes other rows for which no corresponding match is found in the other table.

For example, when we use LEFT JOIN to join Tables A and B based on customer ID, it will produce a set of records with customer IDs coming from Table A only plus the set of records with customer IDs matching in both Table A and B. In other words, it returns all of Table A customer ID records, regardless of whether these customer IDs are available in Table B.



## Continue... RIGHT JOIN

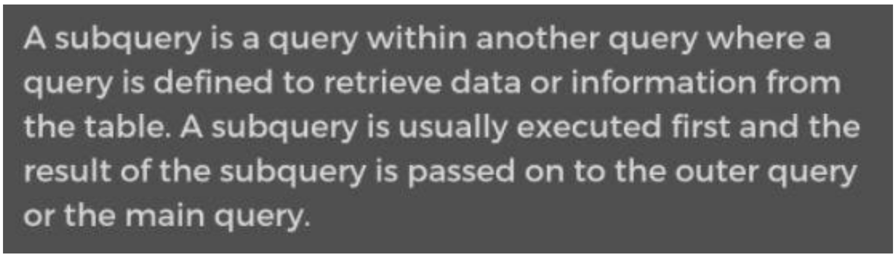
Similarly, when we use RIGHT JOIN to join Tables A and B based on customer ID, it will produce a set of records with customer IDs coming from Table B only plus the set of records with customer IDs matching in both Table A and B. In other words, it returns all of Table B customer ID records, regardless of whether these customer IDs are available in Table A.



## Continue... FULL JOIN

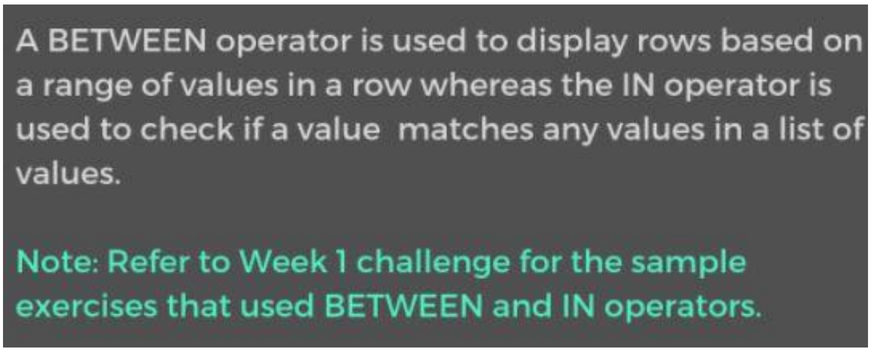
Finally, when we use FULL JOIN to join Tables A and B based on customer ID, it will return rows from both tables even if there are no matching customer IDs in the other table. A FULL JOIN is like a RIGHT JOIN and a LEFT JOIN at the same time.

3. **What is subquery in SQL?**



A subquery is a query within another query where a query is defined to retrieve data or information from the table. A subquery is usually executed first and the result of the subquery is passed on to the outer query or the main query.

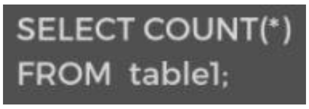
4. **Describe the main difference between a BETWEEN operator and IN operator?**



A BETWEEN operator is used to display rows based on a range of values in a row whereas the IN operator is used to check if a value  matches any values in a list of values.

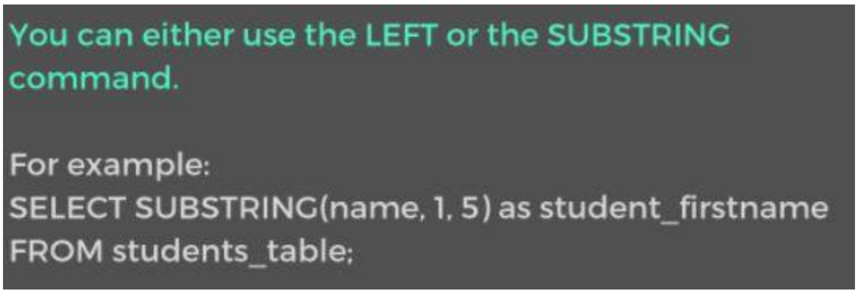
Note: Refer to Week 1 challenge for the sample exercises that used BETWEEN and IN operators.

5. **What is the SQL query to get the count of records in a table?**



```
SELECT COUNT(*)
FROM  table1;
```

6. **How can you fetch the first 5 characters of a string?**

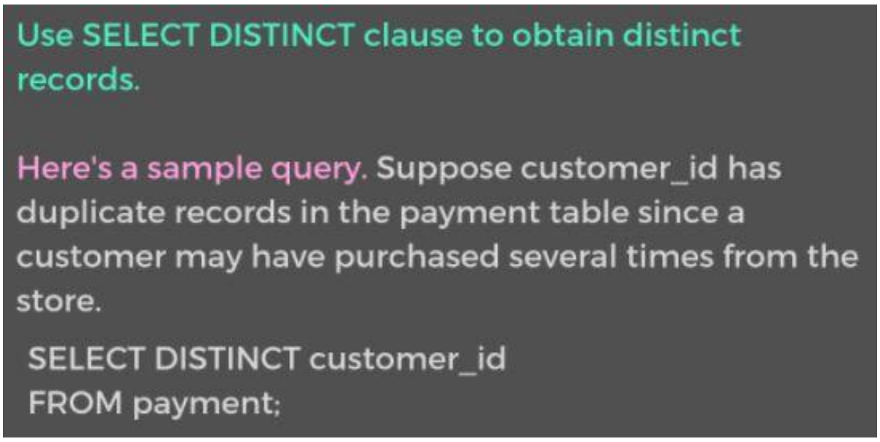


You can either use the LEFT or the SUBSTRING command.

For example:

```
SELECT SUBSTRING(name, 1, 5) as student_firstname
FROM students_table;
```

7. **How can you select unique records from a table?**

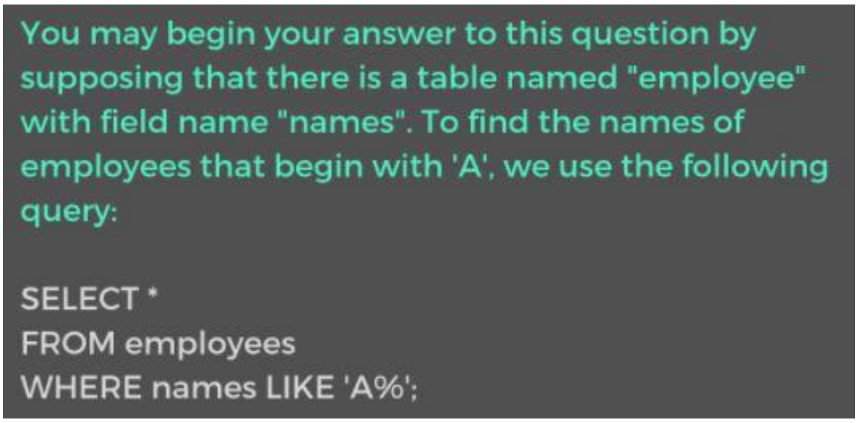


Use SELECT DISTINCT clause to obtain distinct records.

Here's a sample query. Suppose customer_id has duplicate records in the payment table since a customer may have purchased several times from the store.

```
SELECT DISTINCT customer_id
FROM payment;
```

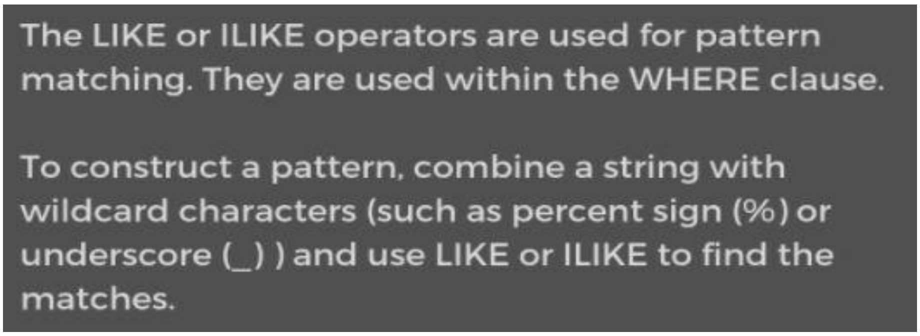
8. **What is the SQL query to find the names of employees that begin with 'A'?**



You may begin your answer to this question by supposing that there is a table named "employee" with field name "names". To find the names of employees that begin with 'A', we use the following query:

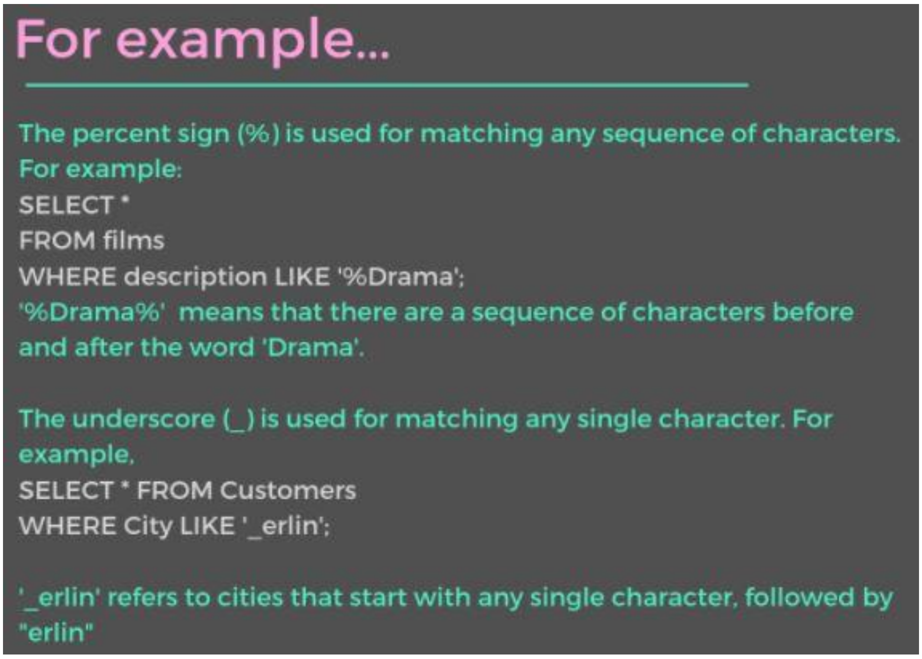
```
SELECT *
FROM employees
WHERE names LIKE 'A%';
```

9. **What operator or operators is used for pattern matching? And how do you construct a pattern?**



The LIKE or ILIKE operators are used for pattern matching. They are used within the WHERE clause.

To construct a pattern, combine a string with wildcard characters (such as percent sign (%) or underscore (_) ) and use LIKE or ILIKE to find the matches.



For example...

The percent sign (%) is used for matching any sequence of characters. For example:

```
SELECT *
FROM films
WHERE description LIKE '%Drama';
```

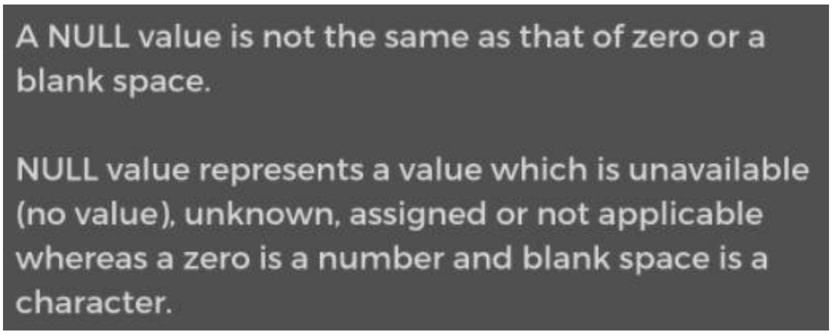
'%Drama%' means that there are a sequence of characters before and after the word 'Drama'.

The underscore (_) is used for matching any single character. For example,

```
SELECT * FROM Customers
WHERE City LIKE '_erlin';
```

'_erlin' refers to cities that start with any single character, followed by "erlin"

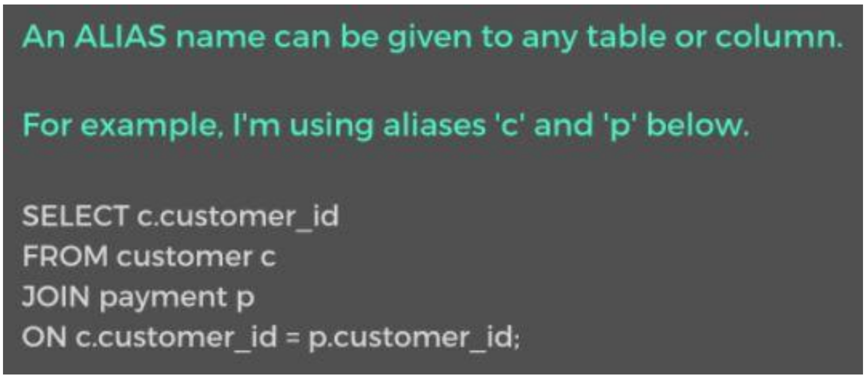
**10. Are null values the same zero or blank space?**



A NULL value is not the same as that of zero or a blank space.

NULL value represents a value which is unavailable (no value), unknown, assigned or not applicable whereas a zero is a number and blank space is a character.

**11. What is an ALIAS command?**

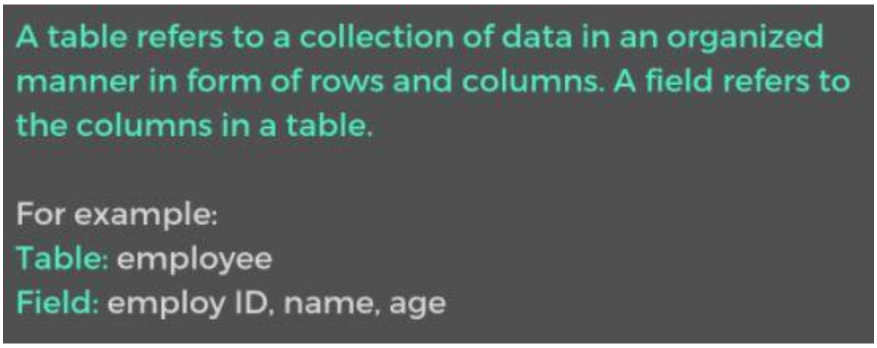


An ALIAS name can be given to any table or column.

For example, I'm using aliases 'c' and 'p' below.

```
SELECT c.customer_id
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id;
```

**12. Can you explain what a table and what a field represent in SQL?**



A table refers to a collection of data in an organized manner in form of rows and columns. A field refers to the columns in a table.

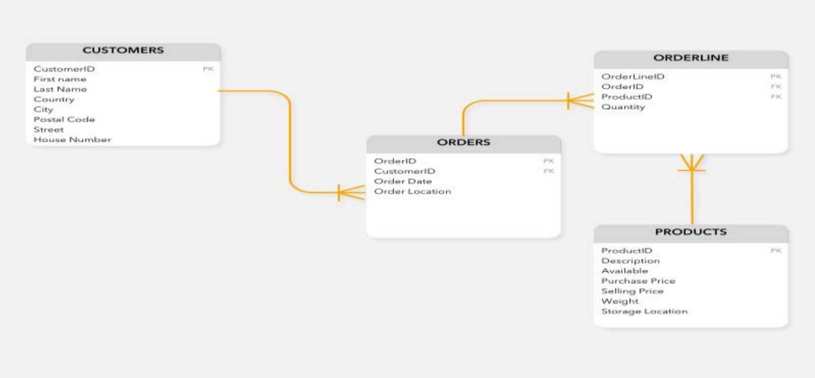
For example:

Table: employee

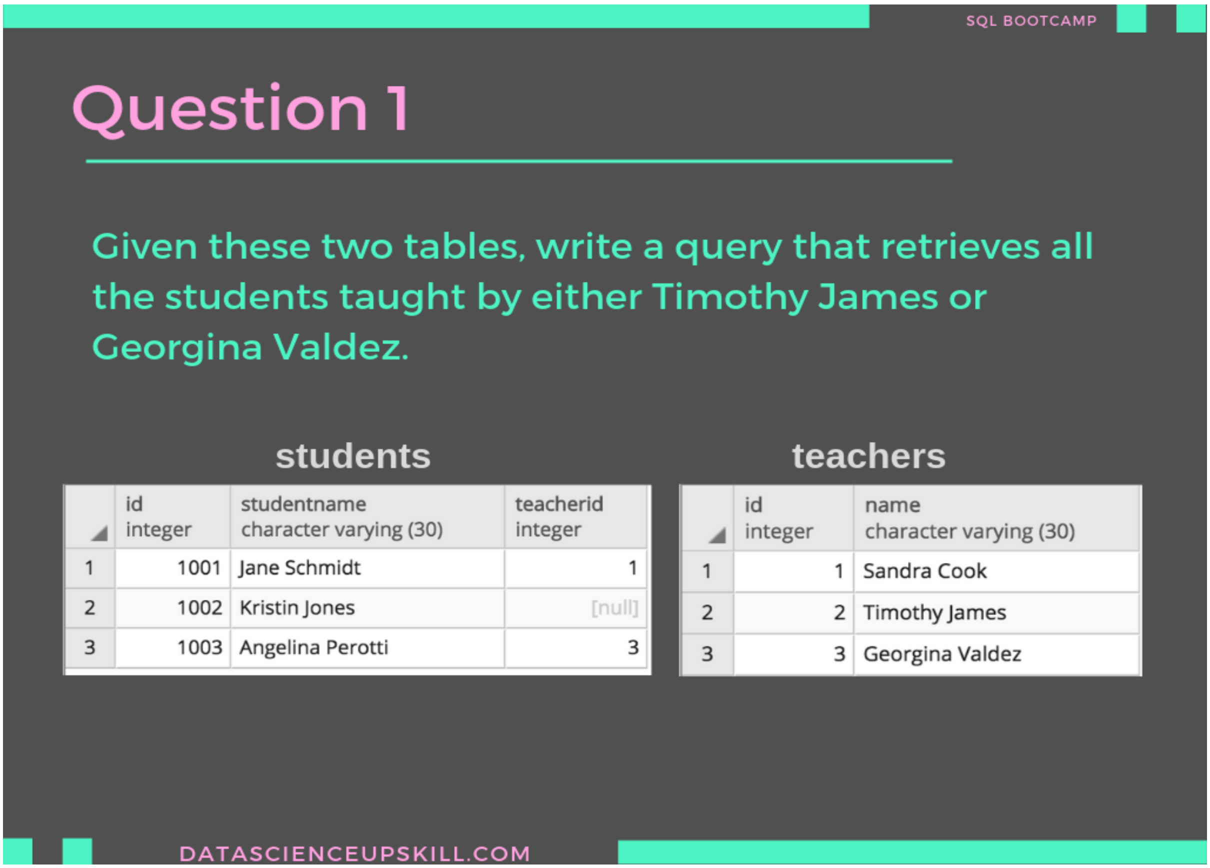
Field: employ ID, name, age

**13. What is ER diagram?**

An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities.



# <<<<<<SQL WRITTEN INTERVIEW PREPARATION>>>>>>



SQL BOOTCAMP

## Question 1

Given these two tables, write a query that retrieves all the students taught by either Timothy James or Georgina Valdez.

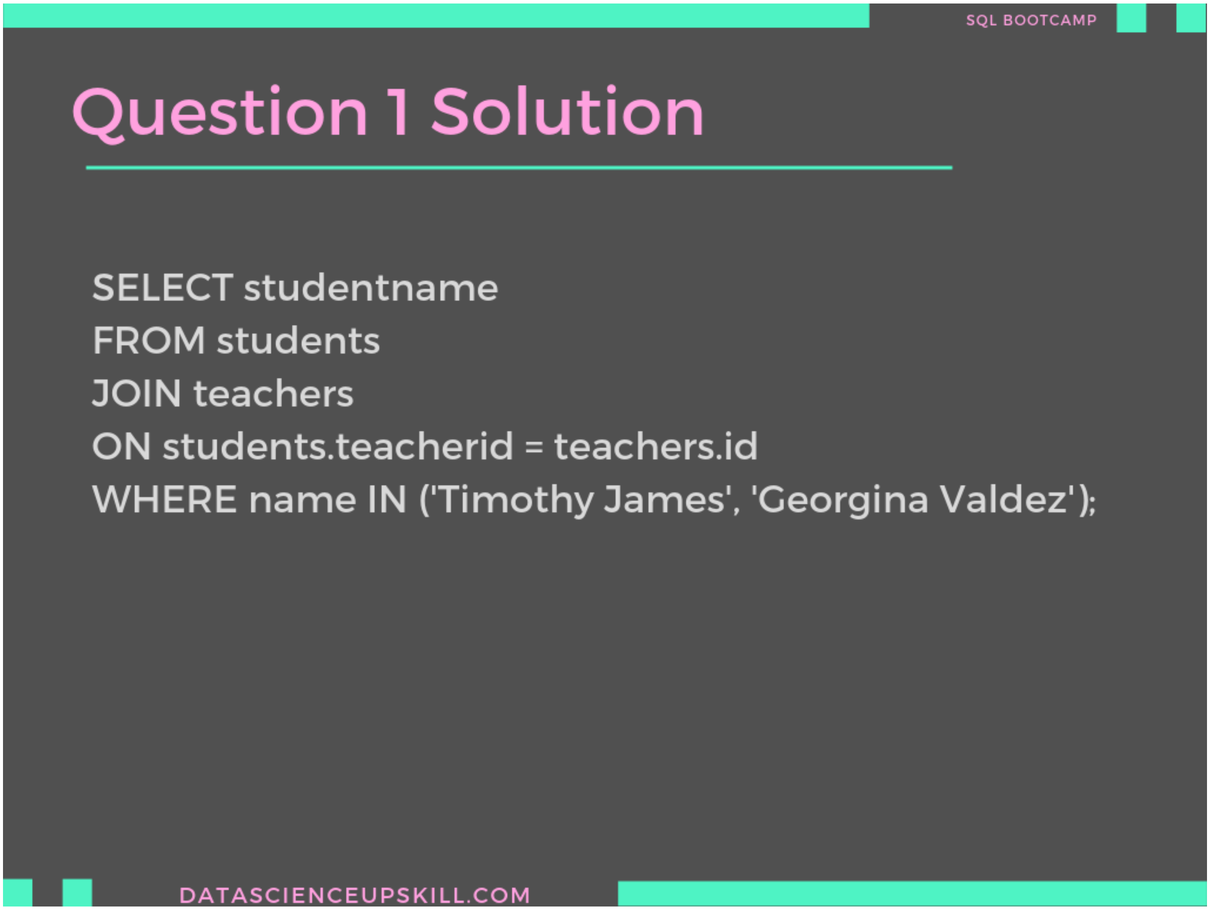
**students**

| | id<br>integer | studentname<br>character varying (30) | teacherid<br>integer |
|---|---|---|---|
| 1 | 1001 | Jane Schmidt | 1 |
| 2 | 1002 | Kristin Jones | [null] |
| 3 | 1003 | Angelina Perotti | 3 |

**teachers**

| | id<br>integer | name<br>character varying (30) |
|---|---|---|
| 1 | 1 | Sandra Cook |
| 2 | 2 | Timothy James |
| 3 | 3 | Georgina Valdez |

DATASCIENCEUPSKILL.COM



SQL BOOTCAMP

## Question 1 Solution

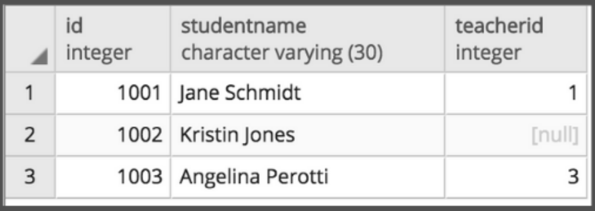
```
SELECT studentname
FROM students
JOIN teachers
ON students.teacherid = teachers.id
WHERE name IN ('Timothy James', 'Georgina Valdez');
```

DATASCIENCEUPSKILL.COM

# Question 2

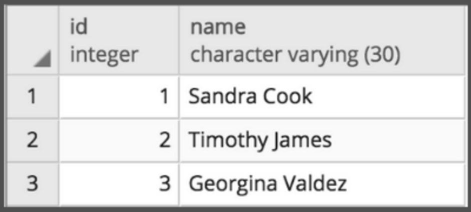
Given these two tables, write a query that retrieves all the students taught with no or missing teacher.

students



|  | id<br>integer | studentname<br>character varying (30) | teacherid<br>integer |
|---|---|---|---|
| 1 | 1001 | Jane Schmidt | 1 |
| 2 | 1002 | Kristin Jones | [null] |
| 3 | 1003 | Angelina Perotti | 3 |

teachers



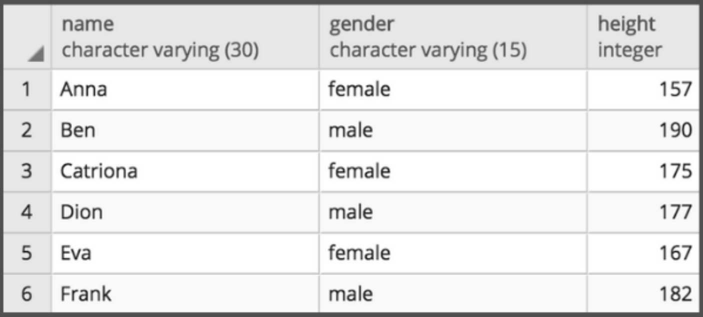
|  | id<br>integer | name<br>character varying (30) |
|---|---|---|
| 1 | 1 | Sandra Cook |
| 2 | 2 | Timothy James |
| 3 | 3 | Georgina Valdez |

# Question 2 Solution

```
SELECT studentname
FROM students
JOIN teachers
ON students.teacherid = teachers.id
WHERE teacherid IS NULL;
```

# Question 3

Suppose you are given a table called 'heights' with field names 'name', 'gender' (indicating male/female) and 'height' (shown in cm). Write a query to find the average height of people over 160 cm tall by gender. Round the average height up to 2 decimal places.

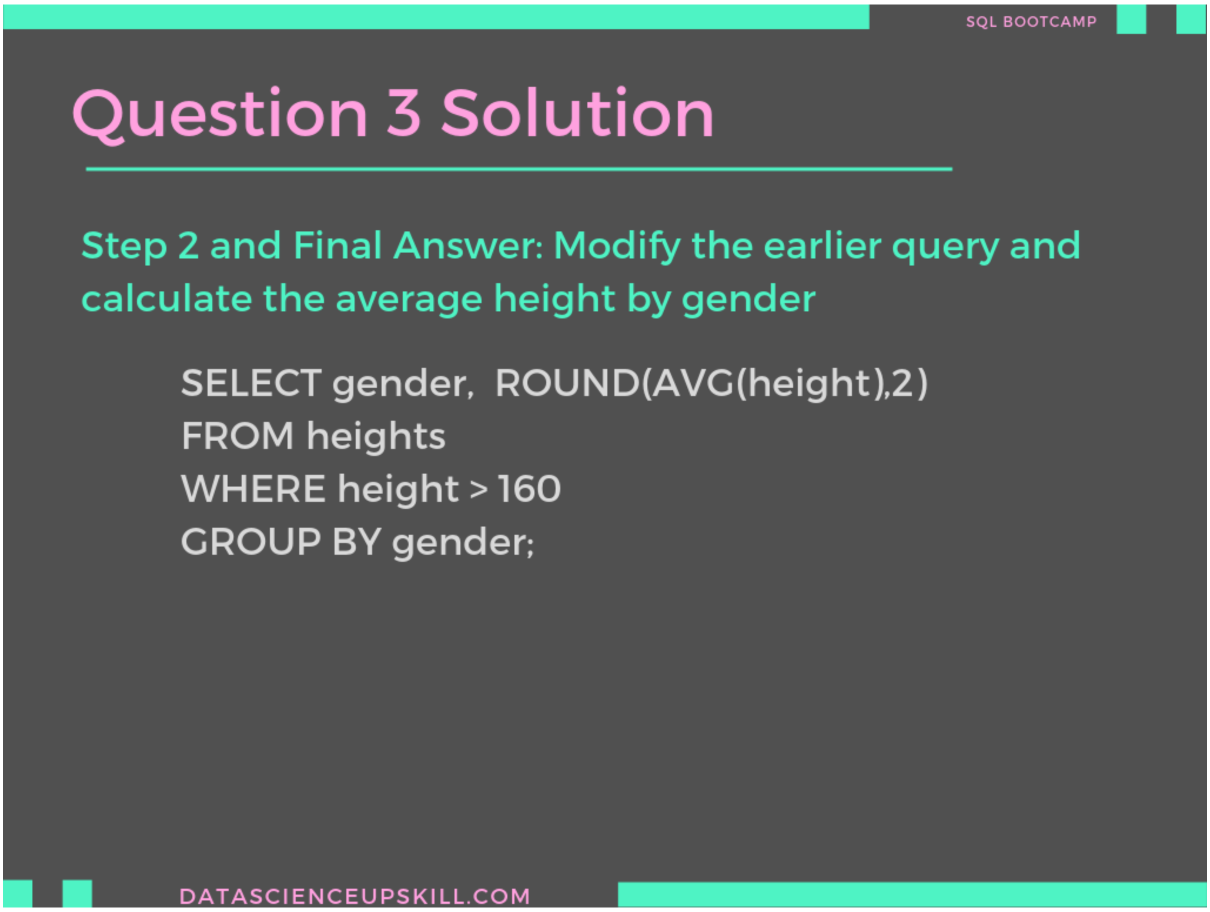


| | name<br>character varying (30) | gender<br>character varying (15) | height<br>integer |
|---|---|---|---|
| 1 | Anna | female | 157 |
| 2 | Ben | male | 190 |
| 3 | Catriona | female | 175 |
| 4 | Dion | male | 177 |
| 5 | Eva | female | 167 |
| 6 | Frank | male | 182 |

# Question 3 Solution

Step 1: First, let's retrieve the height of people over 160 cm tall

```
SELECT height
FROM heights
WHERE height > 160
```



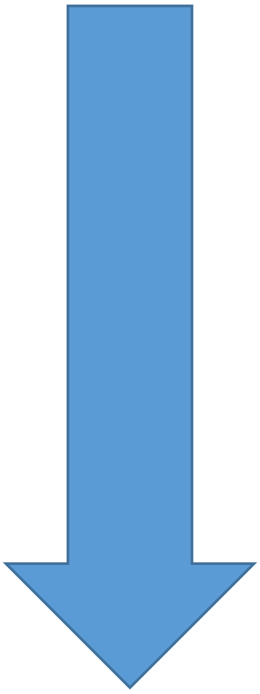
SQL BOOTCAMP

# Question 3 Solution

Step 2 and Final Answer: Modify the earlier query and calculate the average height by gender

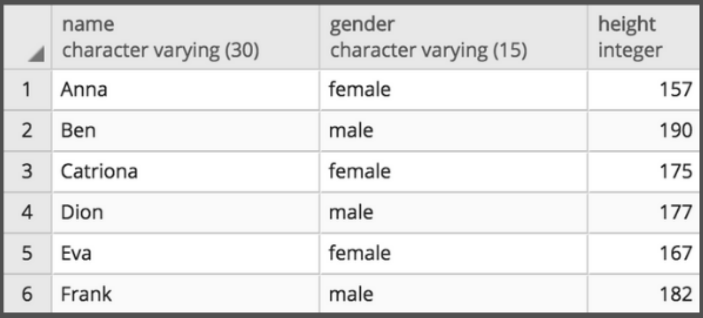
```
SELECT gender,  ROUND(AVG(height),2)
FROM heights
WHERE height > 160
GROUP BY gender;
```

DATASCIENCEUPSKILL.COM



# Question 4

Given the heights table, write a query showing the minimum and maximum heights of people belonging to a particular gender. Only show the case wherein the minimum height for the group is at least 160 cm.

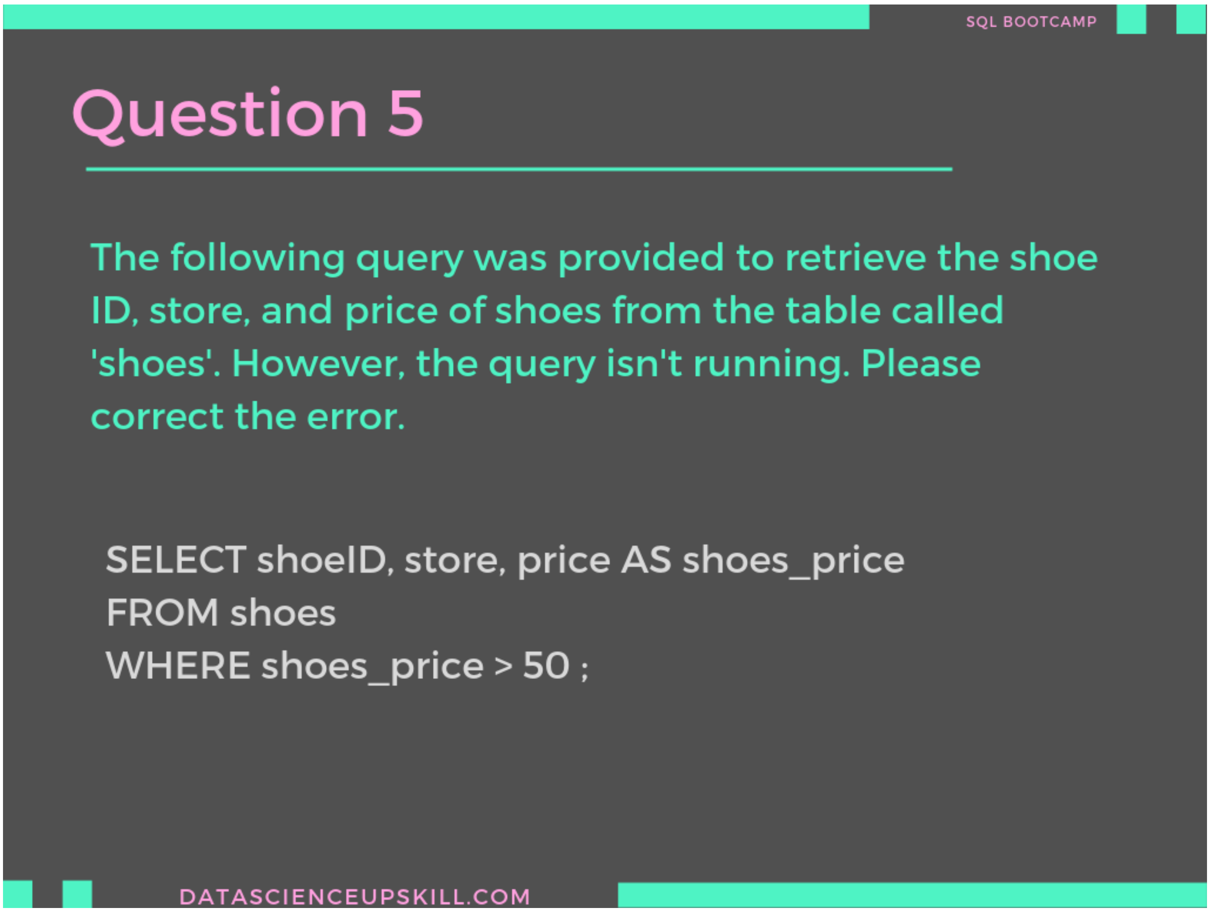


|  | name<br>character varying (30) | gender<br>character varying (15) | height<br>integer |
|---|---|---|---|
| 1 | Anna | female | 157 |
| 2 | Ben | male | 190 |
| 3 | Catriona | female | 175 |
| 4 | Dion | male | 177 |
| 5 | Eva | female | 167 |
| 6 | Frank | male | 182 |

# Question 4 Solution

```
SELECT gender, MIN(height) AS MinHeight,
MAX(height) AS MaxHeight
FROM heights
GROUP BY gender
HAVING MIN(height) > 160;
```

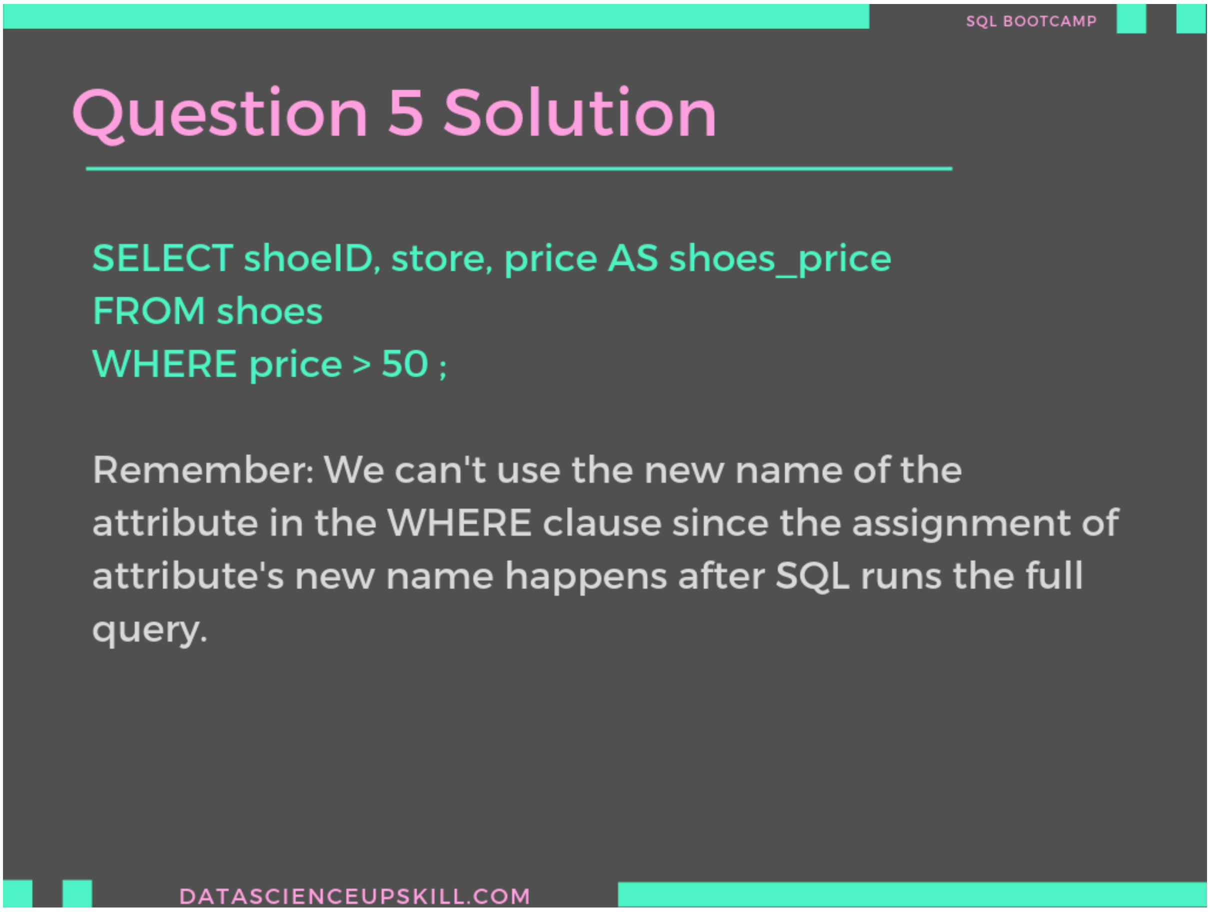
*Remember, we use HAVING statement (not a WHERE clause) to add a condition using an aggregate function.



# Question 5

The following query was provided to retrieve the shoe ID, store, and price of shoes from the table called 'shoes'. However, the query isn't running. Please correct the error.

```
SELECT shoeID, store, price AS shoes_price
FROM shoes
WHERE shoes_price > 50 ;
```



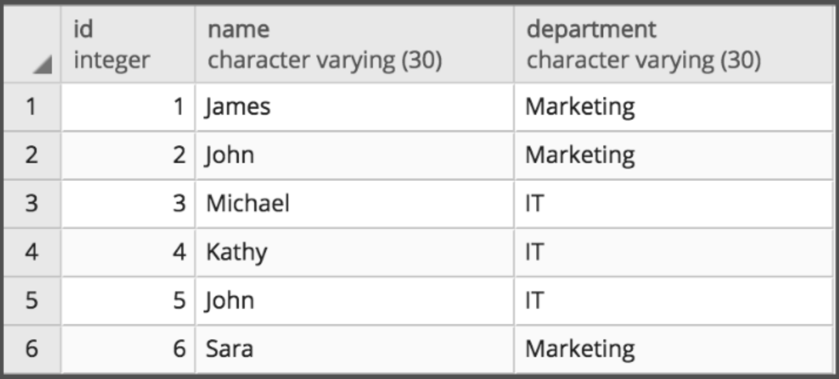
# Question 5 Solution

```
SELECT shoeID, store, price AS shoes_price
FROM shoes
WHERE price > 50 ;
```

Remember: We can't use the new name of the attribute in the WHERE clause since the assignment of attribute's new name happens after SQL runs the full query.

# Question 6

- Write a query to find all the duplicate names in a table named Employee.



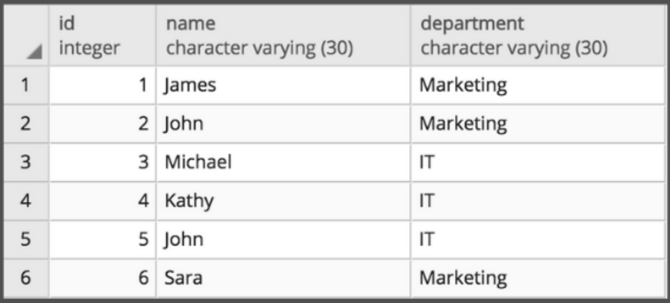
| | id<br>integer | name<br>character varying (30) | department<br>character varying (30) |
|---|---|---|---|
| 1 | 1 | James | Marketing |
| 2 | 2 | John | Marketing |
| 3 | 3 | Michael | IT |
| 4 | 4 | Kathy | IT |
| 5 | 5 | John | IT |
| 6 | 6 | Sara | Marketing |

# Question 6 Solution

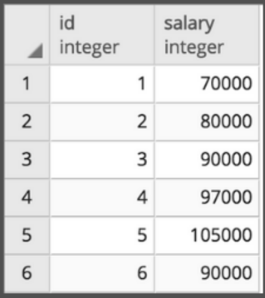
```
SELECT Name
FROM Employee
GROUP BY Name
HAVING COUNT(Name) > 1
```

# Question 7

- Using the two tables below, write a query to find employees who have the highest salary in each of the departments



| | id<br>integer | name<br>character varying (30) | department<br>character varying (30) |
|---|---|---|---|
| 1 | 1 | James | Marketing |
| 2 | 2 | John | Marketing |
| 3 | 3 | Michael | IT |
| 4 | 4 | Kathy | IT |
| 5 | 5 | John | IT |
| 6 | 6 | Sara | Marketing |

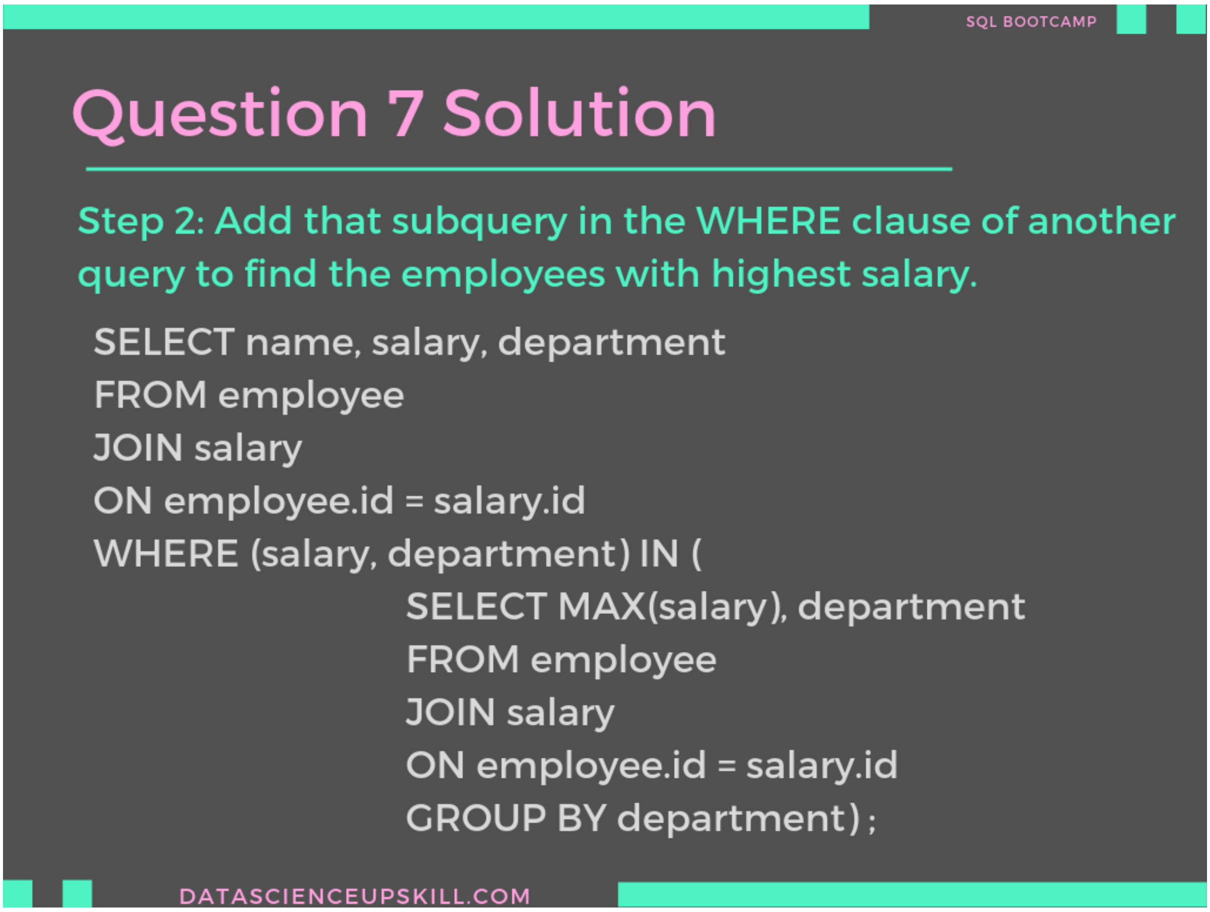


| | id<br>integer | salary<br>integer |
|---|---|---|
| 1 | 1 | 70000 |
| 2 | 2 | 80000 |
| 3 | 3 | 90000 |
| 4 | 4 | 97000 |
| 5 | 5 | 105000 |
| 6 | 6 | 90000 |

# Question 7 Solution

Step 1: Start by calculating the maximum salary per department. This will be a subquery.

```
SELECT MAX(salary), department
FROM employee
JOIN salary
ON employee.id = salary.id
GROUP BY department ;
```



SQL BOOTCAMP

# Question 7 Solution

Step 2: Add that subquery in the WHERE clause of another query to find the employees with highest salary.

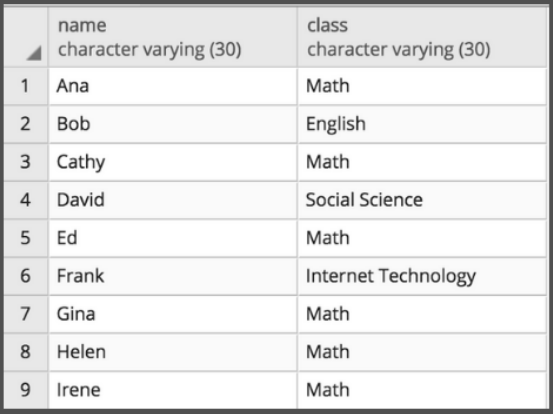
```
SELECT name, salary, department
FROM employee
JOIN salary
ON employee.id = salary.id
WHERE (salary, department) IN (
                              SELECT MAX(salary), department
                              FROM employee
                              JOIN salary
                              ON employee.id = salary.id
                              GROUP BY department) ;
```

DATASCIENCEUPSKILL.COM

# Question 8

- Write a query to find the class or classes with at least 5 students.

courses

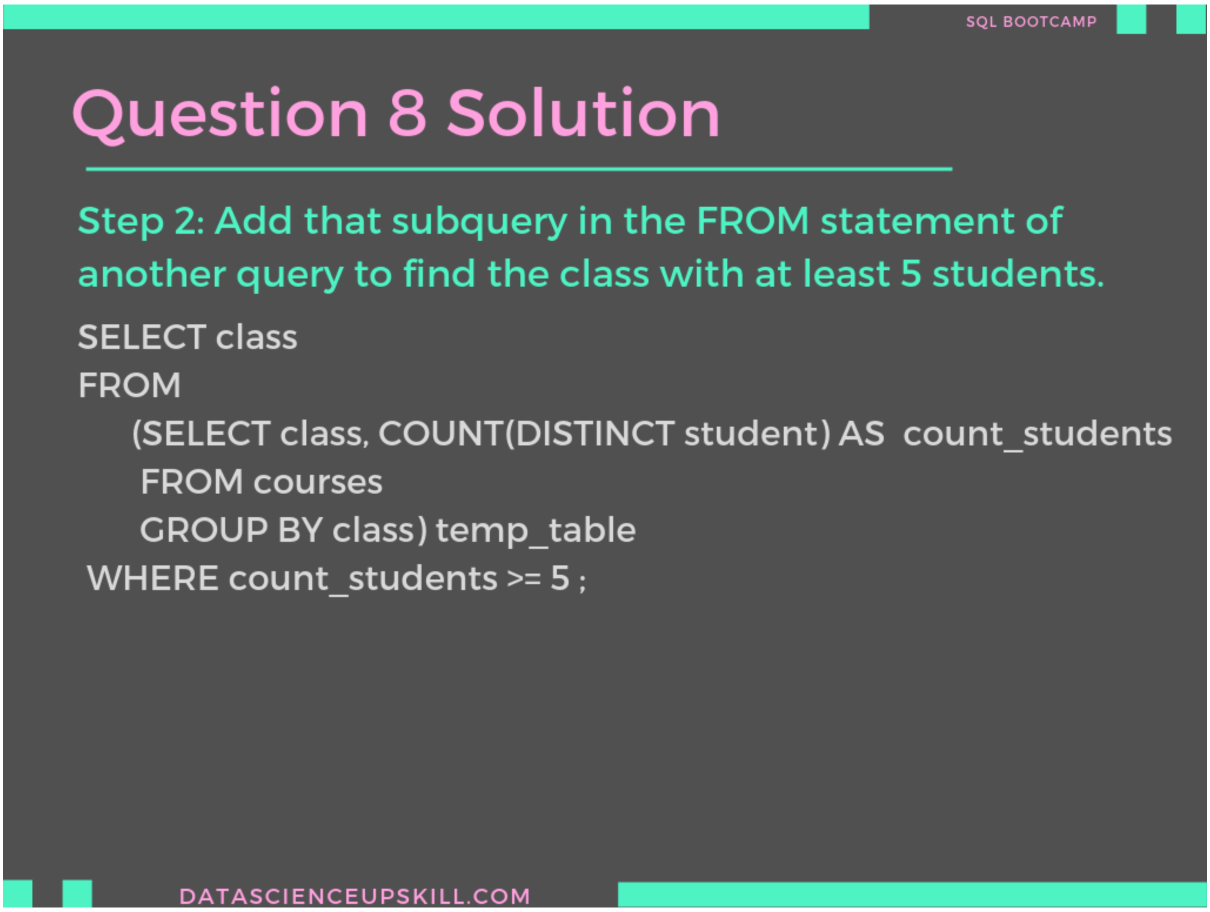


|  | name<br>character varying (30) | class<br>character varying (30) |
|---|---|---|
| 1 | Ana | Math |
| 2 | Bob | English |
| 3 | Cathy | Math |
| 4 | David | Social Science |
| 5 | Ed | Math |
| 6 | Frank | Internet Technology |
| 7 | Gina | Math |
| 8 | Helen | Math |
| 9 | Irene | Math |

# Question 8 Solution

Step 1: First, create a subquery which counts the number of students in each class.

```
SELECT class, COUNT(DISTINCT student) AS  count_students
     FROM courses
     GROUP BY class ;
```



# Question 8 Solution

Step 2: Add that subquery in the FROM statement of another query to find the class with at least 5 students.

```
SELECT class
FROM
      (SELECT class, COUNT(DISTINCT student) AS  count_students
       FROM courses
       GROUP BY class) temp_table
 WHERE count_students >= 5 ;
```