Technical Test IC – QA Automation

Juan Gabriel Orozco

Part 2

1. **Explain the difference, in databases, between 'Having' and 'where' when it comes to a query. Develop one example for each one of these two cases and point out the difference.**

Both instructions are used to filter, the main difference is that WHERE is to filter a data set and the sentence doesn't need grouping, while HAVING is used with on aggregate functions and must be used with grouped data.

For example, If I want to know employees that lives in Medellín I can use WHERE:

```
SELECT first_name FROM employees

WHERE city='Medellín';
```

But if I want to know customers with orders over $50, I must use HAVING, because the SUM function is necessary here to obtain a total:

```
SELECT first_name, address FROM customers

GROUP BY first_name, address

 HAVING SUM(order_total) > 50;
```

2. **Write a query to create a data table 'Student' with the following attributes in it: 'Name', 'Code', 'Class', 'Age', 'Favorite Subject', 'GPA' (5.0 scale)**

```
CREATE TABLE Student (

   Name varchar(255),

   Code int,

   Class varchar(255),

   Age int,

   FavoriteSubject varchar(255),

   Gpa float(8)

);
```

3. **Insert at least 40 records in the last table with close to real data.**

```
INSERT INTO Students (Name, Code, Class, Age, FavoriteSubject,
Gpa)
VALUES
('Juan', '1', 'Engineering', '25', 'Physics', '3.1'),
('Andrea', '2', 'Engineering', '19', 'Calculus', '4.2'),
('Carlos', '3', 'Engineering', '20', 'Physics', '3.5'),
('Esteban', '4', 'Engineering', '23', 'Mechanics', '4.8'),
('Sofia', '5', 'Engineering', '25', 'Metalurgy', '3.1'),
('Paula', '6', 'Engineering', '21', 'Physics', '3.5'),
('Andrey', '7', 'Engineering', '22', 'Calculus', '4.2'),
('Pablo',  '8',  'Engineering',  '25',  'Electromagnetism',
'3.1'),
('Gabriel', '9', 'Engineering', '21', 'Physics', '3.7'),
('Andrey', '10', 'Engineering', '22', 'Calculus', '3.1'),
('Mary', '11', 'Medicine', '20', 'Biology', '3.5'),
('Isabella', '12', 'Medicine', '30', 'Morphology', '4.8'),
('Arturo', '13', 'Medicine', '28', 'Morphology', '4.1'),
('Anna', '14', 'Medicine', '23', 'Pathology', '4.8'),
('Sebastian', '15', 'Medicine', '20', 'Biology', '3.1'),
('Daniel', '16', 'Medicine', '21', 'Pathology', '4.1'),
('Lauren', '17', 'Medicine', '22', 'Pathology', '3.1'),
('Manuel', '18', 'Medicine', '26', 'Biology', '4.1'),
('Rodrigo', '19', 'Medicine', '21', 'Biology', '3.2'),
('Fernando', '20', 'Medicine', '23', 'Chemistry', '3.5'),
('Mara', '21', 'Pyshics', '25', 'Calculus', '3.2'),
('Antony', '22', 'Pyshics', '33', 'Thermodynamics', '4.6'),
('Antonella', '23', 'Pyshics', '20', 'Thermodynamics', '4.2'),
('Valentina', '24', 'Pyshics', '22', 'Thermodynamics', '3.5'),
('Alvaro', '25', 'Pyshics', '23', 'Quantum Mechanics', '3.7'),
('laura', '26', 'Pyshics', '21', 'Quantum Mechanics', '3.7'),
('Marcos', '27', 'Pyshics', '21', 'Thermodynamics', '3.5'),
('Mateo', '28', 'Pyshics', '25', 'Quantum Mechanics', '4.2'),
('Jimmy', '29', 'Pyshics', '21', 'Quantum Mechanics', '3.1'),
('Michael', '30', 'Pyshics', '22', 'Calculus', '2.8'),
('Allyson', '31', 'Nursing', '22', 'First Aids', '3.9'),
('Laura', '32', 'Nursing', '35', 'First Aids', '3.2'),
('John', '33', 'Nursing', '20', 'Palliative Care', '4.2'),
('Jhonny', '34', 'Nursing', '19', 'First Aids', '3.9'),
('Sergio', '35', 'Nursing', '23', 'Palliative Care', '4'),
('Vanessa', '36', 'Nursing', '22', 'First Aids', '3.9'),
('Daniela', '37', 'Nursing', '26', 'First Aids', '3.5'),
('Alejandro', '38', 'Nursing', '25', 'Palliative Care', '3.4'),
('Paul', '39', 'Nursing', '24', 'First Aids', '3.1'),
('Michelle', '40', 'Nursing', '26', 'Palliative Care', '4.2');
```

4. **Write a query to get the average GPA from all the students whose name starts with ‘A’.**

```
SELECT gpa FROM Students

WHERE name LIKE 'A%';
```

5. **Write a query to get the list of students that are in the same class, have a GPA higher than 3.5/5.0 and order them by Age and Name.**

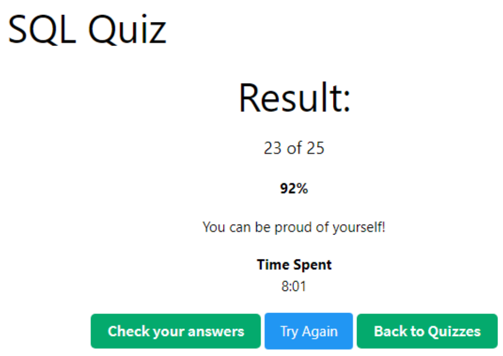
```
SELECT * FROM Students
WHERE Class='Nursing'
AND gpa > 3.5
ORDER BY age, name;
```

6. **Write a query to get the list of all the students with ‘Name', ‘Code', ‘Class’, ‘Age’, ‘Favorite Subject', ‘GPA’**

```
SELECT Name, Code, Class, Age, FavoriteSubject, Gpa

FROM Students
```

7. **Take the following 25 question quiz about SQL, please include a screenshot about the results and time it took to take the test.**



SQL Quiz

Result:

23 of 25

92%

You can be proud of yourself!

Time Spent

8:01

Part 3

1. **What is the difference between a unit test, an acceptance test, an integration test, and an end-to-end test?**

    - **Unit test** – Focused on testing of application modules. Every module is tested isolated from the others.
    - **Integration test -** Focused on the interactions between modules and if when they are put together no errors are introduced.
    - **Acceptance test** – These tests focus on verifying whether the application meets the requirements.
    - **End-to-end test** – Its purpose is to simulate the real user scenario and interactions. Is focused on testing the application flow under real world scenario. \
    -

2. **Could you explain Cohn's automation pyramid?**

    This pyramid shows different testing layers and the recommended number of tests created by layer. The lower you go in the pyramid the more tests need to be written, the more isolated must be and they are faster to execute. For example, unit test is the lowest layer and the tests in this layer must test one component isolated from the others, while the highest layer (GUI Tests) cannot be isolated, because an interface is a combination of elements and modules interacting with each other. Here we can see that GUI tests are slower to execute than the unit test, because the scenarios are more complex, and more components are involved.

    At the top of the pyramid, we have a cloud that represents manual testing. This is very important because creating automated test for every scenario is impossible, and manual testing is very useful with these scenarios that are complex to automate, also manual test explore diverse scenarios that are not planned and leads to a deep understanding of the application.

3. **Could you explain the difference between a black box testing and white box testing?**
    We talk about **Black Box Testing** when we don't know how the systems works in technical terms. Only the responses to inputs are known, so, it's focused on checking if the outputs are expected for every input, while the **White Box Testing** requires technical knowledge about the application and what processes are running inside, this type of testing needs technical knowledge that's why not everyone can made it.

4. **What is the purpose of an exploratory test and when is it useful to run them?**
   Its purpose is to discover, investigate and learn about the application. Is useful to run this type of test when the tester wants to learn more about the application and its functionalities.

5. **Mention at least 5 test design techniques and explain them briefly.**
   - **Decision table:** Aims to create a table with combinations of inputs or events and their respective outputs.
   - **Boundary Value Analysis:** Is based on testing the boundaries between partitions. It includes the boundaries limits inside and outside values.
   - **State transition:** This technique aims to evaluate the system behavior through entering input conditions in a sequence.
   - **Error guessing:** This technique is based on guessing errors using a list of possible errors. This list is created by an experienced tester which can use his skills and past experiences to identify similar conditions that lead to errors.
   - **Equivalence class portioning:** Aims to split the input data into classes of data from the test cases should be designed.

6. **What is the purpose of the following types of tests?**
   a. **Functional test:** Validate if an application meets the requirements.
   b. **Performance test:** Verify the speed and reliability of an application.
   c. **Security test:** Detect if exists security vulnerabilities in the system.
   d. **Usability test:** Identify how people are interacting with the product and how to make it more intuitive and easier to use.
   e. **API test:** Improve the security of APIs, authentication, data protection, performance, response times, etc.
   f. **Unit test:** Check small pieces of code, fix errors earlier and make sure that some parts of the code are still working after changes.