**SQL TEST FOR ANALYTICS POSITIONS**

**1. Tell me how you’ve used Relational Database Management System (RDBM) in the past. What did you use it for and why?**

I used RDBM during my studies and personal projects (one project was to analyze international debt data collected by The World Bank in order to know what is the average amount of debt owed by countries across different debt indicators etc.)

In my previous job, I implemented a mock database from online extracts (downloading csv files from our online ordering website for example) in order to build better reports and indicators. Unfortunately, the CEO did not appreciate the solution so it was discarded (one of the reasons why I decided to move on – no use of technology and data to generate better insights in the decision-making process).

**2. What standard SQL commands do you know?**

I mostly use data definition language (to create my database and tables) and data manipulation language. I haven’t used TCL or DCL.

List of commands: CREATE, ALTER, DROP, TRUNCATE, RENAME, SELECT, UPDATE, INSERT, DELETE

**3. Give me an example of how you’ve used Null and Zero values in the past?**

Null values are dangerous for calculations (operations with a null value return a null value). It is particularly dangerous when adding a new numeric column that will be used for calculation: historical values (meaning data existing before the addon) will be nulls and can alter calculations. It is then better to coalesce the nulls to transform them into another value.

For example, in adding a column of percentage of debt held by domestic actors (which will then be used in a multiplication to determine the amount of debt held by domestic and foreign actors), I had to replace the null values for prior years (when data was not available).

**4. Explain the definition of a dimension of data warehousing? What are the primary functions of the dimensions?**

A dimension is generally an attribute of the transaction data that is stored in its own table for normalization.

For example, a car dealership just sold a car. Let’s say we have a table with transaction\_id (for this sale transaction\_id = 123) and car\_id (car\_id = abc). In this case, we would have a dimension table that has a list of car ids, make, model, year, mileage etc. For example car abc is a: Peugeot, 206, 2017, 150 000 kms etc.

**5. Explain the difference between an inner join and outer join using an example.**

An inner join will combine two tables only for the records that are present in both tables (it returns the intersection of the two tables). The left outer join will return the entire left table and add the attributes for the records that are also in the right table (and create nulls for the other records). The right outer join is the inverse (but return the full right table …). The full outer join will return the integrality of the two tables.

Inner join:

Employees

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Country\_id** |
| **1** | **Josh** | **32** |
| **2** | **Maria** | **23** |

Countries

|  |  |
| --- | --- |
| **ID** | **Name** |
| **32** | **Brasil** |
| **27** | **France** |

Inner join Query

**SELECT e.Name as Name, c.Name as Country**

**FROM Employees as e**

**INNER JOIN Countries as c**

**ON e.Country\_id = c.ID**

|  |  |
| --- | --- |
| **Name** | **Country** |
| **Josh** | **Brasil** |

Left join Query:

**SELECT e.Name as Name, c.Name as Country**

**FROM Employees as e**

**LEFT JOIN Countries as c**

**ON e.Country\_id = c.ID**

|  |  |
| --- | --- |
| **Name** | **Country** |
| **Maria** | **Null** |
| **Josh** | **Brasil** |

Full join Query:

**SELECT e.Name as Name, c.Name as Country**

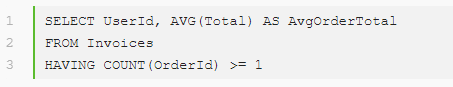
**FROM Employees as e**

**FULL JOIN Countries as c**

**ON e.Country\_id = c.ID**

|  |  |
| --- | --- |
| **Name** | **Country** |
| **Maria** | **Null** |
| **Josh** | **Brasil** |
| **Null** | **France** |

**6. I am looking for average order by customer when the customer has at least one order. What is wrong with the SQL query below?**

This query will return an error because the HAVING argument must be present in the SELECT statement.

Also, we can imagine that an OrderID might be repeated (if the order is composed of several things, depending on the database/table structure). For example, an order might be for one soccer ball, a Barcelona jersey and a pair of cleats. We could add DISTINCT to the HAVING clause (HAVING COUNT (DISTINCT OrderId) >= 1 - after having corrected the SELECT statement.

**7. Please review the table below. Write a query that retrieves the employees’ names and recruiters’ names. If an employee was not hired by a recruiter, leave the cell blank.**

**People table**

|  |  |  |
| --- | --- | --- |
| Person\_id | Name | Recruited\_by |
| 1 | Jean Grayson | Null |
| 2 | Paul Smith | 7 |
| 3 | John Do | Null |
| 4 | Alex Lee | 7 |
| 5 | Lisa Kim | 7 |
| 6 | Bob Thompson | 3 |
| 7 | Mike Keen | Null |
| 8 | Raymond Red | 3 |
| 9 | Alisson Jones | 1 |
| 10 | Kate James | 3 |

Query:

SELECT

a.Name as Recruitee,

b.Name as Recruiter

FROM people as a

LEFT JOIN people as b

ON a.Recruited\_by = b.Person\_id;

**8. Please review the table below. Write a query that retrieves the names of the recruiters that hire more than 3 employees, and the # of employees who were not hired by a recruiter.**

**People table**

|  |  |  |
| --- | --- | --- |
| Person\_id | Name | Recruited\_by |
| 1 | Jean Grayson | 1 |
| 2 | Paul Smith | 7 |
| 3 | John Do | Null |
| 4 | Alex Lee | 7 |
| 5 | Lisa Kim | 7 |
| 6 | Bob Thompson | 3 |
| 7 | Mike Keen | Null |
| 8 | Raymond Red | 3 |
| 9 | Alisson Jones | 1 |
| 10 | Kate James | 3 |

I wrote two queries because I did not see the “value” of adding them together (the answer to the second part of the question is a single number while the first part might return several rows).

Query 1:

WITH staging as (

SELECT

a.Name as Recruitee,

b.Name as Recruiter

FROM people as a

LEFT JOIN people as b

ON a.Recruited\_by=b.Person\_id

)

SELECT

Recruiter

FROM staging

GROUP BY Recruiter

HAVING COUNT(Recruiter) > 3;

Query 2:

SELECT

COUNT(\*) as nbr\_not\_hired\_by\_recruiter

FROM people

WHERE COALESCE(recruited\_by, 9) = 9;

**9. Please review the table below. Write a query that retrieves the two companies that have the highest duplicate employee entrees.**

**People table**

|  |  |  |
| --- | --- | --- |
| Person\_id | Name | Company\_id |
| 1 | Jean Grayson | 2 |
| 2 | Paul Smith | 7 |
| 3 | John Do | 1 |
| 4 | Alex Lee | 7 |
| 5 | Lisa Kim | 7 |
| 6 | Bob Thompson | 3 |
| 7 | Mike Keen | 1 |
| 8 | Raymond Red | 3 |
| 9 | Alisson Jones | 1 |
| 10 | Kate James | 3 |
| 11 | Lisa Kim | 7 |
| 12 | Bob Thompson | 3 |
| 13 | Alex Lee | 7 |
| 14 | Lisa Kim | 7 |
| 15 | Bob Thompson | 3 |
| 16 | Mike Keen | 1 |
| 17 | Raymond Red | 3 |
| 18 | Mike Keen | 1 |
| 19 | Raymond Red | 3 |

WITH duplicates as (

SELECT

Name,

Company\_id,

COUNT(\*) as duplicate\_count

FROM people

GROUP BY Name, Company\_id

HAVING COUNT(\*) > 1

)

SELECT

company\_id

FROM duplicates

GROUP BY company\_id

ORDER BY SUM(duplicate\_count) DESC

LIMIT 2;