**Tasks for Data Engineer position:**

**SQL:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | order\_item | |  | orders | |  | article | |  | parts\_list | |  | category | |  |
|  | KEY | ID\_ORDER\_ITEM |  | KEY | PID\_ORDER |  | KEY | PID\_ARTICLE |  | KEY | ID\_PARTS\_LIST |  | KEY | PID\_CATEGORY |  |
|  | ABC | PID\_ORDER\_ITEM |  | TS | ORDER\_TIME |  | ABC | FID\_CATEGORY |  | ABC | FID\_PARTS\_LIST |  | ABC | FID\_CATEGORY |  |
|  | ABC | FID\_ARTICLE |  |  |  |  | ABC | FID\_PARENT\_ARTICLE |  | ABC | FID\_ARTICLE |  | ABC | FID\_COMPANY |  |
|  | 123 | UNITS |  |  |  |  | ABC | FID\_PARTS\_LIST |  | 123 | UNITS |  | 123 | HIERARCHY\_LEVEL |  |
|  | 123 | UNITPRICE |  |  |  |  | ABC | ERP\_NUMBER |  |  |  |  | ABC | NAME |  |
|  | ABC | FID\_ORDERS |  |  |  |  | ABC | NAME |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

These are some tables in the DWH. We prepared you sample Data in setup file, that you can use to setup a test environment. After Initializing you should write the following SQL Queries.

Write the following SQL Queries:

1. All orders and their SUM, between 02.01.2019 and 04.01.2019.
2. How many red, green and blue blocks have been sold in total including the ones that are part of the Starter Set? Articles may have multiple levels of child articles. The sample data contain only one level, but the query should be able to handle multiple levels.
3. Top 3 Products of 2019 in comparison to 2018 by SUM OF PRICE

Our Setupfile (.sql) is optimized for SQL Server but you are free to use alternatives!

**PYTHON:**

This is a short problem to measure your knowledge of data structures and code design in Python ( preferred Python 3.8).

There are 3 tasks below. You will first share your code with us and explain your solution shortly in the interview.

Let’s assume that you have an arbitrary dictionary d.

e.g.

d = {

    'key11':

        {

            'key21': 'a',

            'key22': 'b',

        },

    'key12': 'c',

    'key13':

            {

                'key31': 'd',

                'key32':

                    {

                        'key21': 'e',

                        'key22': 'f',

                    },

            },

}

Q1- How can you update leaves (values) of a dictionary ? e.g., values 'a', 'b', ....'f'.

such that the maps 'a' -> 'aa', 'b'->'bb' and so on are applied!

Q2- Write up a function that does this algorithmically for an arbitrary dictionary ?-----------------------

2.1 just update d

2.2 return a dictionary without mutating d

------------------------

Now, what if we had other data types

d2 = {

    'key11':

        {

            'key21': 'a',

            'key22': 'b',

        },

    'key12': 'c',

    'key13':

            {

                'key31': 'd',

                'key32':

                    {

                        'key21': 'e',

                        'key22': 'f',

                    },

            },

    'key14': [1,2,3],

    1 : (2,3),

}

Q3. Can you generalize your function to deal with the cases above ?

where list and strings are doubled e.g., 'a' -> 'aa',

[1,2,3] -> [1,2,3,1,2,3]  and tuples are not modified.

---------------------

3.1 just update d2

3.2 return a dictionary without mutating d2

--------------------

Hint: a neat style of coding with doc strings and type indication is appreciated.

Hint2: The dictionaries above are small but what if you have a large dictionary

with 10-times nested sub dictionaries.

**PYSPARK:**

Introduction

Set up an Apache Spark environment and import the provided CSV files(articles.csv, transactions.csv). These files must be combined to generate the target table as described below. This task is estimated to take about 2-4 hrs.

You could use a [Databricks trial account](https://www.databricks.com/try-databricks?utm_medium=cpc&utm_source=google&utm_campaign=8984002507&utm_offer=try-databricks&utm_content=trial&utm_term=databricks#account) to work on the data and use it as your development

environment.

Project “Data Integration”

As a user, I would like to combine the files TRANSACTION and ARTICLE into a table using as foreign key the ARTICLE\_ID0MATERIAL column. The target table should include all fields and be transformed as necessary to not have duplicate rows.

The files have been provided by the customer as is and need to be examined for problems and the data validated and cleaned if necessary. This is very close to a real world scenario and documenting the issues encountered is as important as the approach taken to address them.

Acceptance Criteria

* The TARGET TABLE has been generated and transformed.
* No duplicate rows exist in the TARGET table.
* All the fields in the TARGET table are in their correct format (string, double, date)
* The transformations, cleaning etc. should be performed in PySpark
* Possible problems with the data and/or issues that could create problems with the
* generation of the TARGET table have been identified and addressed and/or reported.
* The code and the comments regarding the approach taken have been included in a

jupyter notebook file (ipynb) or a databricks notebook file (DBC) or a python source file(.py)