const sql = require('mssql');

const connectionString = 'some-cnn-string';

const query1 = 'some-query-1';

const query2 = 'some-query-2';

const query3 = 'some-query-3';

async function queryAndLog(*connection*, *query*, *result*){

    const result = await connection.query(query, result)

*console*.log(result);

    return result;

}

function logAndReturn(*result*) {

*console*.log(result);

return {

...result

};

}

*module*.*exports* = {

    getAccountId: function(*apiKey*, *cb*) {

        sql.connect(connectionString, (*connection*)=>{

            connection.query(query1, (*result*)=>{

*console*.log(result1);

                connection.query(query2, { ...result1 }, (*result*)=>{

*console*.log(result2);

                    connection.query(query3, { ...result2 },

                    (*result*)=>{

*console*.log(result3);

                    })

                })

            });

        })

    }

    // Issues: from current function:

    //      - It is having 'cb', I assumed it is a callback function for getAccountId, however there is no place so we call it , I think it is redundant.

    //      - As the function name is getAccountId, but when I check the code, there is no code for returning the 'Id' value, I think it is missing something.

    //      - I think the line: 'console.log(result1)' just log an undefined value, I think it should be 'result', instead of 'result1', the console.log(result1)

    //      - I am not sure it supported many input parameters in .query method or not, when I check the examples on

    //  //

    //

    // My suggestion to enhance the function:

    // Resolve by return an id when completed instead of calling a callback, I assume the result is an object which has a property is 'id'

    // Remove callback hell

    // function name is get id so it should return id (in the question, it doesn't)

    // remove duplicated console.log va logAndReturn

    // Method1:

    goodGetAccountId: async (*apiKey*)=>{

        let connection = null;

        try {

     connection = await sql.connect(connectionString);

     const result1 = await connection.query(connection, query1);

    const result2 = await connection.query(connection, query2, {...result1});

     const result3 = await connection.query(connection, query3, {...result2});

     return result3.id;

     } catch (err) {

     // ... error handle

     }   finally {

         // close connection

         if(connection!==null){

             connection.close();

         }

     }

}

// Method 2

goodGetAccountId: async (*apiKey*) => {

try {

const connection = await sql.connect(connectionString);

return await connection.query(query1)

.then(logAndReturn)

.then(*result* => connection.query(query2, result))

.then(logAndReturn)

.then(*result* => connection.query(query3, result))

.then(logAndReturn)

.then(*result* => result.id)

} catch (err) {

// ... error handle

} finally {

    if (connection!== null){

        connection.close();

    }

}

}

};

//=============================