**Adam Roberts**

Technical Manager Assessment

Integral Ad Science

**Technical Manager Assessment**

**INTRODUCTION**

Each question is answered within this document from the Technical Manager Assessment. In addition to this document there are a few other resources that can be reviewed:

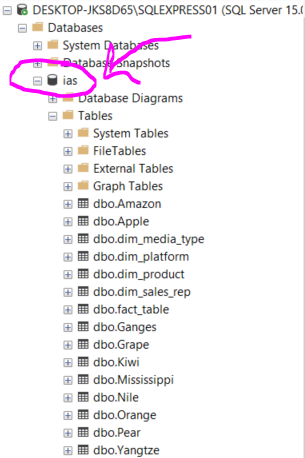
* **Test SQL Database**: Only created locally, but can share on screen during the interview. That may be helpful as the SQL in this document is quite long.
* **Jupyter Notebook**: Can be accessed via Github here: <https://github.com/admiral810/assessment_exercise>

**SQL (SQL Server) #1**

*1. In the provided dataset (located in Google Sheets, but feel free to export), there are several sheets representing data from different companies and one sheet entitled “Fact Table”. Imagine that each company sheet is a unique table in a SQL environment and write a query to combine the data from those into the format of the Fact Table.*

**CREATION OF DATABASE:**

I loaded the data to a local instance of SQL server, creating a database instance of **ias**. I can pull this up when I share my screen during the presentation, as I only created it locally.



**PRODUCT TYPE ASSUMPTION AND USER DEFINED FUNCTION:**

I made an assumption that product types that were not in the dim tables were “core” products for this exercise (Ex: ‘A’, ‘B’, ‘C’ were found as product types and not in the dim table). As this would be something managed multiple times, I created a user defined function to manage it.

CREATE FUNCTION udf\_clean\_prod(@prod\_in VARCHAR(20))

RETURNS VARCHAR(20)

AS

BEGIN

DECLARE @prod\_out VARCHAR(20)

SET @prod\_out = CASE

WHEN @prod\_in = 'Social' THEN 'Social'

WHEN @prod\_in = 'Video Platform' THEN 'Video Platform'

WHEN @prod\_in = 'New Business' THEN 'New Business'

WHEN @prod\_in = 'Experiment' THEN 'Experiment'

ELSE 'Core'

END

RETURN @prod\_out

END;

This user defined function will be called multiple times in the insert statements.

**INSERT STATEMENT**

Here is the insert query for each of the tables in SQL.

-- insert amazon data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format)

SELECT plat.ID, s.Company, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Revenue, s.Device, s.Format

FROM Amazon s

LEFT JOIN dim\_platform plat

ON s.Company = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert apple data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format, Units)

SELECT plat.ID, s.Platform, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Cost, s.Device, s.Format, s.Units

FROM Apple s

LEFT JOIN dim\_platform plat

ON s.Platform = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert ganges data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format)

SELECT plat.ID, 'Ganges', CAST(Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Revenue, s.Device, s.Format

FROM Ganges s

LEFT JOIN dim\_platform plat

ON 'Ganges' = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert grape data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format, Units, Environment)

SELECT plat.ID, s.Company, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Revenue, s.Device, s.Format, s.Widgets, s.Environment

FROM Grape s

LEFT JOIN dim\_platform plat

ON s.Company = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert kiwi data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format, Units, [Price per Thousand Units])

SELECT plat.ID, s.Company, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Revenue, s.Device, s.Format, s.Units, s.[Cost per thousand units]

FROM Kiwi s

LEFT JOIN dim\_platform plat

ON s.Company = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert mississippi data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format)

SELECT plat.ID, s.Platform, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Revenue, s.Device, s.Format

FROM Mississippi s

LEFT JOIN dim\_platform plat

ON s.Platform = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert nile data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format)

SELECT plat.ID, s.Partner, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.[Cost to Partner], s.Device, s.Format

FROM Nile s

LEFT JOIN dim\_platform plat

ON s.Partner = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert nile data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format)

SELECT plat.ID, s.Partner, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.[Cost to Partner], s.Device, s.Format

FROM Nile s

LEFT JOIN dim\_platform plat

ON s.Partner = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert orange data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format)

SELECT plat.ID, 'Orange', CAST(Date AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Revenue, s.Device, s.Format

FROM Orange s

LEFT JOIN dim\_platform plat

ON 'Orange' = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert pear data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format)

SELECT plat.ID, 'Pear', TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.[Revenue to Partner], s.Device, s.Format

FROM Pear s

LEFT JOIN dim\_platform plat

ON 'Pear' = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

-- insert yangtze data

INSERT INTO fact\_table ([Platform ID], Platform, Date, [Product ID], Product, Revenue, Device, Format, Environment)

SELECT plat.ID, s.Company, TRY\_CAST(CAST(Year AS varchar(4)) + Month AS DATE), prod.[Product ID], dbo.udf\_clean\_prod(s.Product), s.Revenue, s.Device, s.Format, s.Environment

FROM Yangtze s

LEFT JOIN dim\_platform plat

ON s.Company = plat.Platform

LEFT JOIN dim\_product prod

ON dbo.udf\_clean\_prod(s.Product) = prod.[Product Type];

**SQL (SQL Server) #2**

*2. Below is a sample SQL query of the test database (tdb) using the Microsoft SQL Server syntax. It is currently not working. Can you debug it?*

*select plat.Platform, rev.[Product] as prod, sum(rev.Revenue), sum.(rev.[Units]), reps.[Sales Representative] as reps*

*from tdb.[Fact\_Table] as rev*

*left join tdb.platform as plat*

*on rev.[Platform ID] = plat.[ID]*

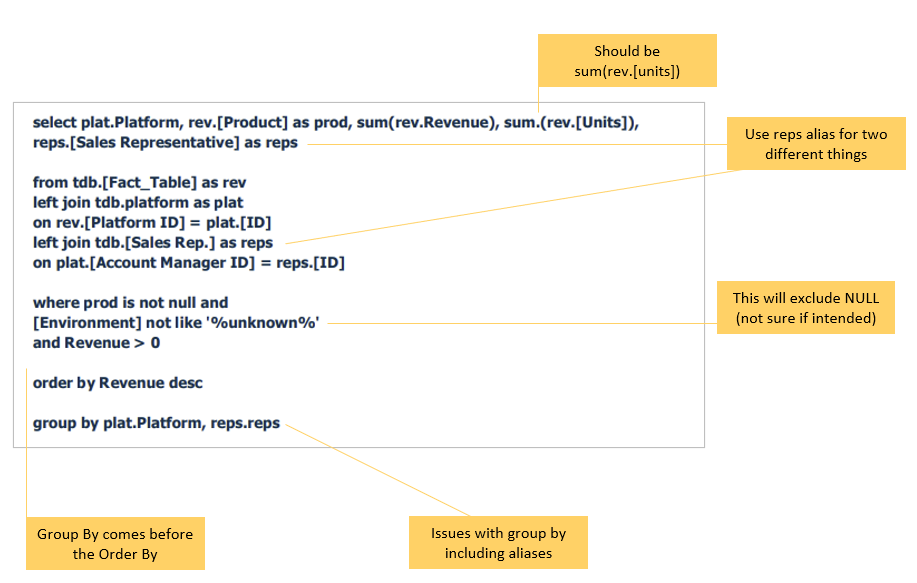
*left join tdb.[Sales Rep.] as reps*

*on plat.[Account Manager ID] = reps.[ID]*

*where prod is not null and [Environment] not like '%unknown%' and Revenue > 0 order by Revenue desc group by plat.Platform, reps.reps*

**IDENTIFIED ISSUES**

Here are some of the identified issues:



**MY QUERY**

This was the query I used to effectively pull the information. I can run this query during the presentation.

USE ias;

SELECT plat.Platform, rev.Product AS prod, reps.[Sales Representative] AS rep, sum(rev.Revenue) AS Revenue, sum(rev.Units) AS Units

FROM fact\_table as rev

LEFT JOIN dim\_platform AS plat

ON rev.[Platform ID] = plat.ID

LEFT JOIN dim\_sales\_rep reps

ON plat.[Account Manager ID] = reps.ID

WHERE rev.Product Is NOT NULL

AND Environment NOT LIKE '%unknown%' -- recommend omitting unless wish to exclude nulls

AND Revenue > 0

GROUP BY plat.Platform, reps.[Sales Representative], rev.Product

ORDER BY Revenue DESC

**Python #1**

*1. In the provided dataset, there are several sheets representing data from different companies. Please provide visualizations showing revenue trends across the full set of clients, products, devices, and formats over time.*

**SEE NOTEBOOK**

This question is answered in the Jupyter notebook.

**Python #2**

*2. Suppose these companies are our clients. How would you go about deciding which products to pitch to which clients? How would your answer change if there were 1000 clients in the dataset?*

**PITCHING CLIENTS PRODUCTS**

The initial view I’d propose would be a gap analysis of what products they currently have and how that compares to their competitive set. For example, if they do not have any video or social our data shows that those have been growing products for companies that have purchased those services. Growth would indicate that those companies are seeing value in those products.

**IF THERE WERE 1000 CLIENTS HOW WOULD THAT DIFFER**

It would certainly make the data set stronger and more compelling! More clients would allow us to better create a competitive set, so rather than show all competitors we would be able to build stories off of similar competitive segments (Ex: geographies, industries, size, etc).

**Python #3**

*3. Assume that each active client provides new data to us, in the format given in their particular tab of the spreadsheet, on a monthly basis. How would you build an aggregated dataset? Which fields would you include, which would you allow to be nullable, and which would be non-nullable?*

**HOW WOULD YOU BUILD AND AGGREGATED DATASET**

The dataset currently has a lowest time period aggregation level of monthly (as opposed to days). That would likely be the preferred level of aggregation, though if daily data is provided that aggregation can be complete in a SQL data load.

Ideally there would be a standardized format with which data is received. I had built load scripts in SQL for each company as that way if one company changes something that can be managed on its own and not disrupt other data loads. However, if there are thousands of datasets coming in, a “standard format” would be great to manage the bulk of the data received. Exceptions to this standard format (hopefully the minority) can be managed.

**WHICH FIELDS INCLUDED AND WHICH ARE NULLABLE**

|  |  |
| --- | --- |
| Field | Nullable |
| Platform | No |
| Date OR Year and Month | No |
| Product | No |
| Revenue | No |
| Device | No |
| Format | No |
| Units | **Yes** |
| Environment | **Yes** |
| Price per Thousand Units | **Yes** |

**BI Data Modeling #1**

*a. If the data in the Fact Table were viewed in a BI visualization broken out by “Media Type” from the Media Type table, why would the revenue total not match the sum of the “Revenue” column in the Fact Table?*

**ANSWER**

This would be because there are multiple Media Types available for what information is in the Fact Table. For example, consider below in the Media Types dimension table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Device** | **Environment** | **Format** | **Media Type ID** | **Media Type** |
| Mobile | In-App | Display | 221 | Mobile In-App Display |
| Mobile | Browser | Display | 211 | Mobile Browser Display |

Device and Format are available for each record in the Fact Table, however Environment is often missing. As such, a record with Device=Mobile and Format=Display may be either Mobile In-App Display or Mobile Browser Display.

If you were to join off all of Device, Environment, and Format a lot of revenue would be missing as Environment is often NULL.

**BI Data Modeling #2**

*b. Using the notion of Parent-Child hierarchies, how would you structure the data and create a measure that checks whether a Platform is assigned to a specific sales rep. or a subordinate of theirs?*

**ANSWER**

In the dim\_sales\_rep table there is a parent-child relationship where the sales representative is assigned to a specific manager. For example, look at Thomas Liu in the below table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sales Representative** | **Manager** | **Office Location** | **BI Login** | **ID** |
| Kia Williams | Karen Phillips | US | kwilliams@company.com | AB2 |
| Ed Cabot | Devin Wexler | London | ecabot@company.com | CD2 |
| Thomas Liu | Olga Smith | Hong Kong | tliu@company.com | EF2 |
| Lena Messing | Thomas Liu | Hong Kong | lmessing@company.com | EF3 |

There are a few approaches that could be taken:

1. I believe there is another dimension table that would assist in this modelling, which has hierarchical roles of sales representatives and their managers. This would help in creating Groups/Sets in the BI interface to allow a drill down for managers to see each of their subordinates in addition to accounts they directly manage.
2. Change Manager to Team Manager if possible. In this case on row 3 Thomas Liu would be both the Sales Representative and the Team Manager. He would have access to all subordinate sales reps, as well as himself as a sales rep.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sales Representative** | **Manager** | **Office Location** | **BI Login** | **ID** |
| Kia Williams | Karen Phillips | US | kwilliams@company.com | AB2 |
| Ed Cabot | Devin Wexler | London | ecabot@company.com | CD2 |
| Thomas Liu | Thomas Liu | Hong Kong | tliu@company.com | EF2 |
| Lena Messing | Thomas Liu | Hong Kong | lmessing@company.com | EF3 |

*\*in this case it may be valuable to add an additional level to the hierarchy, the Managers manager*

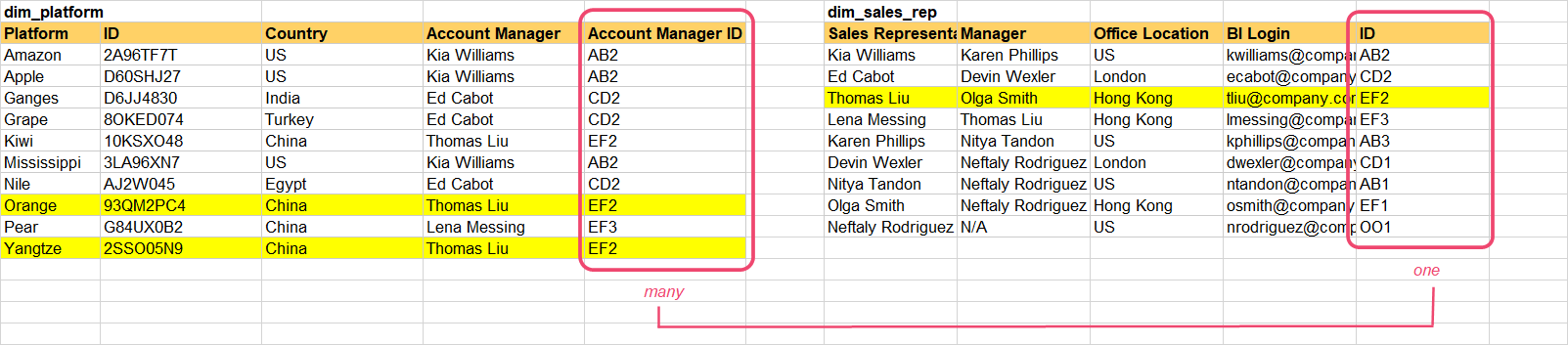
**BI Data Modeling #3**

*c. In our data model, the Platform table is joined to the Sales Rep. table on “Account Manager ID” and “ID”, respectively. Imagine applying the above measure to a filter on a visualization so it only shows data belonging to a rep. or their team. This visualization shows nothing when “Account Manager” is a field, but does populate correctly when “Sales Representative” is a field. What does this say about the relationship between the Platform table and the Sales Rep. table?*

**ANSWER**

I am interested in discussing this one more on our call as I want to make sure I’m understanding the circumstances. This one may be easier to troubleshoot in practice in a BI Tool, however I’ll look at some potential circumstances that may be a cause.

Here’s how those relationships work between the platform and sales\_rep dimensions tables. Thomas Liu is highlighted as an example as of a sales rep, however he is also a manager in a row (Pear Platform).



I believe that creating Thomas Liu as a Manager of himself could be the most straightforward solution, as he’d have an opportunity to look at his team as well as where he is a sales rep individually.

As BI login is only for the Sales Rep level in the table, it will be important that an Account Manager also has a login which provides access to their level.