MCB Assignment Documents

- **Based on the document provided, here are clear and understandable documentation points for the tasks outlined in the "MCB SQL Assessment**
- **The SQL script begins with the creation of a table named XXBCM_ORDER_MGT and includes the insertion of data into this table. Here's a summary of the key points based on the content**

1) Load the raw data into SQL server
2) To use DB_Prequisite.sql to create 3 tables like Invoice table, Orders table and Supplier table
3) 3rd point should be like
   A. Clean the data (Address, contact number split into 2 rows) part of contact number column cleans like placed on 5 mentioned in S, placed in Zero mentioned in 'o'
   B. Data Normalization: Following appropriate naming conventions and ensure tables are Normalized.
   C. Data Migration: Develop a SQL procedure to migrate the data "XXBCM_ORDER_MGT" to the newly created tables with proper data formatting.
4) Supplier tables are designed into a 1st Normal form and part of that second contact number inserted into another row like new row.

Inserting Order Management Data: The script contains multiple INSERT INTO statements to add records to the XXBCM_ORDER_MGT table. The fields included are:

- **ORDER_REF**: A reference number for the order.
- **ORDER_DATE**: The date the order was made.
- **SUPPLIER_NAME**: The name of the supplier.

Examples: ORDER_REF: 'PO001'

ORDER_DATE: '03-JAN-2022'

SUPPLIER_NAME: 'PEGASUS LTD'

SUPP_CONTACT_NAME: 'Georges Neero'

... and so on.

Examples:

4) The owner wishes to have a report displaying a summary of Orders with their corresponding list of distinct invoices and their total amount to be able to reconcile his orders and payments. The report shall contain the details as per table below ordered by latest Order Date on top. Implement a Stored Procedure or Function to return the required information.

```
SELECT SUBSTRING(O.[ORDER_REF],5,LEN(O.[ORDER_REF])) [Order Reference],

        LEFT(UPPER(CONVERT(VARCHAR,DATENAME(MM,[ORDER_DATE]))),3)+ '-
'+CONVERT(VARCHAR,YEAR([ORDER_DATE])) [Order Date],

        LEFT(S.[SUPPLIER_NAME],1)+''
+LOWER(SUBSTRING(S.[SUPPLIER_NAME],2,LEN(S.[SUPPLIER_NAME]))) [Supplier Name],

        Format([ORDER_TOTAL_AMOUNT],'##,##0.00') [Order Total Amount],

        [ORDER_STATUS] [Order Status],I.[INVOICE_REFERENCE] [Invoice Reference],

        Format(B.[INVOICE_TOTAL_AMOUNT],'##,##0.00')  [Invoice Total Amount],

        CASE WHEN I.[INVOICE_STATUS] ='Paid' THEN 'OK'

                WHEN I.[INVOICE_STATUS] ='Pending' THEN 'To follow up'

                WHEN I.[INVOICE_STATUS] ='' THEN 'Verify'

                else '' end AS Action

        FROM [dbo].[Orders] O

        LEFT JOIN [dbo].[Invoice] I ON I.[REFF_NO_SK]=O.[REFF_NO_SK]

        LEFT JOIN [dbo].[Supplier] S ON S.[SUPPLIER_NAME]=O.[SUPPLIER_NAME]

        LEFT JOIN

        (

        SELECT SUBSTRING([INVOICE_REFERENCE],5,5) INV_REF, SUM([INVOICE_AMOUNT])
[INVOICE_TOTAL_AMOUNT] FROM [MSBI].[dbo].[Invoice]

        GROUP BY SUBSTRING([INVOICE_REFERENCE],5,5)

        ) B ON O.[ORDER_REF]=B.INV_REF
```

Return details for the SECOND (2nd) highest Order Total Amount from the list. Only one record is expected with the following information. Implement a Stored Procedure or Function to return the required information

```
SELECT SUBSTRING(B.[ORDER_REF],5,LEN(B.[ORDER_REF]))  AS [Order Reference],
    CONVERT(VARCHAR,B.[ORDER_DATE],109) [Order Date],
      UPPER( B.[SUPPLIER_NAME]) [Supplier Name],
    Format(B.[ORDER_TOTAL_AMOUNT], '##,##0.00') [Order Total Amount],
       [ORDER_STATUS] [Order Status],
       I.[INVOICE_REFERENCE] AS [Invoice References]
FROM (
SELECT * FROM (
SELECT [ORDER_REF],[ORDER_DATE],[ORDER_TOTAL_AMOUNT]
,[ORDER_DESCRIPTION],[ORDER_STATUS],[ORDER_LINE_AMOUNT],[SUPPLIER_NAME
]
        ,ROW_NUMBER() over(order by [ORDER_TOTAL_AMOUNT] desc) rowno FROM
[MSBI].[dbo].[Orders]
  ) A   WHERE A.rowno=2) B
  INNER JOIN  (  SELECT LEFT([ORDER_REF],5)[ORDER_REF],
STRING_AGG([INVOICE_REFERENCE], ', ') [INVOICE_REFERENCE] FROM [dbo].[Invoice]
GROUP BY LEFT([ORDER_REF],5))
  I ON I.[ORDER_REF] = B.[ORDER_REF]
```

5) List all suppliers with their respective number of orders and total amount ordered from them between the period of 01 January 2022 and 31 August 2022. Output details as per below. Implement a Stored Procedure or Function to return the required information.

```
SELECT A.[SUPPLIER_NAME] [Supplier Name],A.[SUPP_CONTACT_NAME]  [Supplier
Contact Name]

,SUBSTRING(REPLACE(SUBSTRING(A.[SUPP_CONTACT_NUMBER],CHARINDEX(',',A.[SU
PP_CONTACT_NUMBER]),LEN(A.[SUPP_CONTACT_NUMBER])),', ',''),1,4)+'-
'+SUBSTRING(REPLACE(SUBSTRING(A.[SUPP_CONTACT_NUMBER],CHARINDEX(',',A.[S
UPP_CONTACT_NUMBER]),LEN(A.[SUPP_CONTACT_NUMBER])),',
',''),5,LEN(REPLACE(SUBSTRING(A.[SUPP_CONTACT_NUMBER],CHARINDEX(',',A.[SUPP_
CONTACT_NUMBER]),LEN(A.[SUPP_CONTACT_NUMBER])),', ',''))) [Supplier Contact No.
1]

,SUBSTRING(SUBSTRING(A.[SUPP_CONTACT_NUMBER],0,CHARINDEX(',',A.[SUPP_CO
NTACT_NUMBER])),1,4)+'-'+
SUBSTRING(SUBSTRING(A.[SUPP_CONTACT_NUMBER],0,CHARINDEX(',',A.[SUPP_CON
TACT_NUMBER])),4,LEN(SUBSTRING(A.[SUPP_CONTACT_NUMBER],0,CHARINDEX(',',A.[
SUPP_CONTACT_NUMBER])))) [Supplier Contact No. 2]
  ,COUNT(*) [Total Orders]
        , Format(SUM([ORDER_TOTAL_AMOUNT]), '##,##0.00') [Order Total Amount ]
        FROM (
        SELECT [SUPPLIER_NAME],[SUPP_CONTACT_NAME],[SUPP_ADDRESS]
     ,STRING_AGG([SUPP_CONTACT_NUMBER], ', ') [SUPP_CONTACT_NUMBER]
        ,[SUPP_EMAIL]
  FROM [MSBI].[dbo].[Supplier] S
  GROUP BY [SUPPLIER_NAME]
     ,[SUPP_CONTACT_NAME]
     ,[SUPP_ADDRESS]
     ,[SUPP_EMAIL]
        ) A INNER JOIN [dbo].[Orders] O ON A.[SUPPLIER_NAME]=O.[SUPPLIER_NAME]
        WHERE O.[ORDER_DATE] BETWEEN '2022-01-01' AND '2022-08-31'
```

```
   GROUP BY A. [SUPPLIER_NAME]
,A. [SUPP_CONTACT_NAME]
,A. [SUPP_ADDRESS]
,A.[SUPP_CONTACT_NUMBER]
,A.[SUPP_EMAIL]
```

Additional Notes:

- Output Listings: Provide sample outputs for each task.
- Tools Used: (SSMS) SQL server completing the assignments no other tools mandatory on this task as part of my experience.
- Assumptions: Assumptions made during the development or scripting process.

This documentation serves as guide to understanding the tasks, requirements, and expected outputs for the SQL assessment.