**Question 2-1:** Without using the ‘distinct’ function, write a query to return the result set in Table 2-2 from the values in Table 2-1.

Table 2-1:

|  |  |
| --- | --- |
| **Year** | **Customer** |
| 2005 | ABC Company |
| 2005 | ABC Company |
| 2005 | XYZ Company |
| 2006 | XYZ Company |

Table 2-2:

|  |  |
| --- | --- |
| **Year** | **CountOfDistinctCustomers** |
| 2005 | 2 |
| 2006 | 1 |

SELECT

A.[Year],

COUNT(A.Customer) CountOfDistinctCustomers -- Get Count

FROM (

SELECT [Year],

Customer

FROM tbl\_2\_1

GROUP BY

[Year],

Customer -- Grouping to remove duplicates

) A

GROUP BY

A.[Year]

Table 2-3:

|  |  |  |  |
| --- | --- | --- | --- |
| **Year** | **Quarter** | **Month** | **Booking Revenue** |
| 2005 | Q1 | 1 | 13 |
| 2006 | Q1 | 1 | 10 |
| 2006 | Q1 | 2 | 15 |
| 2006 | Q1 | 3 | 35 |
| 2006 | Q2 | 4 | 11 |
| 2006 | Q2 | 5 | 15 |
| 2006 | Q2 | 6 | 9 |
| 2007 | Q1 | 1 | 6 |
| 2007 | Q1 | 2 | 14 |
| 2007 | Q1 | 3 | 7 |
| 2007 | Q2 | 4 | 20 |
| 2007 | Q2 | 5 | 6 |
| 2007 | Q2 | 6 | 6 |
|  |  |  |  |

Table 2-5:

|  |  |  |  |
| --- | --- | --- | --- |
| **Quarter** | **2005 Bookings** | **2006 Bookings** | **2007 Bookings** |
| Q1 | 13 | 60 | 27 |
| Q2 |  | 35 | 32 |

Table 2-4:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Year** | **Quarter** | **Month** | **Current Year Quarter To Date Booking Revenue** | **Last Year Quarter To Date Booking Revenue** |
| 2005 | Q1 | 1 | 13 |  |
| 2006 | Q1 | 1 | 10 | 13 |
| 2006 | Q1 | 2 | 25 | 13 |
| 2006 | Q1 | 3 | 60 | 13 |
| 2006 | Q2 | 4 | 11 |  |
| 2006 | Q2 | 5 | 26 |  |
| 2006 | Q2 | 6 | 35 |  |
| 2007 | Q1 | 1 | 6 | 10 |
| 2007 | Q1 | 2 | 20 | 25 |
| 2007 | Q1 | 3 | 27 | 60 |
| 2007 | Q2 | 4 | 20 | 11 |
| 2007 | Q2 | 5 | 26 | 26 |
| 2007 | Q2 | 6 | 32 | 35 |

**Question 2-2:** Suppose you had the fact table, Table 2-3. Construct a query to calculate the year-on-year comparison of quarter-to-date revenue as in Table 2-4.

**Solution:**

-- Creating a CTE so I can reference it as a table

;WITH CTE AS (

SELECT

Year,

Quarter,

Month,

BookingRevenue,

SUM(BookingRevenue)OVER(PARTITION BY Year,Quarter

ORDER BY

YEAR ROWS UNBOUNDED PRECEDING

) [Current Year Quarter To Date Booking Revenue]

FROM tbl\_2\_3

)

,

Query AS (

SELECT

X.Year, X.Quarter, X.Month, X.[Current Year Quarter To Date Booking Revenue],

X.[Last Year Quarter To Date Booking Revenue],

COUNT([Last Year Quarter To Date Booking Revenue]) OVER (PARTITION BY X.QUARTER

ORDER BY X.ID

) countValue

FROM

(

SELECT

A.Year, A.Quarter, A.Month, A.BookingRevenue,

A.[Current Year Quarter To Date Booking Revenue],

B.[Current Year Quarter To Date Booking Revenue] [Last Year Quarter To Date

Booking Revenue],

ROW\_NUMBER()OVER(ORDER BY (SELECT 1)) ID

FROM CTE A

LEFT JOIN CTE B

ON A.Year = (B.Year+1)

AND A.Quarter = B.Quarter

AND A.Month = B.Month

)X

)

-- ===============================

-- FINAL Result Set

-- ===============================

SELECT

Year,

Quarter,

Month,

[Current Year Quarter To Date Booking Revenue],

ISNULL(CAST(CASE

WHEN [Last Year Quarter To Date Booking Revenue] IS NULL

THEN MAX([Last Year Quarter To Date Booking Revenue]) OVER (PARTITION BY

countValue)

ELSE [Last Year Quarter To Date Booking Revenue]

END AS VARCHAR(MAX)),'') [Last Year Quarter To Date Booking Revenue]

FROM Query

ORDER BY

Year,

Quarter,

Month

Table 2-5:

|  |  |  |  |
| --- | --- | --- | --- |
| **Quarter** | **2005 Bookings** | **2006 Bookings** | **2007 Bookings** |
| Q1 | 13 | 60 | 27 |
| Q2 |  | 35 | 32 |

**Question 2-3:** Without using the Crosstab or Pivot statement, construct the SQL statement to convert Table 2-3 into the pivoted form in Table 2-5

**Solution:**

-- Defining temp table to store aggregated result set

CREATE table #tempData

(

Year INT, Quarter VARCHAR(2), TotalBookings INT

)

-- Inserting data into temp table

INSERT INTO #tempData (Year, Quarter, TotalBookings)

SELECT

Year,

Quarter,

SUM(BookingRevenue) TotalBookings

FROM tbl\_2\_3

GROUP BY

Year,

Quarter

Order By

Year,

Quarter

-- ========================

-- DEFINE a Cursor

-- ========================

DECLARE PivotData CURSOR

FOR SELECT DISTINCT Year FROM #tempData ORDER BY Year

DECLARE @Year INT, @YearStr VARCHAR(MAX), @sql VARCHAR(MAX)

OPEN PivotData

FETCH PivotData INTO @Year

SET @sql=''

WHILE (@@FETCH\_STATUS = 0)

BEGIN

SET @YearStr=cast(@Year AS VARCHAR)

SET @sql = @sql+',ISNULL(CAST(SUM(CASE when Year =' + @YearStr+'THEN

TotalBookings ELSE NULL END) as VARCHAR),'''') ['+@YearStr+']'

FETCH PivotData INTO @Year

END

SET @sql='SELECT Quarter'+@sql+' FROM #tempData GROUP BY Quarter'

EXEC(@sql)

CLOSE PivotData

DEALLOCATE PivotData

DROP TABLE #tempData

# Data transformation examples

In this set of questions, you are asked to design a data transformation processes that will work in any SQL compliant DB (Postgres, MSSQL, MySQL, etc…) . The process should produce the requested output tables and should only require updates to a fix set of predefined tables (no manual intermediate steps).

Table 3-1: Transactions

|  |  |  |  |
| --- | --- | --- | --- |
| **Quarter** | **Product** | **Partner** | **List Price** |
| Q1-2006 | A | Reseller XYZ | $125 |
| Q1-2006 | B | Reseller PQ | $50 |
| Q1-2006 | C | Reseller PQ | $60 |
| Q2-2006 | A | Reseller PQ | $120 |
| Q2-2006 | B | Reseller XYZ | $50 |
| Q2-2006 | C | Reseller XYZ | $55 |

Table 3-2: Product Families

|  |  |
| --- | --- |
| **Product** | **Family** |
| A | Low End |
| B | Mid Range |
| C | High End |

Table 3-3: Product Class

|  |  |
| --- | --- |
| **Family** | **Class** |
| Low End | Commodity |
| Mid Range | Commodity |
| High End | Value |

Table 3-4: Partners

|  |  |
| --- | --- |
| **Partner ID** | **Partner** |
| 1 | Reseller XYZ |
| 2 | Reseller PQ |

Table 3-5: Partner Tiers

|  |  |  |
| --- | --- | --- |
| **Partner ID** | **Class** | **Tier** |
| 1 | Commodity | 3 |
| 1 | Value | 4 |
| 2 | Commodity | 2 |
| 2 | Value | 2 |

Table 3-6: Discounts

|  |  |  |  |
| --- | --- | --- | --- |
| **Tier** | **Low End** | **Mid Range** | **High End** |
| 1 | 10 | 15 | 25 |
| 2 | 13 | 18 | 28 |
| 3 | 14 | 19 | 30 |
| 4 | 15 | 21 | 32 |

Table 3-7: Net Price

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Quarter** | **Product** | **Partner** | **List Price** | **Net Price** |
| Q1-2006 | A | Reseller XYZ | $125 | $107.5 |
| Q1-2006 | B | Reseller PQ | $50 | $41 |
| Q1-2006 | C | Reseller PQ | $60 | $43.2 |
| Q2-2006 | A | Reseller PQ | $120 | $104.4 |
| Q2-2006 | B | Reseller XYZ | $50 | $40.5 |
| Q2-2006 | C | Reseller XYZ | $55 | $37.4 |

**Question 3-1:** Calculate the net price (= List Price x (1-Discount) ) for each transaction based on the discount structure available to each partner. The inputs are Tables 3-1 to 3-6 and the output table should be Table 3-7.

**Solution:**

ALTER TABLE dbo.ProductFamilies

ADD Class VARCHAR(50)

UPDATE pf

SET pf.Class = pc.Class

FROM dbo.ProductFamilies pf

INNER JOIN dbo.ProductClass pc

ON pf.Family = pc.Family

SELECT

A.Quarter,

A.Product,

A.Partner,

B.Discount,

A.ListPrice [List Price],

Round(A.ListPrice\*(1-(Cast(B.Discount as FLOAT)/100)),1) [Net Price]

FROM (

SELECT

T.Quarter, T.Product, T.ListPrice, T.Partner, P.PartnerID, pf.Family,pf.Class

FROM dbo.Transactions T

inner join dbo.Partners P

on T.Partner = P.Partner

INNER JOIN dbo.ProductFamilies pf

ON pf.Product = T.Product

)A

INNER JOIN (

SELECT

x.Tier,

Y.Discount,

Y.Family,

X.Class,

X.PartnerID

FROM (

SELECT

D.Tier,

D.LowEnd,

D.MidRange,

D.HighEnd,

PT.PartnerID,

PT.Class

FROM dbo.Discounts D

INNER JOIN dbo.PartnerTiers PT

on D.Tier = PT.Tier

)X

CROSS APPLY (

VALUES ('Low End', X.LowEnd)

('Mid Range', X.MidRange)

,('High End', X.HighEnd)

) y (Family, Discount)

)B

ON A.PartnerID = B.PartnerID

AND A.Family = B.Family

AND A.Class = B.Class

ORDER BY

A.Quarter,

A.Product

Table 3-8: Customer Master

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Customer Site ID** | **Parent Customer** | **City** | **State** | **Country** | **DUNS** |
| 1 | GE | Boston | MA | USA | 123123123 |
| 2 | GE | Boston | MA | USA |  |
| 3 | GE | Worcester | MA | USA |  |
| 4 | GE | Syracuse | NY | USA | 456456456 |
| 5 | GE | Syracuse | NY | USA |  |
| 6 | HSBC | London |  | UK | 789789789 |
| 7 | HSBC |  |  | UK |  |

Table 3-9: Customer Master w/Best Guess DUNS Numbers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Customer Site ID** | **Parent Customer** | **City** | **State** | **Country** | **DUNS** |
| 1 | GE | Boston | MA | USA | 123123123 |
| 2 | GE | Boston | MA | USA | 123123123 |
| 3 | GE | Worcester | MA | USA | 123123123 |
| 4 | GE | Syracuse | NY | USA | 456456456 |
| 5 | GE | Syracuse | NY | USA | 456456456 |
| 6 | HSBC | London |  | UK | 789789789 |
| 7 | HSBC |  |  | UK | 789789789 |

**Question 3-2:** Create Table 3-9, the Customer Master augmented with the best-guess DUNS number, from Table 3-8 the bare Customer Master. Select the DUNS number based on the closest match by city, state, and country in that order.

**Solution:**

-- Updating Table to Set blanks to NULL

UPDATE Customer\_Master

SET City = CASE WHEN City = '' THEN NULL ELSE City END,

State = CASE WHEN State = '' THEN NULL ELSE State END,

Country = CASE WHEN Country = '' THEN NULL ELSE Country END

FROM Customer\_Master

SELECT

Customer\_Site\_id,

Parent\_Customer,

City,

State,

Country,

DUNS

FROM Customer\_Master

WHERE DUNS IS NOT NULL

UNION

SELECT

A.Customer\_Site\_Id,

A.Parent\_Customer,

A.City,

A.State,

A.Country,

B.DUNS

FROM

(

SELECT \* FROM Customer\_Master

WHERE DUNS IS NULL

)A

INNER JOIN (

SELECT \* FROM Customer\_Master

WHERE DUNS IS NOT NULL

) B

ON CASE

WHEN A.City IS NULL OR B.City IS NULL

THEN

CASE

WHEN A.State IS NULL OR B.State IS NULL THEN A.Country

ELSE A.State END

ELSE

CASE WHEN A.City IS NOT NULL and B.City IS NOT NULL THEN A.City

WHEN A.State IS NOT NULL AND B.State IS NOT NULL THEN A.State

ELSE A.Country END

END

=

CASE

WHEN A.City IS NULL OR B.City IS NULL

THEN

CASE

WHEN A.State IS NULL OR B.State IS NULL THEN B.Country

ELSE B.State END

ELSE

CASE WHEN A.City IS NOT NULL and B.City IS NOT NULL THEN B.City

WHEN A.State IS NOT NULL AND B.State IS NOT NULL THEN B.State

ELSE B.Country END

END

**Question 3-3:** Calculate the employee attrition Table 3-11 from the Employee master, Table 3-10

**Solution:**

-- Static Date Paramters based on Output Requirements

DECLARE @Date\_Start DATE = '01/01/2007'

DECLARE @Date\_End DATE = '12/31/2007'

SELECT

Period,

[Total Employees at end of Quarter],

[High Performers],

[Medium Performers],

[Low Performers],

[Total Attrition],

[Low Performer Attrition],

[Medium Performer Attrition],

[High Performer Attrition]

FROM (

SELECT

-- Total Employee Counts Per Quarter

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date>'03/31/2007' OR Termination\_Date IS NULL) THEN 1 ELSE 0 END) [Q1 2007],

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date>'06/30/2007' OR Termination\_Date IS NULL) THEN 1 ELSE 0 END) [Q2 2007],

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date>'09/30/2007' OR Termination\_Date IS NULL) THEN 1 ELSE 0 END) [Q3 2007],

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date>'12/31/2007' OR Termination\_Date IS NULL) THEN 1 ELSE 0 END) [Q4 2007],

-- Q1 Performance Levels

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date>'03/31/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q1 Low],

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date>'03/31/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q1 Medium],

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date>'03/31/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q1 High],

-- Q1 Total Attrition

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date between '01/01/2007' AND '03/31/2007') THEN 1 ELSE 0 END) [Q1 Attrition],

-- Q1 Attrition By Levels

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date between '01/01/2007' AND '03/31/2007') AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q1 Low Attrition],

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date between '01/01/2007' AND '03/31/2007') AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q1 Medium Attrition],

SUM(CASE WHEN Start\_Date <= '01/01/2007' and (Termination\_Date between '01/01/2007' AND '03/31/2007') AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q1 High Attrition],

-- Q2 Performance Levels

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date>'06/30/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q2 Low],

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date>'06/30/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q2 Medium],

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date>'06/30/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q2 High],

-- Q2 Total Attrition

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date between '04/01/2007' and '06/30/2007') THEN 1 ELSE 0 END) [Q2 Attrition],

-- Q2 Attrition By Levels

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date between '04/01/2007' and '06/30/2007') AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q2 Low Attrition],

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date between '04/01/2007' and '06/30/2007') AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q2 Medium Attrition],

SUM(CASE WHEN Start\_Date <= '06/30/2007' and (Termination\_Date between '04/01/2007' and '06/30/2007') AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q2 High Attrition],

-- Q3 Performance Levels

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date>'09/30/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q3 Low],

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date>'09/30/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q3 Medium],

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date>'09/30/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q3 High],

-- Q3 Total Attrition

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date between '07/01/2007' and '09/30/2007') THEN 1 ELSE 0 END) [Q3 Attrition],

-- Q3 Attrition By Levels

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date between '07/01/2007' and '09/30/2007') AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q3 Low Attrition],

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date between '07/01/2007' and '09/30/2007') AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q3 Medium Attrition],

SUM(CASE WHEN Start\_Date <= '09/30/2007' and (Termination\_Date between '07/01/2007' and '09/30/2007') AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q3 High Attrition],

-- Q4 Performance Levels

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date>'12/31/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q4 Low],

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date>'12/31/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q4 Medium],

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date>'12/31/2007' OR Termination\_Date IS NULL) AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q4 High],

-- Q4 Total Attrition

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date between '10/01/2007' and '12/31/2007') THEN 1 ELSE 0 END) [Q4 Attrition],

-- Attrition By Levels

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date between '10/01/2007' and '12/31/2007') AND Performance\_level = 'Low' THEN 1 ELSE 0 END) [Q4 Low Attrition],

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date between '10/01/2007' and '12/31/2007') AND Performance\_level = 'Medium' THEN 1 ELSE 0 END) [Q4 Medium Attrition],

SUM(CASE WHEN Start\_Date <= '12/31/2007' and (Termination\_Date between '10/01/2007' and '12/31/2007') AND Performance\_level = 'High' THEN 1 ELSE 0 END) [Q4 High Attrition]

FROm Employee\_Master

) A

CROSS APPLY(

VALUES ('Q1 2007', [Q1 2007], [Q1 High], [Q1 Medium], [Q1 Low], [Q1 Attrition], [Q1 Low Attrition], [Q1 Medium Attrition], [Q1 High Attrition]),

('Q2 2007', [Q2 2007], [Q2 High], [Q2 Medium], [Q2 Low], [Q2 Attrition], [Q2 Low Attrition], [Q2 Medium Attrition], [Q2 High Attrition]),

('Q3 2007', [Q3 2007], [Q3 High], [Q3 Medium], [Q3 Low], [Q3 Attrition], [Q3 Low Attrition], [Q3 Medium Attrition], [Q3 High Attrition]),

('Q4 2007', [Q4 2007], [Q4 High], [Q4 Medium], [Q4 Low], [Q4 Attrition], [Q4 Low Attrition], [Q4 Medium Attrition], [Q4 High Attrition])

) x (Period, [Total Employees at end of Quarter], [High Performers], [Medium Performers], [Low Performers], [Total Attrition], [Low Performer Attrition], [Medium Performer Attrition], [High Performer Attrition])

Table 3-10: Employee Master

|  |  |  |  |
| --- | --- | --- | --- |
| **Employee ID** | **Start Date** | **Termination Date** | **Performance Level** |
| 1 | 1/1/2007 | 3/1/2007 | Low |
| 2 | 6/5/2004 |  | Medium |
| 3 | 4/3/2003 |  | High |
| 4 | 9/1/2002 | 4/15/2007 | Medium |
| 5 | 4/6/2007 | 11/1/2007 | Low |
| 6 | 7/1/2007 |  | High |
| 7 | 3/2/2005 | 8/1/2007 | Low |

Table 3-11: Employee Attrition

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Period** | **Total Employees at end of quarter** | **High Performers** | **Medium Performers** | **Low Performers** | **Total Attrition** | **Low Performer Attrition** | **Medium Performer Attrition** | **High Performer Attrition** |
| Q1 2007 | 4 | 1 | 2 | 1 | 1 | 1 | 0 | 0 |
| Q2 2007 | 4 | 1 | 1 | 2 | 1 | 0 | 1 | 0 |
| Q3 2007 | 4 | 2 | 1 | 1 | 1 | 1 | 0 | 0 |
| Q4 2007 | 3 | 2 | 1 | 0 | 1 | 1 | 0 | 0 |

**Question 3-4:** Convert a sales hierarchy. Table 3-12 lists the data structure for the sales hierarchy. The table is organized so that the Role is a unique key. Each role is assigned to one person. If the role covers more than one District (or Region or Geo), then those columns are left blank. The parent role defines the hierarchy. The objective is to convert this structure into Table 3-13. In Table 3-13, the Role is again a key, but this time it is augmented with the people who populate the Geo/Region/District for that individual. Note that there may be districts with the same name that are managed by different people (e.g., NY). In this case we would like to distinguish these two districts by adding a count to the district as in the Unique District column of Table 3-13. Also note that one individual may have two different roles (as in the case for Marc Benioff).

**Solution:**

I wasn’t able to under this question in its entirety. I have written a query to get hierarchy of all roles using recursion. If I get some clarifications I should be able to solve it.

;WITH CTE AS

(

SELECT \* ,ROW\_NUMBER()OVER(Order BY (SELECT 1)) flag, 1 level

FROM tbl\_3\_12

WHERE Role in (SELECT Role FROM tbl\_3\_12)

UNION ALL

SELECT e.\*,e2.flag, level + 1

FROM

tbl\_3\_12 e

INNER JOIN CTE e2 ON e.Role = e2.Parent\_Role

)

SELECT

Role,

Person,

level,

flag

FROM CTE

ORDER BY

Flag,

level

Table 3-12:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Role | Parent Role | Person | Geo | Region | District |
| Rep-1 | DM-1 | Lou Gertsner | AME | West | CA |
| Rep-2 | DM-1 | Steve Ballmer | AME | West | CA |
| DM-1 | RVP-1 | Marc Benioff | AME | West | CA |
| RVP-1 | GEO-1 | Ray Ozzie | AME | West |  |
| WW |  | Larry Ellison | AME |  |  |
| GEO-1 | WW | Bill Gates | AME |  |  |
| Rep-3 | DM-1 | Vinod Khosla | AME | West | CA |
| Rep-4 | DM-1 | Marc Benioff | AME | West | CA |
| DM-2 | RVP-2 | Mark Hurd | AME | East | NY |
| RVP-2 | GEO-1 | Mike Ruttgers | AME | East |  |
| Rep-5 | DM-2 | Sergey Brin | AME | East | NY |
| DM-3 | RVP-2 | Larry Page | AME | East | NY |
| Rep-6 | DM-3 | Scott McNealy | AME | East | NY |
| Rep-7 | DM-3 | Sam Palmisano | AME | East | NY |
| Rep-8 | DM-3 | Joe Tucci | AME | East | NY |

Table 3-13:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| WW Person | Geo Person | Region Person | District Person | Role | Person | Geo | Region | District | Unique District |
| Larry Ellison | Bill Gates | Ray Ozzie | Marc Benioff | Rep-1 | Lou Gertsner | AME | West | CA | CA |
| Larry Ellison | Bill Gates | Ray Ozzie | Marc Benioff | Rep-2 | Steve Ballmer | AME | West | CA | CA |
| Larry Ellison | Bill Gates | Ray Ozzie | Marc Benioff | DM-1 | Marc Benioff | AME | West | CA | CA |
| Larry Ellison | Bill Gates | Ray Ozzie |  | RVP-1 | Ray Ozzie | AME | West |  |  |
| Larry Ellison |  |  |  | WW | Larry Ellison |  |  |  |  |
| Larry Ellison | Bill Gates |  |  | GEO-1 | Bill Gates | AME |  |  |  |
| Larry Ellison | Bill Gates | Ray Ozzie | Marc Benioff | Rep-3 | Vinod Khosla | AME | West | CA | CA |
| Larry Ellison | Bill Gates | Ray Ozzie | Marc Benioff | Rep-4 | Marc Benioff | AME | West | CA | CA |
| Larry Ellison | Bill Gates | Mike Ruttgers | Mark Hurd | DM-2 | Mark Hurd | AME | East | NY | NY - 1 |
| Larry Ellison | Bill Gates | Mike Ruttgers |  | RVP-2 | Mike Ruttgers | AME | East |  |  |
| Larry Ellison | Bill Gates | Mike Ruttgers | Mark Hurd | Rep-5 | Sergey Brin | AME | East | NY | NY -1 |
| Larry Ellison | Bill Gates | Mike Ruttgers | Larry Page | DM-3 | Larry Page | AME | East | NY | NY -2 |
| Larry Ellison | Bill Gates | Mike Ruttgers | Larry Page | Rep-6 | Scott McNealy | AME | East | NY | NY -2 |
| Larry Ellison | Bill Gates | Mike Ruttgers | Larry Page | Rep-7 | Sam Palmisano | AME | East | NY | NY -2 |
| Larry Ellison | Bill Gates | Mike Ruttgers | Larry Page | Rep-8 | Joe Tucci | AME | East | NY | NY -2 |

**Question 3-5:** Provide the SQL code for the query(ies) to calculate the total number of tests running each Tuesday from 1/1/2016-2/1/2016. The inputs can be found in table 4-3. Desired result can be found in Table 4-4.

Table 3-14

|  |  |  |  |
| --- | --- | --- | --- |
| id | test\_name | start\_date | end\_date |
| 1 | Test 1 | 1/1/16 | 1/10/16 |
| 2 | Test 2 | 1/1/16 | 1/25/16 |
| 3 | Test 3 | 1/6/16 | 1/10/16 |
| 4 | Test 4 | 1/6/16 | 1/31/16 |
| 5 | Test 5 | 1/11/16 | 1/25/16 |
| 6 | Test 6 | 1/11/16 | 2/1/16 |
| 7 | Test 7 | 1/15/16 | 1/21/16 |
| 8 | Test 8 | 1/19/16 | 1/20/16 |
| 9 | Test 9 | 1/19/16 | 1/22/16 |
| 10 | Test 10 | 1/22/16 | 1/25/16 |
| 11 | Test 11 | 1/22/16 | 1/28/16 |
| 12 | Test 12 | 1/25/16 | 2/1/16 |
| 13 | Test 13 | 1/28/16 | 2/1/16 |

Table 3-15

|  |  |
| --- | --- |
| date | number\_of\_active\_test |
| 1/5/16 | 2 |
| 1/12/16 | 4 |
| 1/19/16 | 7 |
| 1/26/16 | 4 |

**Solution:**

-- Defining a temp table to store all the Tuesdays that

-- fall between '1/1/2016' and '2/1/2016'

CREATE table #tuesday

(

Tues DATE

)

-- Static Parameters Defining Date Range

DECLARE @DateStart AS DATE = '1/1/2016'

DECLARE @DateEnd AS DATE = '2/1/2016'

DECLARE @Date AS DATE

IF @DateStart < @DateEnd -- Check if Date Parameters make sense

BEGIN

SET @Date= @DateStart

WHILE @Date <= @DateEnd

BEGIN

IF DATEPART(dw, @Date) = 3 -- Check If the day is Tuesday

BEGIN

-- If Day is Tuesday Insert into temp table

INSERT INTO #tuesday VALUES (@Date)

SET @Date = DATEADD(dd, 6, @Date)

END

SET @Date = DATEADD(dd,1,@Date)

END

END

-- =======================

-- FINAL RESULT SET

-- =======================

SELECT

t1.Tues, COUNT(t2.Id) [number\_of\_active\_test]

FROM #tuesday t1

INNER JOIN tbl\_4\_3 t2

ON t1.Tues between t2.start\_date and t2.end\_date

GROUP BY

t1.Tues

DROP TABLE #Tuesday

Question 3-6: Provide the SQL code for the query(ies) to calculate for each location how far away in km the nearest neighbor is.

Table 3-16

|  |  |  |  |
| --- | --- | --- | --- |
| location\_id | location\_name | lattitude | longitude |
| 321678 | Chinatown Cafe | 42.347479 | -71.062537 |
| 321983 | City Table | 42.349034 | -71.07955 |
| 482305 | An Tain | 42.35836 | -71.053032 |
| 646829 | Adams Bed & Breakfast | 42.346059 | -71.087909 |
| 777353 | Aisling Bed and Breakfast | 42.337396 | -71.074549 |
| 1954989 | Italian Express Pizzeria | 42.367298 | -71.035713 |
| 1993528 | MET Back Bay | 42.351107 | -71.077586 |
| 6765701 | Bonapita | 42.355417 | -71.058854 |
| 7906434 | Boston Food Tours | 42.363171 | -71.05632 |
| 10027342 | Caffe Bene | 42.342252 | -71.084461 |

Table 3-17

|  |  |  |
| --- | --- | --- |
| location\_id | location\_name | distance\_in\_km\_to\_nearest\_neighbor\_location |
| 321678 | Chinatown Cafe | 0.9331 |
| 321983 | City Table | 0.2814 |
| 482305 | An Tain | 0.5796 |
| 646829 | Adams Bed & Breakfast | 0.5094 |
| 777353 | Aisling Bed and Breakfast | 0.9774 |
| 1954989 | Italian Express Pizzeria | 1.7357 |
| 1993528 | MET Back Bay | 0.2814 |
| 6765701 | Bonapita | 0.5796 |
| 7906434 | Boston Food Tours | 0.5993 |
| 10027342 | Caffe Bene | 0.5094 |

**Solution:**

SELECT

x.Location\_Id,

x.location\_name,

ROUND(MIN(x.MinDist),4) [distance\_in\_km\_to\_nearest\_neighbor\_location]

FROM (

SELECT

B.Location\_id,

B.location\_Name, GEOGRAPHY::Point(B.Lattitude,B.Longitutde,4326).STDistance(GEOGRAPHY::Point(A.Lattitude,A.Longitutde,4326))/1000 MinDist –- Calculate Distance and Divide by 1000 to get in Km

FROM tbl\_3\_16 A, tbl\_3\_16 B -- Cartesian Product As I want to compare each location with every other location

WHERE A.Location\_Name <> B.Location\_name

)x

GROUP BY x.Location\_Id, x.location\_name