1. Write a Function in SQL server to replicate the behavior of TRANSLATE FUNCTION from oracle.

Hint: select dbo. Translate ('challenge', 'aen', 'xyz');

Output: chxllyzgy (replace ‘a’ with ‘x’, replace ‘e’ with ‘y’ and ‘n’ with ‘z’)

USING CTE

;With cte\_tvp

as

(

Select 1 as level,convert(varchar,mainword) as mainword, Originalwords,Replacewords from @t

union all

Select level+1 as level ,

convert(varchar,replace(mainword,substring(Originalwords,level,1),substring(Replacewords,level,1)))

,Originalwords,Replacewords from cte\_tvp where level<=len(Originalwords)

)

Select mainword from cte\_tvp

order by level desc

1. The @tbl is as follows. Please write a Query to show this in date format.

|  |  |  |  |
| --- | --- | --- | --- |
| Yr | Mon | Dy | Dyno |
| 2010 | Jan | Sun | 2 |
| 2005 | Jan | Mon | 3 |
| 1995 | Feb | Sun | 1 |
| 2000 | Feb | Wed | 4 |
| 1982 | Mar | Tue | 2 |
| 2010 | Mar | Tue | 8 |



INPUT SCRIPT

DECLARE @tbl TABLE (Yr INT,Mon VARCHAR(50),Dy VARCHAR(50),Dyno INT)

INSERT INTO @tbl(Yr, Mon, Dy, Dyno)

SELECT 2010,'Jan','Sun',2 UNION ALL

SELECT 2005,'Jan','Mon',3 UNION ALL

SELECT 1995,'Feb','Sun',1 UNION ALL

SELECT 2000,'Feb','Wed',4 UNION ALL

SELECT 1982,'Mar','Tue',2 UNION ALL

SELECT 2010,'Mar','Tue',8

SELECT \* FROM @tbl

Output:-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Yr | Mon | Dy | Dyno | Date |
| 1982 | Mar | Tue | 2 | 3/9/1982 |
| 1995 | Feb | Sun | 1 | 2/5/1995 |
| 2000 | Feb | Wed | 4 | 2/23/2000 |
| 2005 | Jan | Mon | 3 | 1/17/2005 |
| 2010 | Jan | Sun | 2 | 1/10/2010 |
| 2010 | Mar | Tue | 8 | Invalid Date |

-- The solution

;with DaysOfWeek as

(Select

-- Get the day number for the dy field for each record

case dy when 'Sun' then 1 when 'Mon' then 2 when 'Tue' then 3 when 'Wed' then 4 when 'Thu'

then 5 when 'Fri' then 6 when 'Sat' then 7 end as DayofWeek,

-- get the day number for the first day of the month for each record

datepart(weekday,convert(DATETIME, Mon + ' 01 ' + convert(varchar(4), yr))) as firstdayofmonth ,

-- get the date for the first day of the month for each record

convert(DATETIME, Mon + ' 01 ' + convert(varchar(4), yr)) AS DateOfFirstDayOfMonth,

\* From @tbl)

,

tmp as (Select Case

-- Get the first 'dy' in the month

When DayOfWeek = firstdayofmonth then DateOfFirstDayOfMonth

When DayOfWeek > firstdayofmonth then dateadd (d ,dayofweek-firstdayofmonth, DateOfFirstDayOfMonth)

else

dateadd (d ,dayofweek +7 - firstdayofmonth, DateOfFirstDayOfMonth)

end as FirstOccurrenceOfDay, \* from DaysOfWeek),

tmp1 as (Select

case

-- Add (dyno-1)\* 7 days to the FirstOccurrenceOfDay and check that the month of that date is 'mon'

when left(datename(m, dateadd(d, (dyno-1)\* 7, FirstOccurrenceOfDay)),3) =Mon Then

convert(varchar(10),dateadd(d, (dyno-1)\* 7, FirstOccurrenceOfDay),20)

Else 'Invalid Date'

End as [Date],\* from tmp)

select yr,Mon,dy,dyno,[date] from tmp1 order by yr

The table Employee is in the below format. (Use Ranking Functions)

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | Department | Salary |
| 1 | T Cook | Finance | 40000 |
| 2 | D Michael | Finance | 25000 |
| 3 | A Smith | Finance | 25000 |
| 4 | D Adams | Finance | 15000 |
| 5 | M Williams | IT | 80000 |
| 6 | D Jones | IT | 40000 |
| 7 | J Miller | IT | 50000 |
| 8 | L Lewis | IT | 50000 |
| 9 | A Anderson | Back-Office | 25000 |
| 10 | S Martin | Back-Office | 15000 |
| 11 | J Garcia | Back-Office | 15000 |
| 12 | T Clerk | Back-Office | 10000 |

Please find the Second Highest salary person from this table department wise ordered by salary. Please don’t use any control structures. (Note: - if a two persons have the second highest salaries please show both of them)

Output:-

|  |  |  |  |
| --- | --- | --- | --- |
| EmployeeID | EmployeeName | Department | Salary |
| 10 | S Martin | Back-Office | 15000 |
| 11 | J Garcia | Back-Office | 15000 |
| 2 | D Michael | Finance | 25000 |
| 3 | A Smith | Finance | 25000 |
| 7 | J Miller | IT | 50000 |
| 8 | L Lewis | IT | 50000 |

1. USING RECURSIVE CTE

Client: - I have a problem with our application so there are lots of modifications we need to do for our database. Currently, my database is not completely normalized. A sample data is like

case # code  
200     A,B,C  
300     D,E

So now we want to use to normalize. Consider that I want it in 2008 version. Are there any new features that help us with better performance?

Output:-

case# code  
200     A  
200     B  
200     C  
300     D  
300     E

;With cte\_tvp

as

(

Select 1 as level,case#,code from @t

union all

Select level+1 as level ,case#,code from cte\_tvp where level<len(replace(code,',',''))

)

Select case#,substring(replace(code,',',''),level,1) as code from cte\_tvp

order by case#

Please help the developer in accomplishing this task.

1. Write at least two possible ways (if possible more) to return zero without using numbers?
2. What is the output of

Select $

Union

Select £

1. Alice (Client):- Chris Can you help me in finding the Errors that we were encountering in data loading. There is a table named Final\_Employee. And it is being loaded from the employee table every day End of the day. Any new employees need to be added. And modification of the data in employee table needs to be updated.

* Client: - “We are managing a retail system that needs some consolidation functionalities of the newly introduced products. Current application stores this data in a table with respect to the saleID and if the product1 was bought it is “1” else “0”. Similarly with product2.

The Challenge   
The challenge is to build the ranges of sequential values. The table with these ranges will have the following properties:

* The first customer identifier of the range
* The last customer identifier of the range
* The product1 status
* The product2 status

The ranges of the customer to take will be selected between two dates. The table has an IDENTITY column to maintain the sequence.

The table structure is like this.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SaleID | CreationDate | Custid | Product1 | Procuct2 |
| 1 | 5/27/09 10:15 PM | Cust #1 | 0 | 0 |
| 2 | 5/28/09 12:39 AM | Cust #2 | 0 | 0 |
| 3 | 5/28/09 3:03 AM | Cust #3 | 1 | 1 |
| 4 | 5/28/09 5:27 AM | Cust #4 | 1 | 1 |
| 5 | 5/28/09 7:51 AM | Cust #5 | 1 | 1 |
| 6 | 5/28/09 10:15 AM | Cust #6 | 1 | 0 |
| 7 | 5/28/09 12:39 PM | Cust #7 | 1 | 0 |
| 8 | 5/28/09 3:03 PM | Cust #8 | 1 | 0 |
| 9 | 5/28/09 5:27 PM | Cust #9 | 1 | 0 |
| 10 | 5/28/09 7:51 PM | Cust #10 | 1 | 1 |

Output:-

|  |  |  |  |
| --- | --- | --- | --- |
| FirstSaleIdInclusive | LastSaleIdInclusive | Product1 | Procuct2 |
| 1 | 2 | 0 | 0 |
| 3 | 5 | 1 | 1 |
| 6 | 9 | 1 | 0 |
| 10 | 10 | 1 | 1 |

Please do even consider the performance of the Query and avoid Cursors and Loops.

--populate sample data

DECLARE @tc9 TABLE(

saleID INT IDENTITY(1,1),

CreationDate DATETIME,

Custid NVARCHAR(10),

Product1 BIT,

Product2 BIT

)

INSERT INTO @tc9 (CreationDate, Custid, Product1, Product2)

SELECT GETDATE()-1.0,'Cust #1',0,0 UNION

SELECT GETDATE()-0.9,'Cust #2',0,0 UNION

SELECT GETDATE()-0.8,'Cust #3',1,1 UNION

SELECT GETDATE()-0.7,'Cust #4',1,1 UNION

SELECT GETDATE()-0.6,'Cust #5',1,1 UNION

SELECT GETDATE()-0.5,'Cust #6',1,0 UNION

SELECT GETDATE()-0.4,'Cust #7',1,0 UNION

SELECT GETDATE()-0.3,'Cust #8',1,0 UNION

SELECT GETDATE()-0.2,'Cust #9',1,0 UNION

SELECT GETDATE()-0.1,'Cust #10',1,1

--SELECT \* FROM @tc9

--solution

Declare @startTime datetime

Declare @endTime datetime

set @startTime = GetDate()-20.8

set @endTime = GetDate()

;With MessageGroups

as

(

Select

\*,

-- assign a unique group number for each

-- consequtive state combination

ID - ROW\_NUMBER()

OVER(PARTITION BY Product1, Product2

ORDER BY saleID) AS GroupID

From @tc9

Where CreationDate between @startTime and @endTime

)

Select MIN(saleID) as FirstSaleIdInclusive,

MAX(saleID) as LastSaleIdInclusive,

Product1, Product2

From MessageGroups

Group by GroupID, SendState, Product2

Order by MIN(saleID)

DBA: - Currently I have huge log files to analyze. But management expects a report from the DBA that will show

SET DATEFIRST 1

SET DATEFORMAT YMD

SET NOCOUNT ON

Create Table TLogs (ID INT IDENTITY(1,1), LogDate DATETIME, ErrorLayer NVARCHAR(15),TLogsCount INT)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-23','Database',10)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-24','Database',14)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-26','Database',22)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-27','Database',3)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-29','Database',4)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-04-01','Database',33)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-04-03','Database',2)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-26','Server',22)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-27','Server',10)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-29','Server',35)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-30','Server',13)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-27','Misc',8)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-03-29','Misc',12)

INSERT INTO TLogs (LogDate,ErrorLayer,TLogsCount) VALUES ('2009-04-01','Misc',16)

Select \* from TLogs

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **LogDate** | **ErrorLayer** | **TLogsCount** |
| 1 | 3/23/2009 0:00 | Database | 10 |
| 2 | 3/24/2009 0:00 | Database | 14 |
| 3 | 3/26/2009 0:00 | Database | 22 |
| 4 | 3/27/2009 0:00 | Database | 3 |
| 5 | 3/29/2009 0:00 | Database | 4 |
| 6 | 4/1/2009 0:00 | Database | 33 |
| 7 | 4/3/2009 0:00 | Database | 2 |
| 8 | 3/26/2009 0:00 | Server | 22 |
| 9 | 3/27/2009 0:00 | Server | 10 |
| 10 | 3/29/2009 0:00 | Server | 35 |
| 11 | 3/30/2009 0:00 | Server | 13 |
| 12 | 3/27/2009 0:00 | Misc | 8 |
| 13 | 3/29/2009 0:00 | Misc | 12 |
| 14 | 4/1/2009 0:00 | Misc | 16 |

SELECT A.[y] + Isnull(A.[w], '') + Isnull(A.[d], 'T') AS 'PERIOD\_ID',

Grouping(A.[w]) + Grouping(A.[d]) AS 'LEVEL',

--COALESCE(Substring('SATSUNMONTUEWEDTHUFRI', ((A.[d] + @@DATEFIRST) % 7) \* 3 + 1, 3)

-- + ' ' + Min(A.[dm]), 'TOTAL WEEK ' + A.[w], 'TOTAL YEAR ' + A.[y]) AS 'PERIOD',

Sum(A.[h]) AS 'Server',

Sum(A.[c]) AS 'Database',

Sum(A.[p]) AS 'Misc'

FROM (SELECT Datename(year, [LogDate]) AS 'Y',

Datename(week, [LogDate]) AS 'W',

Cast(Datepart(weekday, [LogDate]) AS VARCHAR(2)) AS 'D',

Datename(day, [LogDate]) + '/'

+ Cast(Month([LogDate]) AS VARCHAR(2)) AS 'DM',

CASE

WHEN [ErrorLayer] = 'Server' THEN [TlogsCount]

ELSE 0

END AS 'H',

CASE

WHEN [ErrorLayer] = 'Database' THEN [TlogsCount]

ELSE 0

END AS 'C',

CASE

WHEN [ErrorLayer] = 'Misc' THEN [TlogsCount]

ELSE 0

END AS 'P'

FROM Tlogs) AS A

GROUP BY A.[y],

A.[w],

A.[d] WITH rollup

HAVING Grouping(A.[y]) = 0

--ORDER BY period\_id