Recursive Common Table Expressions in Oracle 11gR2

Iggy Fernandez
RMOUG Training Days 2010
Beginning Oracle Database 11g Administration:
From Novice to Professional

Iggy Fernandez
CTE Recap
Inline Views

```
SELECT *
FROM (SELECT *
FROM Suppliers
MINUS
SELECT *
FROM (SELECT SupplierName
FROM (SELECT *
FROM (SELECT *
FROM Suppliers, Parts)
MINUS
SELECT *
FROM (SELECT SupplierName, PartName
FROM Quotes))));
```
CTE Recap
All Supplier Part Pairs

WITH

AllSupplierPartPairs AS

(  SELECT *
    FROM Suppliers, Parts
  ),
CTE Recap
Valid Supplier Part Pairs

ValidSupplierPartPairs AS
 (  
    SELECT SupplierName, PartName 
    FROM Quotes 
  ),
CTE Recap
Invalid Supplier Part Pairs

InvalidSupplierPartPairs AS

  (  
    SELECT *  
    FROM AllSupplierPartPairs  
    MINUS  
    SELECT *  
    FROM ValidSupplierPartPairs  
  )
CTE Recap
Suppliers Who Don’t Supply All Parts

SuppliersWhoDontSupplyAllParts AS
(
    SELECT SupplierName
    FROM InvalidSupplierPartPairs
),
CTE Recap
Suppliers Who Supply All Parts

SuppliersWhoSupplyAllParts AS

(  
  SELECT *  
  FROM Suppliers  
  MINUS  
  SELECT *  
  FROM SuppliersWhoDontSupplyAllParts  
)

SuppliersWhoDontSupplyAllParts
CTE Recap

Suppliers Who Supply All Parts

SELECT *
FROM SuppliersWhoSupplyAllParts;
Traditional Hierarchical Queries
Managers and Employees

```
SELECT
    LPAD (' ', 4 * (LEVEL - 1)) || first_name || ' ' ||
    last_name AS name
FROM employees
START WITH manager_id IS NULL
CONNECT BY manager_id = PRIOR employee_id;
```
Traditional Hierarchical Queries
Managers and Employees

Name
----------------------------------------------------------

Steven King
Neena Kochhar
Nancy Greenberg
Daniel Faviet
John Chen
Ismael Sciarra
Jose Manuel Urman
Luis Popp
Jennifer Whalen
Susan Mavris
Hermann Baer
Shelley Higgins
William Gietz
Traditional Hierarchical Queries
Managers and Employees

WITH

RCTE (employee_id, first_name, last_name, lvl) AS
(

SELECT
  employee_id,
  first_name,
  last_name,
  1 AS lvl
FROM
  employees
WHERE manager_id IS NULL

)
Traditional Hierarchical Queries
Managers and Employees

```
UNION ALL

SELECT
    e.employee_id,
    e.first_name,
    e.last_name,
    lvl + 1 AS lvl
FROM
    RCTE INNER JOIN employees e
    ON (RCTE.employee_id = e.manager_id)
)
-- SEARCH DEPTH FIRST BY employee_id ASC SET seq#
```
Traditional Hierarchical Queries
Managers and Employees

```
SELECT LPAD (' ', 4 * (lvl - 1)) || first_name || ' ' || last_name AS name
FROM RCTE
--ORDER BY seq#;
```
Traditional Hierarchical Queries
Breadth First Search

Steven King
Michael Hartstein
Neena Kochhar
Lex De Haan
Den Raphaely
Matthew Weiss
Adam Fripp
Payam Kaufling
Shanta Vollman
Kevin Mourgos
John Russell
Karen Partners
Alberto Errazuriz
Gerald Cambrault
Eleni Zlotkey
Pat Fay
Jennifer Whalen
Algorithm

1. Split the CTE expression into anchor and recursive members.
2. Run the anchor member(s) creating the first invocation or base result set (T0).
3. Run the recursive member(s) with Ti as an input and Ti+1 as an output.
4. Repeat step 3 until an empty set is returned.
5. Return the result set. This is a UNION ALL of T0 to Tn.
Number Generator
New Style

WITH numbers(n) AS
(
    SELECT 1 FROM dual
    UNION ALL
    SELECT n + 1 FROM numbers WHERE n < 100
)
SELECT * FROM numbers;
Number Generator
Old Style

```
SELECT level AS n
FROM dual
CONNECT BY level <= 100;
```
Railroad Diagram
Restrictions

The recursive member cannot contain any of the following elements:

- The DISTINCT keyword or a GROUP BY clause
- The model Clause
- An aggregate function. However, analytic functions are permitted in the select list.
- Subqueries that refer to the recursive member.
- Outer joins that refer to recursive member as the right table.
Other Goodies

- SYS_CONNECT_BY_PATH
- CONNECT_BY_ROOT
- CONNECT_BY_CYCLE
- CONNECT_BY_ISLEAF
- ORDER SIBLINGS BY
# Managers and Employees

<table>
<thead>
<tr>
<th>EMPLOYEE_ID</th>
<th>NAME</th>
<th>SALARY</th>
<th>LVL</th>
<th>EMPLOYEES</th>
<th>TOTAL_SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Steven King</td>
<td>24000</td>
<td>3</td>
<td>106</td>
<td>667416</td>
</tr>
<tr>
<td>101</td>
<td>Neena Kochhar</td>
<td>17000</td>
<td>2</td>
<td>11</td>
<td>92816</td>
</tr>
<tr>
<td>148</td>
<td>Gerald Cambrault</td>
<td>11000</td>
<td>1</td>
<td>6</td>
<td>51900</td>
</tr>
<tr>
<td>145</td>
<td>John Russell</td>
<td>14000</td>
<td>1</td>
<td>6</td>
<td>51000</td>
</tr>
<tr>
<td>146</td>
<td>Karen Partners</td>
<td>13500</td>
<td>1</td>
<td>6</td>
<td>51000</td>
</tr>
<tr>
<td>149</td>
<td>Eleni Zlotkey</td>
<td>10500</td>
<td>1</td>
<td>6</td>
<td>50000</td>
</tr>
<tr>
<td>147</td>
<td>Alberto Errazuriz</td>
<td>12000</td>
<td>1</td>
<td>6</td>
<td>46600</td>
</tr>
<tr>
<td>108</td>
<td>Nancy Greenberg</td>
<td>12008</td>
<td>1</td>
<td>5</td>
<td>39600</td>
</tr>
<tr>
<td>102</td>
<td>Lex De Haan</td>
<td>17000</td>
<td>2</td>
<td>5</td>
<td>28800</td>
</tr>
<tr>
<td>123</td>
<td>Shanta Vollman</td>
<td>6500</td>
<td>1</td>
<td>8</td>
<td>25900</td>
</tr>
<tr>
<td>121</td>
<td>Adam Fripp</td>
<td>8200</td>
<td>1</td>
<td>8</td>
<td>25400</td>
</tr>
<tr>
<td>122</td>
<td>Payam Kaufling</td>
<td>7900</td>
<td>1</td>
<td>8</td>
<td>23600</td>
</tr>
<tr>
<td>124</td>
<td>Kevin Mourgos</td>
<td>5800</td>
<td>1</td>
<td>8</td>
<td>23000</td>
</tr>
<tr>
<td>120</td>
<td>Matthew Weiss</td>
<td>8000</td>
<td>1</td>
<td>8</td>
<td>22100</td>
</tr>
<tr>
<td>103</td>
<td>Alexander Hunold</td>
<td>9000</td>
<td>1</td>
<td>4</td>
<td>19800</td>
</tr>
<tr>
<td>114</td>
<td>Den Raphaely</td>
<td>11000</td>
<td>1</td>
<td>5</td>
<td>13900</td>
</tr>
<tr>
<td>205</td>
<td>Shelley Higgins</td>
<td>12008</td>
<td>1</td>
<td>1</td>
<td>8300</td>
</tr>
<tr>
<td>201</td>
<td>Michael Hartstein</td>
<td>13000</td>
<td>1</td>
<td>1</td>
<td>6000</td>
</tr>
</tbody>
</table>
Managers and Employees

WITH CTE (employee_id, last_name, first_name, lvl, manager_salary, employee_salary, manager_id) AS
(

SELECT employee_id, last_name, first_name, 0, salary, salary, manager_id
FROM employees

UNION ALL

SELECT e.employee_id, e.last_name, e.first_name, c.lvl+1, e.salary, c.employee_salary, e.manager_id
FROM employees e INNER JOIN CTE c ON (e.employee_id = c.manager_id)
)

Iggy Fernandez
Database Specialists
Managers and Employees

SELECT
    employee_id,
    first_name||' '||last_name AS name,
    manager_salary AS salary,
    max(lvl) AS lvl,
    count(*) - 1 AS employees,
    sum(employee_salary) - manager_salary AS total_salary
FROM CTE
GROUP BY employee_id, last_name, first_name,
        manager_salary
HAVING max(lvl) > 0
ORDER BY 6 desc, 1;
Coupon Clipping

Given a list of products and a list of discount coupons, we needed to find the minimum price for all the products based on certain rules. Here are those rules:

- A maximum of ten coupons can be applied on the same product.
- The discount price cannot be less than 70% of the original price.
- The total amount of the discount cannot exceed 30$.
## Coupon Clipping

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>Discounted Price</th>
<th>Discounted Amount</th>
<th>Rate Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PROD 1</td>
<td>100.00</td>
<td>72.00</td>
<td>28.00 CP 1 : -15$ + CP 2 : -5$ + CP 3 : -10%</td>
</tr>
<tr>
<td>2</td>
<td>PROD 2</td>
<td>220.00</td>
<td>193.00</td>
<td>12.27 CP 1 : -15$ + CP 4 : -12$</td>
</tr>
<tr>
<td>3</td>
<td>PROD 3</td>
<td>15.00</td>
<td>13.50</td>
<td>10.00 CP 3 : -10%</td>
</tr>
<tr>
<td>4</td>
<td>PROD 4</td>
<td>70.00</td>
<td>49.50</td>
<td>29.29 CP 1 : -15$ + CP 3 : -10%</td>
</tr>
<tr>
<td>5</td>
<td>PROD 5</td>
<td>150.00</td>
<td>121.50</td>
<td>19.00 CP 1 : -15$ + CP 3 : -10%</td>
</tr>
</tbody>
</table>
Coupon Clipping

CREATE TABLE products (ID INTEGER PRIMARY KEY, Name VARCHAR2(20), Price NUMBER);

INSERT INTO products VALUES (1, 'PROD 1', 100);
INSERT INTO products VALUES (2, 'PROD 2', 220);
INSERT INTO products VALUES (3, 'PROD 3', 15);
INSERT INTO products VALUES (4, 'PROD 4', 70);
INSERT INTO products VALUES (5, 'PROD 5', 150);

CREATE TABLE coupons (ID INTEGER PRIMARY KEY, Name VARCHAR2(20), Value INTEGER, IsPercent CHAR(1));

INSERT INTO coupons VALUES (1, 'CP 1 : -15$', 15, 'N');
INSERT INTO coupons VALUES (2, 'CP 2 : -5$', 5, 'N');
INSERT INTO coupons VALUES (3, 'CP 3 : -10%', 10, 'Y');
INSERT INTO coupons VALUES (4, 'CP 4 : -12$', 12, 'N');
Coupon Clipping

WITH

RCTE(ID, Name, Price, DiscountedPrice, DiscountAmount, DiscountRate, CouponNames, CouponCount, CouponId) AS

(SELECT
    ID,
    Name,
    Price,
    Price AS DiscountedPrice,
    0 AS DiscountAmount,
    0 AS DiscountRate,
    CAST(' ' AS VARCHAR2(1024)) AS CouponNames,
    0 AS CouponCount,
    -1 AS CouponId
FROM
    products)
Coupon Clipping

UNION ALL

SELECT
    RCTE.ID, RCTE.Name, RCTE.Price,
    DECODE(C.IsPercent, 'N', RCTE.DiscountedPrice - C.Value, 
    RCTE.DiscountedPrice - (RCTE.DiscountedPrice / 100 * 
    C.Value)) DiscountedPrice,
    RCTE.Price - DiscountedPrice AS DiscountAmount,
    (RCTE.Price - DiscountedPrice) / RCTE.Price * 100 AS DiscountRate,
    DECODE(RCTE.CouponNames, ' ', C.Name, RCTE.CouponNames 
    || ' + ' || C.Name) AS CouponNames,
    RCTE.CouponCount + 1 AS CouponCount,
    C.ID AS CouponID
FROM RCTE, coupons C
WHERE
    c.id > RCTE.CouponID AND CouponCount <= 2 AND 
    DiscountAmount <= 30 AND DiscountRate <= 30
Coupon Clipping

SortedPrices AS
((
  SELECT
    RCTE.*,
    RANK() OVER (PARTITION BY ID ORDER BY DiscountedPrice) AS Rank
  FROM RCTE
)

SELECT
  ID, Name, Price,
  DiscountedPrice, DiscountAmount, DiscountRate,
  CouponNames
FROM SortedPrices
WHERE Rank = 1
ORDER BY ID;
Sudoku

with x( s, ind ) as
(
  select sud, instr( sud, '' )
from (select '53  7   6 195  98  6 8  6 34  8 3  17   2 6 6  28  419  5  8 79' sud
from dual ) union all
select substr( s, 1, ind - 1 ) || z || substr( s, ind + 1 ), instr( s, '', ind + 1 )
from x, (select to_char( rownum ) z from dual connect by rownum <= 9) z
where ind > 0
and not exists
(
  select *
  from (select rownum lp from dual connect by rownum <= 9)
  where z = substr(s,trunc((ind-1)/9 )*9+lp,1)
  or z = substr(s,mod(ind-1,9)-8+lp*9,1)
  or z = substr(s,mod( trunc(( ind-1)/3),3)*3+ trunc((ind-1)/27)*27+lp+trunc((lp-1)/3)*6,1))
)

select s from x
Thanks For Listening

iggy_fernandez@hotmail.com
iggyfernandez.wordpress.com

Please submit evaluation forms