



Расширенные возможности языка SQL

Часть 3. Иерархические и рекурсивные запросы

Борисенков Д.В.
НПП "РЕЛЭКС"
Кафедра МО ЭВМ ВГУ

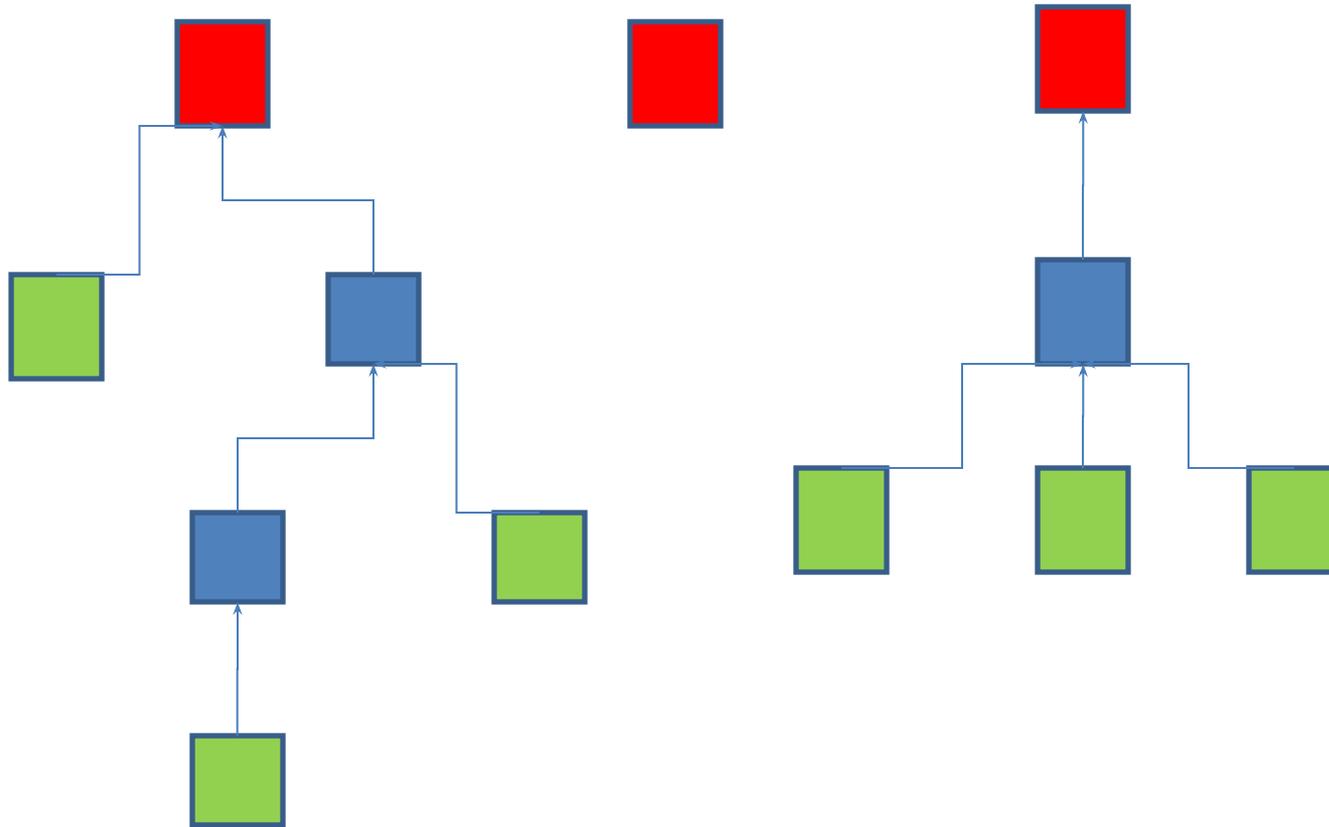
Программа обучения практикантов и сотрудников компании в области современных информационных технологий, методологий управлений проектами в области ИТ

Основные цели:

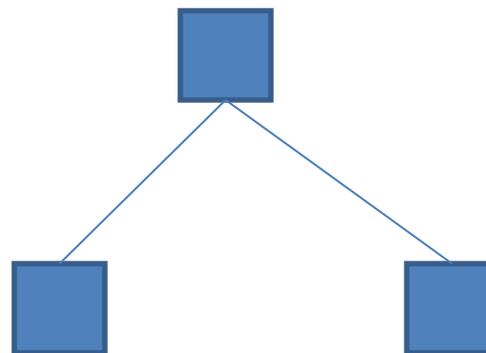
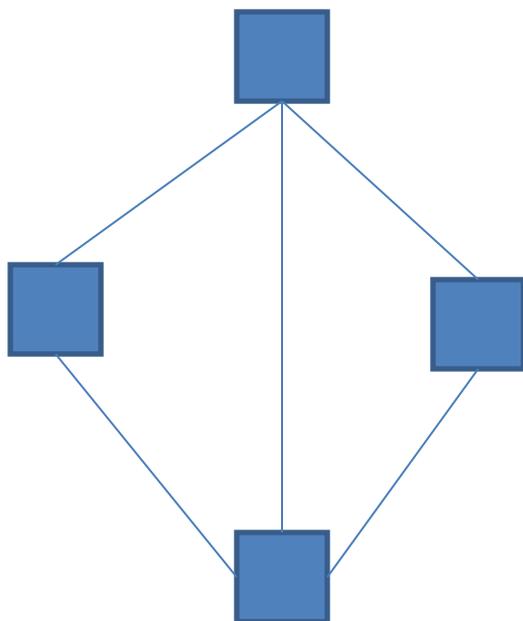
- подготовка специалистов к участию в реальных проектах
- обучение и повышение квалификации по современным информационным технологиям
- обмен опытом между сотрудниками компании

- Иерархия обычно рассматривается как отношение подчиненности типа “один ко многим” между сущностями одного типа. Иерархическая связь между экземплярами сущности является направленной
- При этом если экземпляр В сущности подчинен (непосредственно или транзитивно) экземпляру А этой сущности, то экземпляр В не может быть подчинен (непосредственно или транзитивно) экземпляру А
- Верхний уровень иерархии – сущности, которые не подчинены никому, нижний – сущности, которые не имеют подчиненных
- Длина цепочки подчинений для разных сущностей нижнего уровня может быть различной
- Всю совокупность сущностей, входящих в иерархию, вместе с их связями, обычно представляют в виде перевернутого дерева (если сущность верхнего уровня иерархии одна), либо совокупности таких деревьев

- Иерархия подчинения должностей в организации “начальник – подчиненный”
- Генеалогическое дерево (простой вариант “отец – сын”)
- Иерархия файлов и каталогов на носителе информации
- Иерархия типа “часть – целое” (например “организация-подразделение” или “изделие – деталь”)
- Иерархия типа “частное-общее” (иерархия классов в программе ООП при одиночном наследовании; систематика органического мира – классификация Линнея)



- Рекурсию можно рассматривать как отношение “многие ко многим” между сущностями одного типа. Отношение “один ко многим” есть частный случай отношения “многие ко многим”
- Например, населенные пункты, связанные между собой транспортными маршрутами или люди, связанные между собой в социальной сети
- Рекурсивные связи могут быть как направленными, так и ненаправленными (симметричными)
- Вместо дерева в случае рекурсии мы имеем граф обобщенного типа



- Представить сущности вместе с их иерархическими или рекурсивными связями в реляционной базе данных
- Получить одним запросом информацию обо всех сущностях, входящие в иерархию, или о какой-то их части (например, обо всех сущностях, подчиненных непосредственно или транзитивно одной сущности верхнего уровня)
- Получить информацию о сущностях, входящих в граф сущностей, соединенных рекурсивными связями, или какую-то его часть



- Фиксированное и небольшое количество уровней иерархии
- Одна таблица на каждый уровень иерархии
- Между каждой парой соседних уровней иерархии организуется связь типа “первичный ключ” – “внешний ключ”
- Доступ к данным организуется с помощью обычных SQL-запросов, без использования специального синтаксиса

```
CREATE TABLE university /* ВУЗ */ (  
    id INT PRIMARY KEY,  
    name VARCHAR (100),  
    ... );
```

```
CREATE TABLE department /* Факультет */ (  
    id INT PRIMARY KEY,  
    university_id INT REFERENCES university,  
    name VARCHAR (100),  
    ... );
```

```
CREATE TABLE chair /*Кафедра */ (  
    id INT PRIMARY KEY,  
    department_id INT REFERENCES department,  
    name VARCHAR (100),  
    ... );
```

Задание: выдать названия всех существующих кафедр, для каждой кафедры указав также названия факультета и ВУЗа

SELECT

chair.name,
department.name,
university.name

FROM

university,
department,
chair

WHERE

university.id = department.university_id AND
department.id = chair.department_id;

- Количество уровней иерархии не известно заранее
- Количество уровней иерархии не является постоянным
- Для небольшого количества уровней запрос может быть организован без использования специальных средств - с помощью внешних соединений. При этом каждому возможному уровню иерархии соответствует один экземпляр таблицы-источника



```
CREATE TABLE organization (  
  id INT PRIMARY KEY,  
  parent_id INT REFERENCES organization,  
  name VARCHAR (100)  
  ...);
```

Одна таблица для хранения всех уровней иерархии (вместо отдельной таблицы для каждого уровня)

Рекурсивная связь “первичный ключ” - “внешний ключ” таблицы с самой этой же таблицей, одна для всех уровней иерархии

Значение поля parent_id равно NULL для записей верхнего уровня иерархии

Задание: выдать названия всех существующих организаций/подразделений, для каждой из них также указав названия всех вышестоящих организаций/подразделений

```
SELECT
  o1.name || ' ' ||
  nvl (o2.name, '') || ' ' ||
  nvl (o3.name, '') || ' ' ||
  nvl (o4.name, '')
FROM
  organization o1
  LEFT JOIN organization o2 ON o1.parent_id = o2.id
  LEFT JOIN organization o3 ON o2.parent_id = o3.id
  LEFT JOIN organization o4 ON o3.parent_id = o4.id;
```

Один экземпляр таблицы-источника на каждый уровень иерархии
Внешнее соединение для случая, когда число уровней меньше
максимального



- Все приведенные запросы не содержали рекурсии (т.е. связь между каждой парой соседних уровней иерархии была представлена отдельным условием)
- Запросы для большого количества уровней иерархии требуют рекурсии (т.е. единого представления для всех связей между парами соседних уровней иерархии)
- Нужно также отдельное условие для отбора записей самого верхнего уровня иерархии

- Стандарт SQL-92 не содержал средств для создания иерархических запросов
- **СУБД Oracle** первой реализовала поддержку иерархических запросов (синтаксис с ключевыми словами **START WITH** и **CONNECT BY**)
- В стандарте **SQL-99** появились рекурсивные запросы на основе **рекурсивных CTE** (общих табличных выражений – синтаксис с ключевым словом **WITH**) с аналогичной функциональностью
- **MS SQL** поддерживает рекурсивные запросы (синтаксис стандарта SQL) начиная с версии **2005**
- **ЛИНТЕР** частично поддерживает синтаксис иерархических запросов СУБД Oracle (с **CONNECT BY**)

```
SELECT
```

```
  name,
```

```
  LEVEL /* уровень записи в иерархии */
```

```
FROM
```

```
  organization /* таблица-источник в иерархическом запросе всегда одна */
```

```
START WITH /* условие для отбора записей начального уровня иерархии */
```

```
  parent_id IS NULL
```

```
CONNECT BY /* условие соединения между соседними уровнями иерархии */
```

```
  PRIOR id = parent_id; /* Префикс PRIOR помечает значение, относящееся */
```

```
    /* к предыдущему уровню иерархии */
```

Без префикса **PRIOR** невозможно соединить разные уровни иерархии.

Если условие **START WITH** опущено, то каждая строка считается корневой в иерархии (при этом ответ будет содержать дубликаты)

Порядок следования записей, возвращаемых таким запросом, зависит от их положения в выбираемой иерархии записей

Каждое дерево записей, подчиняющееся в этой иерархии некоторой



NAME	LEVEL	
ВГУ	1	Первая – всегда запись самого верхнего уровня
ПММ	2	За записью для ВГУ идут все подчиненные ей записи - для всех его подразделений
МО ЭВМ	3	За записью для факультета ПММ идут все подчиненные ей записи -
...		
ММИО	3	для всех кафедр факультета ПММ
Матфак	2	Дальше идет следующий факультет ВГУ
КМА	3	и за ним все его кафедры, и т.д.
...		
КУЧП	3	
...		
ВГТУ	1	После того, как закончились все подразделения
ФАЭМ	2	ВГУ, идет следующий ВУЗ со всеми его
АВС	3	факультетами и кафедрами, и т.д.
...		

- Если использовать для сортировки результатов иерархических запросов стандартную конструкцию ORDER BY, то иерархический порядок нарушится - в полученном результате подчиненные записи уже не будут следовать за соответствующей записью верхнего уровня
- Поэтому для сортировки результатов иерархических запросов в СУБД Oracle есть специальная конструкция – **ORDER SIBLINGS BY**
- Ключевое слово SIBLINGS означает, что сортировка ведется только между записями одного уровня (и если уровень больше 1, то сортируемые записи должны быть подчинены одной и той же записи предыдущего уровня)

```
SELECT name, LEVEL FROM organization
START WITH parent_id IS NULL CONNECT BY PRIOR id = parent_id
ORDER SIBLINGS BY name;
```

NAME	LEVEL	
...		
ВГТУ	1	ВГТУ со всеми своими подразделениями - переместился перед ВГУ (по алфавиту)
...		
ВГУ	1	Все подразделения ВГУ переместились после сортировки вместе с ним
...		
Матфак	2	Матфак – впереди ПММ
...		
ПММ	2	ПММ переместился со всеми своими кафедрами
...		
ММИО	3	ММИО – впереди МО ЭВМ
МО ЭВМ	3	и т.д.
...		

LEVEL – псевдостолбец, значение которого равно уровню соответствующей строки. Уровни начинаются с 1

```
SELECT name, LEVEL  
FROM organization START WITH parent_id IS NULL  
CONNECT BY PRIOR id = parent_id; /* ВГУ 1 ; ПММ 2 ; МО ЭВМ 3 */
```

Для вывода разных уровней иерархии с разными отступами обычно применяются выражения типа **SELECT LPAD(' ', (LEVEL-1)*3, ' ') || name**

CONNECT_BY_ISLEAF – псевдостолбец, значение равно 1 для строк, у которых нет подчиненных, и 0 для остальных

```
SELECT name, CONNECT_BY_ISLEAF  
FROM organization START WITH parent_id IS NULL  
CONNECT BY PRIOR id = parent_id; /* ВГУ 0 ; ПММ 0 ; МО ЭВМ 1 */
```



CONNECT_BY_ROOT – префикс, который указывает, что значение столбца берется не из текущей записи, а из корневой записи иерархии

```
SELECT name || '(' || CONNECT_BY_ROOT name || ')'
FROM organization START WITH parent_id IS NULL
CONNECT BY PRIOR id = parent_id; /* ВГУ (ВГУ) ; ПММ (ВГУ) ; МО ЭВМ
(ВГУ) */
```

SYS_CONNECT_BY_PATH – агрегатная функция, выдающая конкатенацию указанного значения для всех уровней иерархии с указанными разделителями

```
SELECT SYS_CONNECT_BY_PATH (name, '/')
FROM organization START WITH parent_id IS NULL
CONNECT BY PRIOR id = parent_id; /* /ВГУ ; /ВГУ/ПММ ; /ВГУ/ПММ/МО
ЭВМ */
```

Если запрос с CONNECT BY применяется для данных с рекурсивной организацией и находит запись, которая является своим собственным потомком (непосредственным или через цепочку потомков), то по умолчанию Oracle выдает ошибку “CONNECT BY loop in user data”

Для того, чтобы в такой ситуации выдать иерархию без зацикливаний, используется опция **NOCYCLE** – это ключевое слово ставится сразу после CONNECT BY

Псевдостолбец **CONNECT_BY_ISCYCLE** содержит значение 1 для всех строк, которые являются своими собственными потомками, и 0 для остальных строк

Правда, создается впечатление, что в версии Oracle 11.2 эти возможности не всегда ведут себя ожидаемым образом (обнаружено экспериментально, аналогичные моменты упомянуты на форуме SQL.RU)

Синтаксис иерархических запросов СУБД Oracle может применяться и для других целей, например, генерации числовой последовательности

```
SELECT ROWNUM AS rn  
FROM DUAL  
CONNECT BY LEVEL <= ?;
```

Запрос генерирует выборку, содержащую один столбец, в строках значения от 1 до указанного параметра

```
WITH query (name, id, level_) AS (  
    SELECT name, id, 1 FROM organization WHERE parent_id IS NULL  
UNION ALL  
    SELECT o.name, o.id, q.level_+1 FROM organization o, query q  
    WHERE q.id = o.parent_id )  
SELECT * FROM query;
```

Выделенное обращение к формируемому запросу – рекурсивное
Первый подзапрос внутри скобки описывает первый уровень иерархии

Второй подзапрос внутри скобки описывает переход от предыдущего уровня иерархии к следующему

Связка подзапросов – обязательно UNION ALL

Oracle не допускает ключевого слова RECURSIVE после WITH и требует задания списка имен столбцов рекурсивного подзапроса

- Возможности, аналогичные LEVEL, CONNECT_BY_ROOT и SYS_CONNECT_BY_PATH легко реализуются путем включения соответствующих столбцов в результат (Q - формируемый подзапрос, T – исходная таблица)
- Q.LEVEL = 1 для первого уровня, Q.LEVEL+1 для последующего
- Q.SYS_CONNECT_BY_PATH = разделитель || T.значение для первого уровня, Q.SYS_CONNECT_BY_PATH || разделитель || T.значение для последующего
- Q.CONNECT_BY_ROOT = T.значение для первого уровня, Q.значение для последующего

- Возможности NOCYCLE и ORDER BY SIBLINGS реализуются Oracle с помощью аналогичных конструкций:
- CYCLE с указанием столбца, содержащего признак зацикливания
- SEARCH с указанием столбца, содержащего нумерацию записей

- Если нужно получить только набор записей, которым подчинена указанная запись (напрямую или транзитивно), иерархию имеет смысл перевернуть

SELECT

name,

LEVEL

FROM

organization

START WITH

name = ?

CONNECT BY

PRIOR parent_id = id;

Здесь каждая подчиненная запись имеет уровень на единицу выше, чем та, которой она подчинена

- В СУБД ЛИНТЕР поддерживаются следующие элементы синтаксиса СУБД Oracle для иерархических запросов:
 - START WITH
 - CONNECT BY
 - PRIOR (в WHERE)
 - LEVEL (в SELECT; в WHERE - с ограничениями)
 - ORDER BY SIBLINGS
- Планируется в 2016 году реализовать поддержку остальных элементов синтаксиса СУБД Oracle для иерархических запросов

1. Иерархические (рекурсивные запросы). Пользователь maovrn
<https://habrahabr.ru/post/43955/> Ноябрь 2008 г.
2. Рекурсивные запросы в Oracle. Пржиялковский В.В.
<http://citforum.ru/database/oracle/recursive/> Июль 2010 г.

Спасибо за внимание!

Вопросы?