ЛАБОРАТОРНАЯ РАБОТА №18

Тема: Преобразование строк результата запроса в столбцы и столбцов результата в строки (транспонирование)

Цель: Пробрести навыки создания и использования аналитических SQL запросов с использованием инструментов PIVOT и UNPIVOT.

Теоретические сведения

В данной лабораторной работе рассматриваются примеры преобразования строк результата SQL запроса в столбцы (PIVOT) и обратное преобразование - UNPIVOT. Такие преобразования востребованы при составлении аналитических сводных отчетов. При выводе большого количества фактических результатов – например, списка продаж преобразование строк в столбцы не используется.

select * from match_results

«Ручное» преобразование строк в столбцы

Пусть требуется составить отчет о количестве сотрудников в каждом департаменте (используется БД HR). Такой запрос можно составить с использование группирования:

```
select department_name
     , count(*)
     from HR.employees a
         join HR.departments b
           on a.department_id=b.department_id
     group by department_name
     order by count(*) desc
```

DEPARTMENT_NAME	COUNT(*)
Shipping	45
Sales	34
Finance	6
Purchasing	6
IT	5
Executive	3
Accounting	2
Marketing	2
Public Relations	1

Human Resources 1
Administration 1

Но нам требуется сделать название департаментов столбцами вместо строк. Для решения этой задачи нужно «повернуть» (PIVOT) результат. Это можно сделать явным определением колонки для каждого департамента. Каждая колонка должна включать строки, относящиеся к этому департаменту. Количество сотрудников не может быть пустым – null. Разработчик может указать нужный департамент. Для вывода в каждой колонке требуемого департамента используется конструкция case в качестве параметра функции count. Case возвращает 1 для строк с заданным названием департамента. Таким образом, ведется подсчет количества сотрудников в каждом департаменте. Конструкция имеет следующий вид:

```
count ( case when department_name = '<нужный
департамент>' then 1 end ) <Нужный департамент>
```

Полностью SQL запрос можно записать следующим образом:

```
select count ( case when department_name = 'Shipping'
then 1 end ) shipping
     , count ( case when department_name = 'Sales' then
1 end ) Sales
     , count ( case when department_name = 'Finance'
then 1 end ) Finance
     , count ( case when department_name = 'Purchasing'
then 1 end ) Purchasing
     , count ( case when department_name = 'IT' then 1
end ) IT
     , count ( case when department_name = 'Executive'
then 1 end ) Executive
     , count ( case when department_name = 'Accounting'
then 1 end ) Accounting
     , count ( case when department_name = 'Marketing'
then 1 end ) Marketing
     , count ( case when department_name = 'Public
Relations' then 1 end ) Public_Relations
     , count ( case when department_name = 'Human
Resources' then 1 end ) Human_Resources
     , count ( case when department_name =
'Administration' then 1 end ) Administration
     from employees a
```

```
          join departments b
            on a.department_id=b.department_id
```

SHIPPING	SALES	FINANCE	PURCHASING	EXECUTIVE	ACCOUNTING	MARKETING	PUBLIC_RELATIONS	HUMAN_RESOURCES	ADMINISTRATION
45	34	6	6	53	2	2	1	1	1

Предложение Pivot

Ручной разворот таблицы (транспонирование) является неуклюжим решением. В этом случае легко совершить ошибку и, кроме того, код предложения плохо читается.

Предложение **pivot**, введенное в Oracle Database 11g, позволяется решить задачу более изящно.

Для использования этого предложения нужно указать три параметра:

1. Значение, которые будут подсчитываться и помещаться в колонки. В нашем случае требуется подсчитать количество, поэтому используется функция count(*).
2. Колонку, содержащую значения, которые станут новыми колонками. В нашем примере это названия департаментов.
3. Список значений, которые выносится в названия колонок. Это названия департаментов. Другими словами, колонки можно формировать не для всех департаментов, а только для указанных в списке.

В итоге получим следующую конструкцию:

```
with rws as (
   select department_name from departments a
        join employees b
          on a.department_id=b. department_id)
select * from rws pivot (
     count(*) for department_name in
      ( 'Shipping', 'Sales', 'Finance', 'IT' ) );
```

'Shipping'	'Sales'	'Finance'	'IT'
45	34	6	5

В такой конструкции, если нас интересует больше департаментов, их названия можно добавить в список IN!

Составим запрос, который выведет в каждой колонке, соответствующей департаменту, дату приема на работу первого сотрудника:

```
with rws as (
   select department_name,hire_date from departments a
        join employees b
          on a.department_id=b.department_id

)
select * from rws
      pivot (min(hire_date) for department_name in
         ( 'Shipping', 'Sales', 'Finance', 'IT' ) );
```

Получим результат:

'Shipping'	'Sales'	'Finance'	'IT'
01-MAY-03	30-JAN-04	16-AUG-02	25-JUN-05

Неявная группировка

Любая колонка из исходного запроса, не указанная pivot предложении формирует неявную группировку. Это может привести к появлению большего количества строк чем ожидается.

Например, если поместить pivot после ключевого слова from, получим неожиданный результат:

```
select * from employees a
join departments b on a.department_id=b.department_id
 pivot ( count(*) for department_name in ( 'Finance',
'Sales', 'IT', 'Shiping' ) );
```

Для преодоления этого препятствия следует использовать внутреннее представление (CTE). При этом выбирают только те колонки, которые используются в **pivot**.

Использование выражений

Часто требуется манипулировать значения, которые требуется использовать в предложении pivot. Пусть требуется вывести количество,

принятых на работу в каждом месяце. В этом случае требуется названия месяцев вывести в колонках.

Итак, требуется преобразовать дату в название месяца. Для извлечения названия месяца из даты можно использовать функцию to_char() с указанием формата 'MON'.

Если эту функцию использовать в предложении pivotполучим ошибку:

```
with rws as ( select hire_date from employees )
select * from rws
pivot ( count (*) for to_char ( hire_date, 'MON' ) in (
'JAN', 'FEB', 'MAR' ) );
```

Для устранения ошибки, нужно извлекать месяц во внутреннем представлении или CTE. Следует присвоить выражению псевдоним и использовать этот псевдоним в предложении **pivot**. Например:

```
with rws as (
select to_char ( hire_date, 'MON' ) mm
   from employees
)
select * from rws
pivot ( count (*) for mm in ( 'JAN', 'FEB', 'MAR' ) );
'JAN' 'FEB' 'MAR'
14    13    17
```

Для изменения значений в колонках, используемых в операции pivot, нужно изменять их во внутреннем представлении. This makes it easy to select and manipulate the columns you want.

Фильтрация строк в предложении Pivot

Для фильтрации строк в предложении pivot можно использовать предложение where. Это предложение вставляют после предложения pivot.

Пусть, например, требуется вывести количество принятых сотрудников с разбивкой по месяцам (как рассматривалось в предыдущем примере). Если необходимо ограничиться только теми департаментами, в которых был один или более прием на работу в январе, нужно использовать

следующий вариант SQL запроса. Отметим, использование условия **"'JAN'" > 0** в конце предложения.

```
with rws as (
select department_name
     , to_char ( hire_date, 'MON' ) hire_month
     from employees a join departments b
       on b.department_id = a.department_id
)
  select * from rws
     pivot ( count (*) for hire_month in ( 'JAN',
'FEB', 'MAR' ) ) where "'JAN'" > 0
```

DEPARTMENT_NAME	'JAN'	'FEB'	'MAR'
Sales	7	2	12
Shipping	5	8	4
IT	1	2	0
Executive	1	0	0

Новые имена колонок

По умолчанию имена колонок устанавливаются такими как значения в IN списке. Имена колонок при этом заключаются в кавычки. Это делает SQL предложение неудобным потому, что теперь для ссылок на колонки нужно использовать кавычки (см. предложение where).

С целью упрощения можно присвоить псевдоним каждому значению в списке:

```
with rws as (
select department_name
     , to_char ( hire_date, 'MON' ) hire_month
     from employees a join departments b
       on b.department_id = a.department_id
)
  select * from rws
     pivot ( count (*) for hire_month in ( 'JAN' jan,
'FEB' feb, 'MAR' mar) ) where jan > 0
```

DEPARTMENT_NAME	JAN	FEB	MAR
Sales	7	2	12
Shipping	5	8	4
IT	1	2	0
Executive	1	0	0

При этом обеспечивается упрощение формирование условия фильтрации.

Поворот (Pivot) многих значений

Имеется возможность преобразовать (повернуть) множество значений. Пусть, например, требуется для каждого товара с разбивкой по месяцам вывести:

- Количество продаж
- Сумму продаж

Это можно решить используя следующие выражения:

- sum(qnt)
- sum (sum)

Обратите внимание на широкое использование псевдонимов. Но основе этих псевдонимов автоматически конструируются имена колонок:

```
with rws as (
   select product_name
        , to_char(order_date,'MON') order_date
        , quantity qnt
        , quantity*unit_price summa
          from product_information a
        join order_items b
          on a.product_id=b.product_id
        join orders c
          on c.order_id=b.order_id
)
select * from rws
      pivot (sum(qnt) qtt
           , sum(summa) sum
      for order_date in
         ( 'JAN' jan, 'FEB' feb, 'MAR' mar, 'APR' apr )
);
```

PRODUCT_NAME	JAN_QTT	JAN_SUM	FEB_QTT	FEB_SUM	MAR_QTT	MAR_SUM	APR_QTT	APR_SUM
LCD Monitor 9/PM	205	46475.4	-	-	-	-	-	-
PS 110V HS/US	17	1632	-	-	-	-	-	-
Business Cards - 1000/2L	1	270.6	-	-	-	-	-	-

PS 220V /HS/FR	17	1547	-	-	-	-	-	-
Mouse C/E	-	-	175	6920	209	8305	-	-
Plastic Stock - W/HD	-	-	-	-	-	-	3	6.6
HD 9.1GB @10000 /I	-	-	-	-	-	-	-	-
Battery Backup (DA-130)	-	-	-	-	-	-	-	-
Compact 400/LQ	-	-	-	-	-	-	-	-
Cable Harness	-	-	19	62.7	-	-	-	-

Каждый новый агрегат буде помещен в новую колонку для каждого значения из IN списка. Общее количество новых колонок определяется формулой:

Количество агрегатов * количество значений в IN списке

Динамическое преобразование (Dynamic Pivoting)

Часто требуется изменить списки колонок для поворота, которые перечисляются в предложении pivot. Например, список отделов может изменяться. Или требуется изменять список месяцев для вывода. Это довольно неудобно.

К сожалению, обычный оператор PIVOT не имеет нужных возможностей. Значения в IN списке фиксированы. Нет возможности для изменения списка использовать например подзапросы.

К счастью, имеется возможность динамически формировать списки использовать средства XML. В этом случае можно формировать значение с использованием подзапросов. Также можно генерировать итоги для каждого значения в pivot колонке используя ключевое слово ANY:

```
with rws as (
  select department_name
  from employees a join departments b on
a.department_id=b.department_id
)
  select xmlserialize ( -- formats an XML document
  document department_name_xml as clob indent size=2
         ) department_name_xml
  from   rws
  pivot xml (
    count (*) matches for department_name in ( any )
```

```
);
```

Этот запрос выводит таблицу в XML формате. Для анализа документа нужно использовать средства работы с XML документами.

Обратное преобразование - Unpivoting

Обратное преобразование (Unpivoting) это процесс преобразования колонок в строки. Для демонстрации особенностей предложения Unpivot рассмотрим таблицу, показанную приложении к данной лабораторной работе. Выполните скрипт указанный в приложении. Пусть требуется преобразовать название команды гостей и команды хозяев в название одной команды. Нужно вывести название команды в одной колонке, неважно где играла команда: дома или в гостях..

Задачу можно решить традиционными средствами с использованием операции union all.

Например, для получения строки для команды хозяев и команды гостей для каждого матча, можно составить два запроса, объединенных операцией union all:

```
select match_date
          , location
          , 'HOME' home_or_away
         , home_team_name team
          from match_results
        union all
 select match_date
           , location
          , 'AWAY' home_or_away
            , away_team_name team
              from match_results
    order by match_date
           , location
           , home_or_away;
```

Предложение Unpivot

Вместо приведенного выше решения разработчик может использовать предложение **unpivot**. Это предложение также введено Oracle Database 11g и делает обратное преобразование проще.

1. Новая колонка содержит разворачиваемые (pivoted) значение

2. Другая новая колонка содержит исходные данные этот значения
3. Колонки преобразуются в строки

Предыдущий запрос можно представить в следующем виде:

```
select match_date
          , location
          , home_or_away
          , team
      from   match_results
             unpivot (
                      team for home_or_away in (
                         home_team_name as 'HOME'
                       , away_team_name as 'AWAY'
                      )
               )
order
by match_date
   , location
   , home_or_away;
```

Объединение предложений Pivot и Unpivot

Разработчик имеет возможность комбинировать предложения **pivot** и **unpivot** в одном предложении. Результат выполнения первого предложения становится входными данными для второго предложения.

Пусть, например, требуется создать таблицу результатов лиги на основе таблицы результатов матчей. Для каждой команды нужно вывести количество выигранных матчей, количество поражений и игр количество ничьих.

Для составления запросов нужно выполнить несколько шагов:

Во-первых, сравнить очки команды гостей и команды хозяев. Это даст возможность определить результат команды: выигрыш, ничья или поражение.

Во-вторых, подсчитать количество результатов для каждой команды. Здесь потребуется одна колонка с различными названиями результатов. Для этого нужно выполнить операцию **unpivot** для гостей и хозяев, также потребуется одна колонка для названия команд.

Наконец, для формирования таблицы результатов лиги, нужно преобразовать (pivot) количество побед, поражения, и ничьих для каждой команды.

Указанные шаги позволяют составить следующий запрос:

```
with rws as (
  select home_team_name, away_team_name,
         case
           when home_team_points > away_team_points
then 'WON'
           when home_team_points < away_team_points
then 'LOST'
           else 'DRAW'
         end home_team_result,
         case
           when home_team_points < away_team_points
then 'WON'
           when home_team_points > away_team_points
then 'LOST'
           else 'DRAW'
         end away_team_result
  from   match_results
)
  select team, w, d, l
  from   rws
  unpivot (
    ( team, result ) for home_or_away in (
      ( home_team_name, home_team_result ) as 'HOME',
      ( away_team_name, away_team_result ) as 'AWAY'
    )
  )
  pivot (
    count (*), min ( home_or_away ) dummy
    for result in (
      'WON' W, 'DRAW' D, 'LOST' L
    )
  )
  order  by w desc, d desc, l;
```

Порядок выполнения работы

1. На основе запросов по БД HR, указанных выше, и в которых группируютя данные по названиям отделов, составить

аналогичные запросы для группировки по названиям должностей.

2. Изменить следующий запрос для преобразования (**unpivot**) очки команд хозяев и гостей для каждого матча:

```
select match_date
     , location
     , home_or_away
     , points
       from   match_results
         unpivot (
               home_or_away in (
               home_team_points as 'HOME'
             , away_team_points as 'AWAY'
  )
)
order  by match_date, location, home_or_away;
```

Составленный запрос должен выводить следующий результат:

MATCH_DATE	LOCATION	HOME_OR_AWAY	POINTS
01-JAN-2018	Coldgate	AWAY	4
01-JAN-2018	Coldgate	HOME	1
01-JAN-2018	Snowley	AWAY	0
01-JAN-2018	Snowley	HOME	2
01-FEB-2018	Dorwall	AWAY	1
01-FEB-2018	Dorwall	HOME	0
01-MAR-2018	Coldgate	AWAY	3
01-MAR-2018	Coldgate	HOME	3
02-MAR-2018	Newdell	AWAY	0
02-MAR-2018	Newdell	HOME	8

3. Получить результат запросов, для которых эти результаты не показаны выше

Содержание отчета.

1. Название работы
2. Цель работы
3. Листинги запросов
4. Скриншоты результатов выполнения запросов.
5. Выводы

Контрольные вопросы

1. Объяснить назначение предложения **PIVOT**.
2. Объяснить назначение предложения **UNPIVOT**.
3. Описать особенности преобразования строк в колонки таблицы
4. Как указываются названия колонок в операции **PIVOT**?
5. С какой целью используются средства **XML** в сочетании с предложением **PIVOT**?

Приложение

Скрипт создания таблицы результатов матчей футбольной лиги.

```
create table match_results ( match_date date, location
varchar2(20), home_team_name varchar2(20),
away_team_name varchar2(20), home_team_points integer,
away_team_points integer );
 insert into match_results values ( date'2018-01-01',
'Snowley', 'Underrated United', 'Terrible Town', 2, 0
);
 insert into match_results values ( date'2018-01-01',
'Coldgate', 'Average Athletic', 'Champions City', 1, 4
);
 insert into match_results values ( date'2018-02-01',
'Dorwall', 'Terrible Town', 'Average Athletic', 0, 1 );
 insert into match_results values ( date'2018-03-01',
'Coldgate', 'Average Athletic', 'Underrated United', 3,
3 );
insert into match_results values ( date'2018-03-02',
'Newdell', 'Champions City', 'Terrible Town', 8, 0 );
commit;
```