9

Работа с датами

В данной главе представлены рецепты для поиска и обработки дат. Запросы с участием дат очень распространены. При работе с датами необходимо соответствующим образом мыслить и хорошо понимать функции, предоставляемые СУБД для обращения с такими данными. Рецепты этой главы формируют необходимый фундамент для перехода к более сложным запросам, в которых задействованы не только даты, но и время.

Прежде чем перейти к рецептам, я хочу еще раз повторить основную мысль (о которой говорилось в предисловии): эти рецепты должны использоваться как руководства к решению конкретных задач. Старайтесь думать «глобально». Например, если рецепт решает задачу для текущего месяца, помните, что его можно использовать (с небольшими изменениями) для любого месяца. Повторюсь, я хочу чтобы представленные рецепты использовались как руководства, а не как абсолютный окончательный вариант. Книга не может охватить ответы на все вопросы, но если понять те решения, которые в ней представлены, то останется просто изменить их соответственно своим задачам. Советую также рассматривать все предлагаемые альтернативные версии решений. Например, если задача решается с использованием определенной предоставляемой СУБД функции, стоит потратить время и силы, чтобы узнать о возможном альтернативном решении, которое может быть более или менее эффективным по сравнению с приведенным здесь. Знание всех возможных вариантов сделает вас лучшим разработчиком на SQL.

В представленных в данной главе рецептах используются простые типы дат. В случае использования более сложных типов дат решения должны быть скорректированы соответствующим образом.

Как определить, является ли год високосным

Задача

Требуется определить, является ли текущий год високосным.

Решение

Те, кто уже имеет некоторый опыт работы с SQL, несомненно, встречали несколько методов решения этой задачи. Хороши практически все известные мне решения, но представленное в данном рецепте, наверное, самое простое: проверяется последний день февраля; если он является 29-м днем, то текущий год – високосный.

DB2

Для возвращения всех дней февраля используйте рекурсивный оператор WITH. С помощью агрегатной функции MAX определите последний день февраля.

```
 1    with x (dy,mth)
 2      as (
 3  select dy, month(dy)
 4    from (
 5  select (current_date -
 6          dayofyear(current_date) days +1 days)
 7           +1 months as dy
 8    from t1
 9         ) tmp1
10   union all
11  select dy+1 days, mth
12    from x
13   where month(dy+1 day) = mth
14  )
15  select max(day(dy))
16    from x
```

Oracle

Чтобы найти последний день февраля, используйте функцию LAST_DAY (последний день):

```
1 select to_char(
2          last_day(add_months(trunc(sysdate,'y'),1)),
3         'DD')
4   from t1
```

PostgreSQL

Чтобы возвратить все дни февраля, используйте функцию GENERATE_SERIES. Затем с помощью агрегатной функции MAX найдите последний день февраля:

```
select max(to_char(tmp2.dy+x.id,'DD')) as dy
  from (
select dy, to_char(dy,'MM') as mth
  from (
select cast(cast(
            date_trunc('year',current_date) as date)
                       + interval '1 month' as date) as dy
  from t1
       ) tmp1
       ) tmp2, generate_series (0,29) x(id)
 where to_char(tmp2.dy+x.id,'MM') = tmp2.mth
```

MySQL

Чтобы найти последний день февраля, используйте функцию LAST_DAY:

```
select day(
       last_day(
       date_add(
       date_add(
       date_add(current_date,
                interval -dayofyear(current_date) day),
                interval 1 day),
                interval 1 month))) dy
  from t1
```

SQL Server

Для возвращения всех дней февраля используйте рекурсивный оператор WITH. С помощью агрегатной функции MAX определите последний день февраля:

```
  with x (dy,mth)
    as (
select dy, month(dy)
  from (
select dateadd(mm,1,(getdate()-datepart(dy,getdate()))+1) dy
  from t1
       ) tmp1
 union all
select dateadd(dd,1,dy), mth
  from x
 where month(dateadd(dd,1,dy)) = mth
)
select max(day(dy))
  from x
```

Обсуждение[1]

DB2

Вложенный запрос TMP1 в рекурсивном представлении X возвращает первый день февраля следующим образом:

1. Определяется текущая дата.
2. С помощью функции DAYOFYEAR определяется, сколько дней нынешнего года прошло до текущей даты.
3. Вычитанием найденного количества дней из текущей даты получается 31 декабря предыдущего года, и добавляется один день, чтобы достичь 1 января текущего года.
4. Добавляется один месяц, чтобы получить 1 февраля.

Результат всех этих вычислений представлен ниже:

```
select (current_date -
          dayofyear(current_date) days +1 days) +1 months as dy
  from t1

DY
-----------
01-FEB-2005
```

Следующий шаг – с помощью функции MONTH из даты, возвращенной вложенным запросом TMP1, получить месяц:

```
select dy, month(dy) as mth
  from (
select (current_date -
          dayofyear(current_date) days +1 days) +1 months as dy
  from t1
       ) tmp1

DY           MTH
-----------  ---
01-FEB-2005    2
```

Представленные до сих пор результаты обеспечивают отправную точку для рекурсивной операции, в ходе которой генерируется каждый день февраля. Чтобы получить все дни февраля, многократно добавляем по одному дню к DY, до тех пор пока не будет возвращено 1 марта. Результаты выполнения операции WITH приведены ниже:

```
  with x (dy,mth)
    as (
select dy, month(dy)
  from (
select (current_date -
```

[1] Последний день февраля также можно получить путем вычитания одного дня из первого дня марта. – *Примеч. науч. ред.*

```
        dayofyear(current_date) days +1 days) +1 months as dy
  from t1
       ) tmp1
union all
select dy+1 days, mth
  from x
 where month(dy+1 day) = mth
)
select dy,mth
  from x
```

```
DY           MTH
-----------  ---
01-FEB-2005    2
...
10-FEB-2005    2
...
28-FEB-2005    2
```

Заключительный шаг – применить к столбцу DY функцию MAX, чтобы выбрать последний день февраля. Если это окажется 29-е число, то год – високосный.

Oracle

Первый шаг – найти начало года, используя функцию TRUNC:

```
select trunc(sysdate,'y')
  from t1
```

```
DY
-----------
01-JAN-2005
```

Поскольку первый день года – 1 января, следующий шаг – добавляем один месяц, чтобы получить 1 февраля:

```
select add_months(trunc(sysdate,'y'),1) dy
  from t1
```

```
DY
-----------
01-FEB-2005
```

Далее с помощью функции LAST_DAY находим последний день февраля:

```
select last_day(add_months(trunc(sysdate,'y'),1)) dy
  from t1
```

```
DY
-----------
28-FEB-2005
```

Последний шаг (необязательный) – используем функцию TO_CHAR, чтобы возвратить 28 или 29.

PostgreSQL

Первый шаг – проверить результаты, возвращенные вложенным запросом TMP1. С помощью функции DATE_TRUNC находим первый день текущего года и приводим результат к типу DATE:

```
select cast(date_trunc('year',current_date) as date) as dy
  from t1

DY
-----------
01-JAN-2005
```

Следующий шаг – добавляем месяц к первому дню текущего года, чтобы получить первый день февраля, приводя результат к типу DATE:

```
select cast(cast(
            date_trunc('year',current_date) as date)
                       + interval '1 month' as date) as dy
  from t1

DY
-----------
01-FEB-2005
```

Далее возвращаем из вложенного запроса TMP1 значение DY, выбирая из него с помощью функции TO_CHAR порядковый номер месяца:

```
select dy, to_char(dy,'MM') as mth
   from (
 select cast(cast(
             date_trunc('year',current_date) as date)
                        + interval '1 month' as date) as dy
   from t1
        ) tmp1

DY           MTH
-----------  ---
01-FEB-2005    2
```

Представленные до сих пор результаты составляют результирующее множество вложенного запроса TMP2. Следующий шаг – с помощью исключительно полезной функции GENERATE_SERIES получить 29 строк (со значениями от 1 до 29). Каждая возвращенная GENERATE_SERIES строка (множество этих строк образует множество под псевдонимом X) добавляется к DY вложенного запроса TMP2. Результаты частично показаны ниже:

```
select tmp2.dy+x.id as dy, tmp2.mth
  from (
select dy, to_char(dy,'MM') as mth
  from (
select cast(cast(
            date_trunc('year',current_date) as date)
                       + interval '1 month' as date) as dy
```

```
  from t1
       ) tmp1
       ) tmp2, generate_series (0,29) x(id)
 where to_char(tmp2.dy+x.id,'MM') = tmp2.mth
```

```
DY           MTH
-----------  ---
01-FEB-2005   02
...
10-FEB-2005   02
...
28-FEB-2005   02
```

Заключительный шаг – с помощью функции MAX выбрать последний день февраля. К полученному значению применяется функция TO_CHAR, которая возвратит 28 или 29.

MySQL

Первый шаг – найти первый день текущего года, вычитая из текущей даты количество прошедших до нее дней года и затем прибавляя один день. Все это делает функция DATE_ADD:

```
select date_add(
       date_add(current_date,
                interval -dayofyear(current_date) day),
                interval 1 day) dy
  from t1
```

```
DY
-----------
01-JAN-2005
```

После этого добавляем один месяц, опять же используя функцию DATE_ADD:

```
select date_add(
       date_add(
       date_add(current_date,
                interval -dayofyear(current_date) day),
                interval 1 day),
                interval 1 month) dy
  from t1
```

```
DY
-----------
01-FEB-2005
```

Добравшись до февраля, используем функцию LAST_DAY, чтобы найти последний день месяца:

```
select last_day(
       date_add(
       date_add(
```

```
       date_add(current_date,
                interval -dayofyear(current_date) day),
                interval 1 day),
                interval 1 month)) dy
  from t1

DY
-----------
28-FEB-2005
```

Заключительный шаг (необязательный) – использовать функцию DAY, чтобы возвратить 28 или 29.

SQL Server

В данном решении для получения всех дней февраля используется рекурсивный оператор WITH. Первый шаг – найти первый день февраля. Для этого ищем первый день текущего года, вычитая из текущей даты количество прошедших до нее дней года и затем прибавляя один день. Получив первый день текущего года, с помощью функции DATEADD добавляем к нему один месяц, чтобы перейти к первому дню февраля:

```
select dateadd(mm,1,(getdate()-datepart(dy,getdate()))+1) dy
  from t1

DY
-----------
01-FEB-2005
```

Далее возвращаем первый день февраля и порядковый номер февраля:

```
select dy, month(dy) mth
  from (
select dateadd(mm,1,(getdate()-datepart(dy,getdate()))+1) dy
  from t1
       ) tmp1

DY          MTH
-----------  ---
01-FEB-2005    2
```

Используем рекурсивные возможности оператора WITH и добавляем по одному дню к DY, возвращаемому вложенным запросом TMP1, до тех пор пока не дойдем до марта (результаты частично показаны ниже):

```
  with x (dy,mth)
    as (
select dy, month(dy)
  from (
select dateadd(mm,1,(getdate()-datepart(dy,getdate()))+1) dy
  from t1
       ) tmp1
 union all
```

```
select dateadd(dd,1,dy), mth
  from x
 where month(dateadd(dd,1,dy)) = mth
)
select dy,mth from x

DY           MTH
-----------  ---
01-FEB-2005   02
...
10-FEB-2005   02
...
28-FEB-2005   02
```

Теперь, возвратив все дни февраля, с помощью функции MAX определяем последний день. Как необязательный заключительный шаг можно использовать функцию DAY, чтобы получить только число 28 или 29, а не всю дату.

Как определить количество дней в году

Задача

Требуется подсчитать количество дней в текущем году.

Решение

Количество дней в текущем году – это разница между первым днем следующего года и первым днем текущего года (в днях). Каждое решение включает в себя следующие этапы:

1. Определение первого дня текущего года.
2. Добавление одного года к этой дате (для получения первого дня следующего года).
3. Вычитание текущего года из результата шага 2.

Решения отличаются только используемыми встроенными функциями.

DB2

С помощью функции DAYOFYEAR найдите первый день текущего года и определите количество дней в текущем году, используя функцию DAYS:

```
1 select days((curr_year + 1 year)) - days(curr_year)
2   from (
3 select (current_date -
4         dayofyear(current_date) day +
5          1 day) curr_year
6   from t1
7        ) x
```

Oracle

С помощью функции TRUNC найдите начало текущего года и, используя функцию ADD_MONTHS, определите начало следующего года:

```
1 select add_months(trunc(sysdate,'y'),12) - trunc(sysdate,'y')
2   from dual
```

PostgreSQL

С помощью функции DATE_TRUNC найдите начало текущего года. Затем, используя арифметику интервалов, найдите начало следующего года:

```
1 select cast((curr_year + interval '1 year') as date) - curr_year
2   from (
3 select cast(date_trunc('year',current_date) as date) as curr_year
4   from t1
5        ) x
```

MySQL

Используйте ADDDATE, чтобы найти начало текущего года. С помощью функции DATEDIFF и арифметики интервалов определите количество дней в году:

```
1 select datediff((curr_year + interval 1 year),curr_year)
2   from (
3 select adddate(current_date,-dayofyear(current_date)+1) curr_year
4   from t1
5        ) x
```

SQL Server

С помощью функции DATEADD найдите первый день текущего года. Чтобы возвратить число дней в текущем году, используйте DATEDIFF:

```
1 select datediff(d,curr_year,dateadd(yy,1,curr_year))
2   from (
3 select dateadd(d,-datepart(dy,getdate())+1,getdate()) curr_year
4   from t1
5        ) x
```

Обсуждение

DB2

Первый шаг – найти первый день текущего года. Для определения количества прошедших дней текущего года используется функция DAYOFYEAR (день года). Вычитая это значение из текущей даты, получаем последний день прошлого года и затем добавляем 1:

```
select (current_date -
        dayofyear(current_date) day +
         1 day) curr_year
```

```
  from t1

CURR_YEAR
-----------
01-JAN-2005
```

Теперь, имея первый день текущего года, просто прибавляем к нему один год, тем самым получаем первый день следующего года. Затем вычитаем начало текущего года из начала следующего года.

Oracle

Первый шаг – найти первый день текущего года, что можно без труда сделать, вызвав встроенную функцию TRUNC и передав в качестве второго аргумента «Y» (таким образом получая, исходя из текущей даты, первый день текущего года):

```
select select trunc(sysdate,'y') curr_year
  from dual

CURR_YEAR
-----------
01-JAN-2005
```

Затем добавляем один год, чтобы получить первый день следующего года. Наконец, находим разность между двумя датами и определяем количество дней в текущем году.

PostgreSQL

Начинаем с получения первого дня текущего года. Для этого вызываем функцию DATE_TRUNC следующим образом:

```
select cast(date_trunc('year',current_date) as date) as curr_year
  from t1

CURR_YEAR
-----------
01-JAN-2005
```

Теперь можно без труда добавить год, чтобы получить первый день следующего года, и тогда останется только вычесть одну дату из другой. Вычитать следует более раннюю дату из более поздней. В результате будет получено количество дней в текущем году.

MySQL

Первый шаг – найти первый день текущего года. С помощью функции DAYOFYEAR находим количество прошедших дней текущего года. Вычитаем это значение из текущей даты и добавляем 1:

```
select adddate(current_date,-dayofyear(current_date)+1) curr_year
  from t1

CURR_YEAR
```

```
-----------
01-JAN-2005
```

Получив первый день текущего года, добавляем к нему один год, в результате чего получаем первый день следующего года. Затем вычитаем начало текущего года из начала следующего года. В итоге находим число дней в текущем году.

SQL Server

Первый шаг – найти первый день текущего года. Вычитаем из текущей даты количество прошедших дней года и добавляем 1, используя функции DATEADD и DATEPART:

```
select dateadd(d,-datepart(dy,getdate())+1,getdate()) curr_year
  from t1

CURR_YEAR
-----------
01-JAN-2005
```

Получив первый день текущего года, добавляем к нему один год, в результате чего получаем первый день следующего года. Затем вычитаем начало текущего года из начала следующего. В итоге находим количество дней в текущем году.

Извлечение единиц времени из даты

Задача

Требуется разложить текущую дату на шесть частей: день, месяц, год, секунды, минуты и часы. Результаты должны быть возвращены в численном виде.

Решение

В данном рецепте текущая дата выбрана для иллюстрации, он свободно может использоваться с другими датами. В главе 1 говорилось о важности изучения и использования преимуществ встроенных функций, предоставляемых СУБД. Это особенно актуально, когда дело касается дат. Существует множество других способов извлечения единиц времени из даты, кроме представленных в данном рецепте. Очень полезно поэкспериментировать с разными техниками и функциями.

DB2

DB2 реализует ряд встроенных функций, которые упрощают задачу по извлечению частей даты. Имена функций HOUR (час), MINUTE (минута), SECOND (секунда), DAY, MONTH и YEAR соответствуют возвращаемым ими единицам измерения времени. Если требуется получить день, используется функция DAY, час – функция HOUR и т. д. Например:

```
1  select   hour( current_timestamp ) hr,
2         minute( current_timestamp ) min,
3         second( current_timestamp ) sec,
4            day( current_timestamp ) dy,
5          month( current_timestamp ) mth,
6           year( current_timestamp ) yr
7    from t1
```

```
  HR   MIN   SEC    DY   MTH    YR
---- ----- ----- ----- ----- -----
  20    28    36    15     6  2005
```

Oracle

Для выбора определенных частей даты, соответствующих тем или иным единицам времени, используйте функции TO_CHAR и TO_NUMBER:

```
1  select to_number(to_char(sysdate,'hh24')) hour,
2         to_number(to_char(sysdate,'mi')) min,
3         to_number(to_char(sysdate,'ss')) sec,
4         to_number(to_char(sysdate,'dd')) day,
5         to_number(to_char(sysdate,'mm')) mth,
6         to_number(to_char(sysdate,'yyyy')) year
7    from dual
```

```
HOUR   MIN   SEC   DAY   MTH  YEAR
---- ----- ----- ----- ----- -----
  20    28    36    15     6  2005
```

PostgreSQL

Для выбора определенных частей даты, соответствующих тем или иным единицам времени, используйте функции TO_CHAR и TO_NUMBER:

```
1 select to_number(to_char(current_timestamp,'hh24'),'99') as hr,
2        to_number(to_char(current_timestamp,'mi'),'99') as min,
3        to_number(to_char(current_timestamp,'ss'),'99') as sec,
4        to_number(to_char(current_timestamp,'dd'),'99') as day,
5        to_number(to_char(current_timestamp,'mm'),'99') as mth,
6        to_number(to_char(current_timestamp,'yyyy'),'9999') as yr
7   from t1
```

```
  HR   MIN   SEC   DAY   MTH    YR
---- ----- ----- ----- ----- -----
  20    28    36    15     6  2005
```

MySQL

Для выбора определенных частей даты, соответствующих тем или иным единицам времени, используйте функцию DATE_FORMAT:

```
1 select date_format(current_timestamp,'%k') hr,
2        date_format(current_timestamp,'%i') min,
3        date_format(current_timestamp,'%s') sec,
4        date_format(current_timestamp,'%d') dy,
```

```
5        date_format(current_timestamp,'%m') mon,
6        date_format(current_timestamp,'%Y') yr
7   from t1
```

```
  HR   MIN   SEC    DY   MTH    YR
---- ----- ----- ----- ----- -----
  20    28    36    15     6  2005
```

SQL Server

Для выбора определенных частей даты, соответствующих тем или иным единицам времени, используйте функцию DATEPART (часть даты):

```
1  select datepart( hour,  getdate()) hr,
2         datepart( minute,getdate()) min,
3         datepart( second,getdate()) sec,
4         datepart( day,   getdate()) dy,
5         datepart( month, getdate()) mon,
6         datepart( year,  getdate()) yr
7    from t1
```

```
  HR   MIN   SEC    DY   MTH    YR
---- ----- ----- ----- ----- -----
  20    28    36    15     6  2005
```

Обсуждение

В этих решениях нет ничего необычного, просто используем то, за что уже заплачено. Потрудитесь изучить функции для работы с датами, которые имеете в своем распоряжении. В решениях этого рецепта мы лишь поверхностно затронули представленные функции. Каждая из них может принимать намного больше аргументов и возвращать больше информации, чем показано здесь.

Определение первого и последнего дней месяца

Задача

Требуется получить первый и последний дни текущего месяца.

Решение

Представленные здесь решения обеспечивают поиск первого и последнего дней текущего месяца. Текущий месяц выбран произвольно. С небольшими корректировками эти решения можно применять к разным месяцам.

DB2

С помощью функции DAY возвращаем количество прошедших в текущем месяце дней на основании представленной текущей даты. Вычитаем это значение из текущей даты и добавляем 1, чтобы получить

первый день месяца. Чтобы определить последний день месяца, добавляем один месяц к текущей дате и вычитаем значение, возвращенное функцией DAY для текущей даты:

```
1 select (current_date - day(current_date) day +1 day) firstday,
2        (current_date +1 month -day(current_date) day) lastday
3   from t1
```

Oracle

Первый день месяца находим с помощью функции TRUNC, последний день – с помощью функции LAST_DAY:

```
1 select trunc(sysdate,'mm') firstday,
2        last_day(sysdate) lastday
3   from dual
```

Описанное здесь применение TRUNC приведет к потере всех компонентов времени даты, тогда как LAST_DAY сохранит время.

PostgreSQL

С помощью функции DATE_TRUNC получаем из текущей даты первый день месяца. Имея первый день месяца, добавляем к нему один месяц и вычитаем один день, чтобы найти последний день текущего месяца:

```
1 select firstday,
2        cast(firstday + interval '1 month'
3                      - interval '1 day' as date) as lastday
4   from (
5 select cast(date_trunc('month',current_date) as date) as firstday
6   from t1
7        ) x
```

MySQL

С помощью функций DATE_ADD и DAY возвращаем количество прошедших в текущем месяце дней согласно представленной текущей дате. Затем вычитаем это значение из текущей даты и добавляем 1, чтобы найти первый день месяца. Последний день месяца получаем, используя функцию LAST_DAY:

```
1 select date_add(current_date,
2                 interval -day(current_date)+1 day) firstday,
3        last_day(current_date) lastday
4   from t1
```

SQL Server

С помощью функций DATEADD и DAY возвращаем количество прошедших в текущем месяце дней согласно представленной текущей дате. Затем вычитаем это значение из текущей даты и добавляем 1, что-

бы найти первый день месяца. Чтобы получить последний день, добавляем один месяц к текущей дате и вычитаем из полученного результата значение, возвращенное функцией DAY для текущей даты, опять используя функции DAY и DATEADD:

```
1 select dateadd(day,-day(getdate())+1,getdate()) firstday,
2        dateadd(day,
3                -day(getdate()),
4                dateadd(month,1,getdate())) lastday
5   from t1
```

Обсуждение

DB2

Для получения первого дня месяца используем функцию DAY: она возвращает день соответственно переданной в нее дате. Если из текущей даты вычесть значение, возвращенное DAY(CURRENT_DATE), то получаем последний день предыдущего месяца; добавляем к нему один день и имеем первый день текущего месяца. Чтобы найти последний день месяца, добавляем один месяц к текущей дате. Тем самым мы попадем в то же число следующего месяца (независимо от того, сколько дней в текущем месяце)[1]. Затем вычитаем значение, возвращенное DAY(CURRENT_DATE), чтобы получить последний день текущего месяца.

Oracle

Чтобы найти первый день текущего месяца, используем функцию TRUNC, передавая в нее в качестве второго аргумента «mm», что обеспечит возвращение первого дня месяца соответственно текущей дате. Чтобы найти последний день, просто используем функцию LAST_DAY.

PostgreSQL

Чтобы найти первый день текущего месяца, используем функцию DATE_TRUNC, передавая в нее в качестве второго аргумента «month», что обеспечит возвращение первого дня месяца соответственно текущей дате. Чтобы найти последний день, добавляем один месяц к первому дню месяца и вычитаем один день.

MySQL

Чтобы найти первый день текущего месяца, используем функцию DAY. Она возвращает день месяца соответственно переданной дате.

[1] Это может вызвать проблему, если в следующем месяце меньше дней, чем текущая дата. Поэтому более корректно сначала вычесть количество прошедших в текущем месяце дней и получить последний день предыдущего месяца, а только потом добавить один месяц для получения последнего дня текущего месяца. – *Примеч. науч. ред.*

Если из текущей даты вычесть значение, возвращенное DAY(CURRENT_DATE), получаем последний день предыдущего месяца; добавляем один день и в итоге имеем первый день текущего месяца. Последний день находим просто с помощью функции LAST_DAY.

SQL Server

Чтобы найти первый день текущего месяца, используем функцию DAY. Она возвращает день месяца соответственно переданной в нее дате. Если из текущей даты вычесть значение, возвращенное DAY(GETDATE()), получаем последний день предыдущего месяца; добавляем один день и в итоге имеем первый день текущего месяца. Последний день находим с помощью функции DATEADD. Добавляем один месяц к текущей дате, затем вычитаем значение, возвращенное DAY(GETDATE()), и получаем последний день текущего месяца.[1]

Выбор всех дат года, выпадающих на определенный день недели

Задача

Требуется найти все даты года, соответствующие заданному дню недели. Например, стоит задача сформировать список пятниц текущего года.

Решение

Независимо от СУБД основой решения являются возвращение всех дней текущего года и выбор из них только тех, которые соответствуют интересующему дню недели. В приведенных примерах извлекаются все пятницы года.

DB2

Рекурсивным оператором WITH возвращаем все дни текущего года. Затем с помощью функции DAYNAME выбираем только пятницы:

```
1    with x (dy,yr)
2      as (
3  select dy, year(dy) yr
4    from (
5  select (current_date -
6           dayofyear(current_date) days +1 days) as dy
7    from t1
8         ) tmp1
9   union all
```

[1] Та же проблема, что и в решении для DB2. Поэтому сначала следует вычесть количество прошедших в текущем месяце дней и получить последний день предыдущего месяца, а только потом добавить один месяц. – *Примеч. науч. ред.*

```
10  select dy+1 days, yr
11    from x
12   where year(dy +1 day) = yr
13  )
14  select dy
15    from x
16   where dayname(dy) = 'Friday'
```

Oracle

Рекурсивным оператором CONNECT BY возвращаем все дни текущего года. Затем с помощью функции TO_CHAR выбираем только пятницы:

```
 1    with x
 2      as (
 3  select trunc(sysdate,'y')+level-1 dy
 4    from t1
 5    connect by level <=
 6       add_months(trunc(sysdate,'y'),12)-trunc(sysdate,'y')
 7  )
 8  select *
 9    from x
10   where to_char( dy, 'dy') = 'fri'
```

PostgreSQL

Используя функцию GENERATE_SERIES, возвращаем все дни текущего года. Затем с помощью функции TO_CHAR выбираем только пятницы:

```
 1 select cast(date_trunc('year',current_date) as date)
 2        + x.id as dy
 3   from generate_series (
 4         0,
 5         ( select cast(
 6                  cast(
 7            date_trunc('year',current_date) as date)
 8                       + interval '1 years' as date)
 9                       - cast(
10                    date_trunc('year',current_date) as date) )-1
11         ) x(id)
12  where to_char(
13           cast(
14     date_trunc('year',current_date)
15                as date)+x.id,'dy') = 'fri'
```

MySQL

Используем сводную таблицу T500, чтобы возвратить все дни текущего года. Затем с помощью функции DAYNAME выбираем только пятницы:

```
1  select dy
2    from (
```

```
 3  select adddate(x.dy,interval t500.id-1 day) dy
 4    from (
 5  select dy, year(dy) yr
 6    from (
 7  select adddate(
 8         adddate(current_date,
 9                 interval -dayofyear(current_date) day),
10                 interval 1 day ) dy
11    from t1
12         ) tmp1
13         ) x,
14         t500
15   where year(adddate(x.dy,interval t500.id-1 day)) = x.yr
16         ) tmp2
17   where dayname(dy) = 'Friday'
```

SQL Server

Рекурсивным оператором WITH возвращаем все дни текущего года. Затем с помощью функции DAYNAME выбираем только пятницы:

```
 1    with x (dy,yr)
 2      as (
 3  select dy, year(dy) yr
 4    from (
 5  select getdate()-datepart(dy,getdate())+1 dy
 6    from t1
 7         ) tmp1
 8   union all
 9  select dateadd(dd,1,dy), yr
10    from x
11   where year(dateadd(dd,1,dy)) = yr
12  )
13  select x.dy
14    from x
15   where datename(dw,x.dy) = 'Friday'
16  option (maxrecursion 400)
```

Обсуждение

DB2

Чтобы выбрать все пятницы текущего года, необходимо суметь возвратить все дни текущего года. Первый шаг – найти первый день года с помощью функции DAYOFYEAR. Вычитая значение, возвращенное DAYOFYEAR(CURRENT_DATE) из текущей даты, получаем 31 декабря предыдущего года. Затем добавляем 1, чтобы получить первый день текущего года:

```
select (current_date -
         dayofyear(current_date) days +1 days) as dy
  from t1
```

```
DY
-----------
01-JAN-2005
```

Имея первый день года, с помощью оператора WITH реализуем многократное добавление одного дня, начиная с первого дня года, до тех пор пока не дойдем до следующего года. Результирующее множество будет включать все дни текущего года (строки, возвращенные рекурсивным представлением X, частично показаны ниже):

```
  with x (dy,yr)
    as (
select dy, year(dy) yr
  from (
select (current_date -
         dayofyear(current_date) days +1 days) as dy
  from t1
       ) tmp1
union all
select dy+1 days, yr
  from x
 where year(dy +1 day) = yr
)
select dy
  from x
```

```
DY
-----------
01-JAN-2005
...
15-FEB-2005
...
22-NOV-2005
...
31-DEC-2005
```

Завершающий шаг – с помощью функции DAYNAME выбрать строки, соответствующие пятницам.

Oracle

Чтобы выбрать все пятницы текущего года, необходимо возвратить все дни текущего года. Сначала используем функцию TRUNC для получения первого дня года:

```
select trunc(sysdate,'y') dy
  from t1
```

```
DY
-----------
01-JAN-2005
```

Далее с помощью оператора CONNECT BY возвращаем все дни текущего года (чтобы понять, как генерировать строки с помощью CONNECT

BY, смотрите раздел «Формирование последовательности числовых значений» главы 10).

В данном рецепте используется оператор WITH, но можно также применять вложенный запрос.

На момент написания данной книги Oracle-оператор WITH не приспособлен для рекурсивных операций (в отличие от DB2 и SQL Server). Рекурсивные операции реализуются с помощью CONNECT BY. Результирующее множество, возвращаемое вложенным представлением X, частично показано ниже:

```
  with x
    as (
select trunc(sysdate,'y')+level-1 dy
  from t1
  connect by level <=
     add_months(trunc(sysdate,'y'),12)-trunc(sysdate,'y')
)
select *
  from x

DY
-----------
01-JAN-2005
...
15-FEB-2005
...
22-NOV-2005
...
31-DEC-2005
```

Завершающий шаг – с использованием функции TO_CHAR выбрать только пятницы.

PostgreSQL

Чтобы выбрать все пятницы текущего года, необходимо сначала получить все дни текущего года (каждый день в отдельной строке). Для этого используем функцию GENERATE_SERIES. Она должна возвратить диапазон значений от 0 до значения, равного количеству дней в текущем году минус 1. Первый передаваемый в GENERATE_SERIES параметр – 0, тогда как второй параметр – это запрос, определяющий количество дней в текущем году (поскольку полученное значение добавляется к первому дню года, необходимо добавить на 1 день меньше, чем есть в текущем году, чтобы не попасть в первый день следующего года). Результат, возвращаемый вторым параметром функции GENERATE_SERIES, показан ниже:

```
select cast(
       cast(
```

```
 date_trunc('year',current_date) as date)
           + interval '1 years' as date)
           - cast(
       date_trunc('year',current_date) as date)-1 as cnt
  from t1

CNT
---
364
```

Учитывая приведенный выше результат, вызов GENERATE_SERIES в операторе FROM будет выглядеть так: GENERATE_SERIES (0, 364). Если текущий год является високосным, как, например, 2004, то второй параметр будет равен 365.

После формирования списка дат года следующий шаг – добавление значений, возвращенных GENERATE_SERIES, к первому дню текущего года. Результаты частично показаны ниже:

```
select cast(date_trunc('year',current_date) as date)
       + x.id as dy
  from generate_series (
       0,
       ( select cast(
                cast(
           date_trunc('year',current_date) as date)
                     + interval '1 years' as date)
                     - cast(
                date_trunc('year',current_date) as date) )-1
       ) x(id)

DY
-----------
01-JAN-2005
...
15-FEB-2005
...
22-NOV-2005
...
31-DEC-2005
```

Завершающий шаг – с использованием функции TO_CHAR выбрать только пятницы.

MySQL

Чтобы выбрать все пятницы текущего года, необходимо возвратить все дни текущего года. Первый шаг – с помощью функции DAYOFYEAR найти первый день года. Вычитаем из текущей даты значение, возвращенное DAYOFYEAR(CURRENT_DATE), и затем добавляем 1, чтобы получить первый день текущего года:

```
select adddate(
       adddate(current_date,
```

```
                interval -dayofyear(current_date) day),
                interval 1 day ) dy
  from t1

DY
-----------
01-JAN-2005
```

Затем используем таблицу T500, чтобы получить количество строк, соответствующее количеству дней текущего года. Сделать это можно, последовательно добавляя значения столбца T500.ID к первому дню года, до тех пор пока не будет достигнут первый день следующего года. Результаты этой операции частично представлены ниже:

```
select adddate(x.dy,interval t500.id-1 day) dy
  from (
select dy, year(dy) yr
  from (
select adddate(
       adddate(current_date,
                interval -dayofyear(current_date) day),
                interval 1 day ) dy
  from t1
       ) tmp1
       ) x,
       t500
 where year(adddate(x.dy,interval t500.id-1 day)) = x.yr

DY
-----------
01-JAN-2005
...
15-FEB-2005
...
22-NOV-2005
...
31-DEC-2005
```

Завершающий шаг – с использованием функции DAYNAME выбрать только пятницы.

SQL Server

Чтобы выбрать все пятницы текущего года, необходимо получить все дни текущего года. Первый шаг – с помощью функции DATEPART найти первый день года. Вычитаем из текущей даты значение, возвращенное DATEPART(DY,GETDATE()), и затем добавляем 1, чтобы получить первый день текущего года

```
select getdate()-datepart(dy,getdate())+1 dy
  from t1

DY
-----------
01-JAN-2005
```

Имея первый день года, с помощью оператора WITH и функции DATEADD реализуем многократное добавление одного дня, начиная с первого дня года, до тех пор пока не будет получен первый день следующего года. Результирующее множество будет включать все дни текущего года (строки, возвращенные рекурсивным представлением X, частично показаны ниже):

```
with x (dy,yr)
  as (
select dy, year(dy) yr
  from (
select getdate()-datepart(dy,getdate())+1 dy
  from t1
       ) tmp1
 union all
select dateadd(dd,1,dy), yr
  from x
 where year(dateadd(dd,1,dy)) = yr
)
select x.dy
  from x
option (maxrecursion 400)
```

```
DY
-----------
01-JAN-2005
...
15-FEB-2005
...
22-NOV-2005
...
31-DEC-2005
```

Наконец, с помощью функции DATENAME выбираем только те строки, которые соответствуют пятницам. Чтобы данное решение было работоспособным, значение параметра MAXRECURSION должно быть задано не менее 366 (ограничение количества дней в году в рекурсивном представлении X гарантирует невозможность получения более 366 строк).

Определение дат первого и последнего появления заданного дня недели

Задача

Требуется найти, например, первый и последний понедельники текущего месяца.

Решение

Понедельник и текущий месяц выбраны в качестве примера. Представленные в данном рецепте решения могут использоваться для любого

дня недели и любого месяца. Поскольку один и тот же день недели повторяется каждые семь дней, получив первую соответствующую ему дату, можно добавить к ней 7 дней и найти вторую дату, а также добавить 14 дней и найти третью. Аналогично, если известна последняя соответствующая заданному дню недели дата месяца, вычитая 7 дней, получаем третью, а, вычитая 14, – вторую дату месяца.

DB2

Используя рекурсивный оператор WITH, получаем все дни текущего месяца. С помощью выражения CASE отмечаем все понедельники. Первым и последним понедельниками будут самая ранняя и самая поздняя из отмеченных дат:

```
 1    with x (dy,mth,is_monday)
 2      as (
 3  select dy,month(dy),
 4         case when dayname(dy)='Monday'
 5              then 1 else 0
 6         end
 7    from (
 8  select (current_date-day(current_date) day +1 day) dy
 9    from t1
10         ) tmp1
11   union all
12  select (dy +1 day), mth,
13         case when dayname(dy +1 day)='Monday'
14              then 1 else 0
15         end
16    from x
17   where month(dy +1 day) = mth
18  )
19  select min(dy) first_monday, max(dy) last_monday
20    from x
21   where is_monday = 1
```

Oracle

Чтобы найти первый и последний понедельники текущего месяца, используйте функции NEXT_DAY и LAST_DAY в сочетании с небольшим объемом операций над датами:

```
select next_day(trunc(sysdate,'mm')-1,'MONDAY') first_monday,
       next_day(last_day(trunc(sysdate,'mm'))-7,'MONDAY') last_monday
  from dual
```

PostgreSQL

С помощью функции DATE_TRUNC получите первый день месяца. После этого, используя простые вычисления над числовыми значениями дней недели (Вс.–Сб. соответствуют 1–7), находите первый и последний понедельники текущего месяца:

```
select first_monday,
       case to_char(first_monday+28,'mm')
            when mth then first_monday+28
                     else first_monday+21
       end as last_monday
  from (
select case sign(cast(to_char(dy,'d') as integer)-2)
            when  0
            then dy
            when -1
            then dy+abs(cast(to_char(dy,'d') as integer)-2)
            when  1
            then (7-(cast(to_char(dy,'d') as integer)-2))+dy
       end as first_monday,
       mth
  from (
select cast(date_trunc('month',current_date) as date) as dy,
       to_char(current_date,'mm') as mth
  from t1
       ) x
       ) y
```

MySQL

Используйте функцию ADDDATE, чтобы получить первый день месяца. После этого путем простых вычислений над числовыми значениями дней недели (Вс.–Сб. соответствуют 1–7) находите первый и последний понедельники текущего месяца:

```
select first_monday,
       case month(adddate(first_monday,28))
            when mth then adddate(first_monday,28)
                     else adddate(first_monday,21)
       end last_monday
  from (
select case sign(dayofweek(dy)-2)
            when  0 then dy
            when -1 then adddate(dy,abs(dayofweek(dy)-2))
            when  1 then adddate(dy,(7-(dayofweek(dy)-2)))
       end first_monday,
       mth
  from (
select adddate(adddate(current_date,-day(current_date)),1) dy,
       month(current_date) mth
  from t1
       ) x
       ) y
```

SQL Server

Используйте рекурсивный оператор WITH, чтобы получить все дни текущего месяца, и затем с помощью выражения CASE отметьте все

понедельники. Первым и последним понедельниками будут самая ранняя и самая поздняя из отмеченных дат:

```
 1    with x (dy,mth,is_monday)
 2      as (
 3  select dy,mth,
 4         case when datepart(dw,dy) = 2
 5              then 1 else 0
 6         end
 7    from (
 8  select dateadd(day,1,dateadd(day,-day(getdate()),getdate())) dy,
 9         month(getdate()) mth
10    from t1
11         ) tmp1
12   union all
13  select dateadd(day,1,dy),
14         mth,
15         case when datepart(dw,dateadd(day,1,dy)) = 2
16              then 1 else 0
17         end
18    from x
19   where month(dateadd(day,1,dy)) = mth
20  )
21  select min(dy) first_monday,
22         max(dy) last_monday
23    from x
24   where is_monday = 1
```

Обсуждение

DB2 и SQL Server

В DB2 и SQL Server для решения поставленной задачи используются разные функции, но методики аналогичны. Если внимательно рассмотреть решения, то единственное отличие будет найдено в способе суммирования дат. В данном обсуждении анализируются оба решения. Для демонстрации промежуточных шагов используется код решения для DB2.

Если в используемой версии SQL Server или DB2 рекурсивный оператор WITH не предоставляется, можно использовать технику, предлагаемую для PostgreSQL.

Первый шаг при поиске первого и последнего понедельников текущего месяца состоит в получении первого дня месяца. Его поиском занимается вложенный запрос TMP1 в рекурсивном представлении X. Сначала определяется текущая дата, а именно день месяца, соответствующий текущей дате. День месяца, соответствующий текущей дате, представляет, сколько дней месяца прошло (например, 10 апреля – это 10-й день апреля). Если вычесть это значение из текущей даты, получится последний день предыдущего месяца (например, в результате

вычитания 10 из 10-го апреля будем иметь последний день марта). После этого вычитания просто добавляем один день, чтобы получить первый день текущего месяца:

```
select (current_date-day(current_date) day +1 day) dy
  from t1

DY
-----------
01-JUN-2005
```

Далее, используя функцию MONTH, получаем месяц текущей даты и с помощью простого выражения CASE определяем, является ли первый день этого месяца понедельником или нет:

```
select dy, month(dy) mth,
        case when dayname(dy)='Monday'
             then 1 else 0
        end is_monday
   from (
 select (current_date-day(current_date) day +1 day) dy
   from t1
        ) tmp1

DY          MTH  IS_MONDAY
----------- --- ----------
01-JUN-2005   6          0
```

После этого используем рекурсивные возможности оператора WITH и последовательно добавляем по одному дню, начиная с первого дня месяца, до тех пор пока не дойдем до первого дня следующего месяца. В ходе этого с помощью выражения CASE отбираем понедельники (отмечая их флагом «1»). Результат рекурсивного представления X частично показан ниже:

```
with x (dy,mth,is_monday)
     as (
 select dy,month(dy) mth,
        case when dayname(dy)='Monday'
             then 1 else 0
        end is_monday
   from (
 select (current_date-day(current_date) day +1 day) dy
   from t1
        ) tmp1
  union all
 select (dy +1 day), mth,
        case when dayname(dy +1 day)='Monday'
             then 1 else 0
        end
   from x
  where month(dy +1 day) = mth
 )
```

```
select *
  from x

DY          MTH  IS_MONDAY
----------- --- ----------
01-JUN-2005  6           0
02-JUN-2005  6           0
03-JUN-2005  6           0
04-JUN-2005  6           0
05-JUN-2005  6           0
06-JUN-2005  6           1
07-JUN-2005  6           0
08-JUN-2005  6           0
...
```

Значения 1 столбца IS_MONDAY будут соответствовать понедельникам. Таким образом, заключительный шаг – применяя агрегатные функции MIN и MAX к строкам, в которых значение поля IS_MONDAY равно 1, находим первый и последний понедельники месяца.

Oracle

Функция NEXT_DAY значительно упрощает решение этой задачи. Чтобы найти первый понедельник текущего месяца, сначала посредством некоторых вычислений с привлечением функции TRUNC возвращаем последний день предыдущего месяца:

```
select trunc(sysdate,'mm')-1 dy
  from dual

DY
-----------
31-MAY-2005
```

Затем используем функцию NEXT_DAY, чтобы найти первый понедельник после последнего дня предыдущего месяца (т. е. первый понедельник текущего месяца):

```
select next_day(trunc(sysdate,'mm')-1,'MONDAY') first_monday
  from dual

FIRST_MONDAY
------------
06-JUN-2005
```

Чтобы найти последний понедельник текущего месяца, сначала с помощью функции TRUNC возвращаем первый день текущего месяца:

```
select trunc(sysdate,'mm') dy
  from dual

DY
-----------
01-JUN-2005
```

Следующий шаг – находим последнюю неделю (последние семь дней) месяца. С помощью функции LAST_DAY получаем последний день месяца и вычитаем семь дней:

```
select last_day(trunc(sysdate,'mm'))-7 dy
  from dual

DY
-----------
23-JUN-2005
```

Здесь мы возвращаемся на семь дней от последнего дня месяца, чтобы гарантированно получить, по крайней мере, по одному разу каждый день недели. Последний шаг – использовать функцию NEXT_DAY, чтобы найти следующий (и последний) понедельник месяца:

```
select next_day(last_day(trunc(sysdate,'mm'))-7,'MONDAY') last_monday
  from dual

LAST_MONDAY
-----------
27-JUN-2005
```

PostgreSQL и MySQL

В PostgreSQL и MySQL используется аналогичный подход, отличие состоит лишь в вызываемых функциях. Соответствующие запросы предельно просты, несмотря на их размеры; некоторые издержки возникают при поиске первого и последнего понедельников текущего месяца.

Первый шаг – найти первый день текущего месяца. Следующий шаг – определить первый понедельник месяца. Поскольку специальной функции, которая возвращала бы следующую дату, соответствующую заданному дню недели, нет, необходимо прибегнуть к небольшим вычислениям. Выражение CASE, начинающееся в строке 7 (любого решения), вычисляет разницу между числовым значением дня недели первого дня месяца и числовым значением, соответствующим понедельнику. Исходя из того, что функция TO_CHAR (PostgresSQL) с форматной маской 'D' или 'd' и функция DAYOFWEEK (MySQL) для дней недели от воскресенья до субботы возвращают числовые значения от 1 до 7, понедельник всегда будет представлен цифрой 2. Сначала выражение CASE проверяет знак (SIGN) полученной разности между первым днем месяца (каким бы он ни был) и числовым значением понедельника (2). Если результат равен 0, первый день месяца выпадает на понедельник, и это первый понедельник месяца. Если результат равен -1, первый день месяца выпадает на воскресенье, и чтобы найти первый понедельник, надо просто добавить в первому дню месяца разницу в днях между 2 и 1 (числовые значения понедельника и воскресенья соответственно).

Если возникают сложности с пониманием логики этих операций, забудьте о названиях дней недели и оперируйте только числами. Например, пусть первым днем месяца является

вторник, и необходимо найти следующую пятницу. При использовании функции TO_CHAR с форматной маской 'd' или функции DAYOFWEEK получаем 6 для пятницы и 3 для вторника. Чтобы получить 6 из 3, просто находим их разность (6 − 3 = 3) и прибавляем ее к меньшему значению ((6 − 3) + 3 = 6). Итак, независимо от реальных дат, если числовое значение дня, с которого начинается отсчет, меньше, чем числовое значение искомого дня, добавление разности между этими двумя днями к начальной дате даст искомую дату.

Если выражение SIGN дает в результате 1, первый день месяца выпадает на день между вторником и субботой (включительно). Если числовое значение первого дня месяца больше 2 (понедельник), вычтете из 7 разность между числовым значением первого дня месяца и числовым значением понедельника (2) и затем прибавьте полученное значение к первому дню месяца. Таким образом, получаем искомый день недели, в данном случае понедельник.

Опять же, если есть затруднения с пониманием логики операций, забудьте о названиях дней недели и просто оперируйте числами. Например, предположим, требуется найти следующий вторник, начиная с пятницы. Числовое значение вторника (3) меньше, чем значение пятницы (6). Чтобы попасть на третий день с шестого, вычитаем из 7 разность между этими двумя значениями (7 − (|3 − 6|) = 4) и добавляем результат (4) к начальному дню (пятнице). (Применение вертикальных черточек в выражении |3 − 6| обеспечивает получение абсолютного значения разности.) Здесь мы не добавляем 4 к 6 (что в результате дало бы 10), мы добавляем четыре дня к пятнице, что обеспечит возвращение следующего вторника.

Идея, стоящая за применением выражения CASE, заключается в создании своего рода функции «следующий день» для PostgreSQL и MySQL. Если вычисления начинаются не с первого дня месяца, в столбце DY будет располагаться значение, возвращенное функцией CURRENT_DATE. Выражение CASE в этом случае возвратит дату следующего после текущей даты понедельника (если сама текущая дата соответствует понедельнику, будет возвращена она).

Получив первый понедельник месяца, добавляем 21 или 28 дней, чтобы найти последний понедельник. Сколько дней необходимо добавить (21 или 28), определяет выражение CASE (строки 2–5). Оно добавляет к текущей дате 28 дней и проверяет, приводит ли это к переходу в следующий месяц. Выражение CASE осуществляет это следующим образом:

1. К значению, возвращенному функцией FIRST_MONDAY, добавляется 28.
2. С помощью функции TO_CHAR (PostgreSQL) или MONTH из результата выражения FIRST_MONDAY + 28 извлекается название получаемого месяца.

3. Результат шага 2 сравнивается со значением MTH из вложенного запроса. Значение MTH – это название текущего месяца, полученное в результате выполнения функции CURRENT_DATE. Если значения совпадают, следовательно, текущий месяц длинный и необходимо добавлять 28 дней; выражение CASE возвращает FIRST_MONDAY + 28. Если значения месяцев не совпадают, значит, при добавлении 28 дней мы выходим за рамки одного месяца, и выражение CASE возвращает FIRST_MONDAY + 21. Длительность наших месяцев такова, что проверять приходится только два возможных значения, 28 и 21. Это очень удобно.

Можно расширить решение, добавляя 7 и 14 дней, для поиска второго и третьего понедельников месяца соответственно.

Создание календаря

Задача

Требуется создать календарь на текущий месяц. Он должен быть отформатирован, как обычный календарь: семь столбцов в ширину и (как правило) пять строк вниз.

Решение

Решения будут выглядеть немного по-разному, но все они решают эту задачу одинаково: возвращают все дни текущего месяца и затем разделяют их на недели по одному из дней недели, создавая календарь.

Существуют разные форматы календарей. Например, команда *cal* UNIX форматирует неделю от воскресенья до субботы. В примерах данного рецепта используется стандартная нумерация недель ISO, поэтому удобнее всего будет создавать неделю, начиная с понедельника. Полностью разобравшись с решениями, вы поймете, что изменение формата календаря – это просто вопрос корректировки значений, определяемых ISO-номерами недель.

Как только мы начинаем использовать в SQL разные типы форматирования для создания удобных для чтения результатов, запросы становятся длиннее. Не позволяйте этим длинным запросам запутать вас. Представленные в данном рецепте запросы окажутся предельно простыми при разложении их на составляющие и выполнении одного за другим.

DB2

Чтобы возвратить все дни текущего месяца, используйте рекурсивный оператор WITH. Затем разбейте месяц на недели по выбранному дню с помощью выражений CASE и функций MAX:

```
 1   with x(dy,dm,mth,dw,wk)
 2     as (
 3 select (current_date -day(current_date) day +1 day) dy,
 4         day((current_date -day(current_date) day +1 day)) dm,
 5         month(current_date) mth,
 6         dayofweek(current_date -day(current_date) day +1 day) dw,
 7         week_iso(current_date -day(current_date) day +1 day) wk
 8   from t1
 9  union all
10 select dy+1 day, day(dy+1 day), mth,
11        dayofweek(dy+1 day), week_iso(dy+1 day)
12   from x
13  where month(dy+1 day) = mth
14  )
15  select max(case dw when 2 then dm end) as Mo,
16         max(case dw when 3 then dm end) as Tu,
17         max(case dw when 4 then dm end) as We,
18         max(case dw when 5 then dm end) as Th,
19         max(case dw when 6 then dm end) as Fr,
20         max(case dw when 7 then dm end) as Sa,
21         max(case dw when 1 then dm end) as Su
22    from x
23   group by wk
24   order by wk
```

Oracle

Чтобы возвратить все дни текущего месяца, используйте рекурсивный оператор CONNECT BY. Затем разбейте месяц на недели по выбранному дню с помощью выражений CASE и функций MAX:

```
 1  with x
 2    as (
 3  select *
 4    from (
 5  select to_char(trunc(sysdate,'mm')+level-1,'iw') wk,
 6         to_char(trunc(sysdate,'mm')+level-1,'dd') dm,
 7         to_number(to_char(trunc(sysdate,'mm')+level-1,'d')) dw,
 8         to_char(trunc(sysdate,'mm')+level-1,'mm') curr_mth,
 9         to_char(sysdate,'mm') mth
10    from dual
11   connect by level <= 31
12         )
13   where curr_mth = mth
14  )
15  select max(case dw when 2 then dm end) Mo,
16         max(case dw when 3 then dm end) Tu,
17         max(case dw when 4 then dm end) We,
18         max(case dw when 5 then dm end) Th,
19         max(case dw when 6 then dm end) Fr,
20         max(case dw when 7 then dm end) Sa,
21         max(case dw when 1 then dm end) Su
```

```
22    from x
23   group by wk
24   order by wk
```

PostgreSQL

Чтобы возвратить все дни текущего месяца, используйте функцию GENERATE_SERIES. Затем разбейте месяц на недели по выбранному дню с помощью выражений CASE и функций MAX:

```
 1  select max(case dw when 2 then dm end) as Mo,
 2         max(case dw when 3 then dm end) as Tu,
 3         max(case dw when 4 then dm end) as We,
 4         max(case dw when 5 then dm end) as Th,
 5         max(case dw when 6 then dm end) as Fr,
 6         max(case dw when 7 then dm end) as Sa,
 7         max(case dw when 1 then dm end) as Su
 8    from (
 9  select *
10    from (
11  select cast(date_trunc('month',current_date) as date)+x.id,
12         to_char(
13            cast(
14      date_trunc('month',current_date)
15                 as date)+x.id,'iw') as wk,
16         to_char(
17            cast(
18      date_trunc('month',current_date)
19                 as date)+x.id,'dd') as dm,
20         cast(
21      to_char(
22         cast(
23   date_trunc('month',current_date)
24                 as date)+x.id,'d') as integer) as dw,
25         to_char(
26            cast(
27      date_trunc('month',current_date)
28                 as date)+x.id,'mm') as curr_mth,
29         to_char(current_date,'mm') as mth
30    from generate_series (0,31) x(id)
31         ) x
32   where mth = curr_mth
33         ) y
34   group by wk
35   order by wk
```

MySQL

Чтобы возвратить все дни текущего месяца, используйте таблицу T500. Затем разбейте месяц на недели по выбранному дню с помощью выражений CASE и функций MAX:

```
select max(case dw when 2 then dm end) as Mo,
       max(case dw when 3 then dm end) as Tu,
       max(case dw when 4 then dm end) as We,
       max(case dw when 5 then dm end) as Th,
       max(case dw when 6 then dm end) as Fr,
       max(case dw when 7 then dm end) as Sa,
       max(case dw when 1 then dm end) as Su
  from (
select date_format(dy,'%u') wk,
       date_format(dy,'%d') dm,
       date_format(dy,'%w')+1 dw
  from (
select adddate(x.dy,t500.id-1) dy,
       x.mth
  from (
select adddate(current_date,-dayofmonth(current_date)+1) dy,
       date_format(
           adddate(current_date,
                   -dayofmonth(current_date)+1),
                   '%m') mth
  from t1
       ) x,
         t500
 where t500.id <= 31
   and date_format(adddate(x.dy,t500.id-1),'%m') = x.mth
       ) y
       ) z
 group by wk
 order by wk
```

SQL Server

Чтобы возвратить все дни текущего месяца, используйте рекурсивный оператор WITH. Затем разбейте месяц на недели по выбранному дню с помощью выражений CASE и функций MAX:

```
  with x(dy,dm,mth,dw,wk)
    as (
select dy,
       day(dy) dm,
       datepart(m,dy) mth,
       datepart(dw,dy) dw,
       case when datepart(dw,dy) = 1
            then datepart(ww,dy)-1
            else datepart(ww,dy)
       end wk
  from (
select dateadd(day,-day(getdate())+1,getdate()) dy
  from t1
       ) x
 union all
 select dateadd(d,1,dy), day(dateadd(d,1,dy)), mth,
```

```
17          datepart(dw,dateadd(d,1,dy)),
18          case when datepart(dw,dateadd(d,1,dy)) = 1
19               then datepart(wk,dateadd(d,1,dy))-1
20               else datepart(wk,dateadd(d,1,dy))
21          end
22     from x
23    where datepart(m,dateadd(d,1,dy)) = mth
24  )
25  select max(case dw when 2 then dm end) as Mo,
26         max(case dw when 3 then dm end) as Tu,
27         max(case dw when 4 then dm end) as We,
28         max(case dw when 5 then dm end) as Th,
29         max(case dw when 6 then dm end) as Fr,
30         max(case dw when 7 then dm end) as Sa,
31         max(case dw when 1 then dm end) as Su
32    from x
33   group by wk
34   order by wk
```

Обсуждение

DB2

Первый шаг – возвратить все дни месяца, для которого создается календарь. Для этого используется рекурсивный оператор WITH (если WITH недоступен, можно применять сводную таблицу, например, T500, как в решении для MySQL). Кроме всех дней месяца (столбец под псевдонимом DM), понадобится возвратить разные составляющие каждой даты: порядковый номер дня недели (под псевдонимом DW), текущий месяц (под псевдонимом MTH) и ISO-номер недели для каждого дня месяца (под псевдонимом WK). Результаты, возвращаемые рекурсивным представлением X до проведения рекурсии (верхняя часть оператора UNION ALL), показаны ниже:

```
select (current_date -day(current_date) day +1 day) dy,
       day((current_date -day(current_date) day +1 day)) dm,
       month(current_date) mth,
       dayofweek(current_date -day(current_date) day +1 day) dw,
       week_iso(current_date -day(current_date) day +1 day) wk
  from t1

DY          DM MTH         DW WK
----------- -- --- ---------- --
01-JUN-2005 01  06          4 22
```

Далее пошагово увеличиваем значение столбца DM (перебираем дни месяца), до тех пор пока не будут возвращены все дни данного месяца. Для каждого дня месяца при этом также будут получены соответствующий ему день недели и ISO-номер недели, в которую попадает данный день. Результаты частично показаны ниже:

```
with x(dy,dm,mth,dw,wk)
  as (
```

```
select (current_date -day(current_date) day +1 day) dy,
       day((current_date -day(current_date) day +1 day)) dm,
       month(current_date) mth,
       dayofweek(current_date -day(current_date) day +1 day) dw,
       week_iso(current_date -day(current_date) day +1 day) wk
  from t1
 union all
 select dy+1 day, day(dy+1 day), mth,
        dayofweek(dy+1 day), week_iso(dy+1 day)
   from x
  where month(dy+1 day) = mth
)
select *
  from x
```

```
DY          DM MTH         DW WK
----------- -- --- ---------- --
01-JUN-2005 01  06          4 22
02-JUN-2005 02  06          5 22
...
21-JUN-2005 21  06          3 25
22-JUN-2005 22  06          4 25
...
30-JUN-2005 30  06          5 26
```

На данном этапе мы получаем все дни текущего месяца, а также следующие данные для каждого дня: двузначный номер дня месяца, двузначный номер месяца, однозначный номер дня недели (1–7 для Вс.–Сб.) и двузначный ISO-номер недели. Имея в своем распоряжении всю эту информацию, с помощью выражения CASE можно определить, на какой день недели выпадает каждое из значений столбца DM (каждый день месяца). Результаты частично показаны ниже:

```
with x(dy,dm,mth,dw,wk)
  as (
select (current_date -day(current_date) day +1 day) dy,
       day((current_date -day(current_date) day +1 day)) dm,
       month(current_date) mth,
       dayofweek(current_date -day(current_date) day +1 day) dw,
       week_iso(current_date -day(current_date) day +1 day) wk
  from t1
 union all
 select dy+1 day, day(dy+1 day), mth,
        dayofweek(dy+1 day), week_iso(dy+1 day)
   from x
  where month(dy+1 day) = mth
)
select wk,
       case dw when 2 then dm end as Mo,
       case dw when 3 then dm end as Tu,
       case dw when 4 then dm end as We,
       case dw when 5 then dm end as Th,
```

```
       case dw when 6 then dm end as Fr,
       case dw when 7 then dm end as Sa,
       case dw when 1 then dm end as Su
  from x

WK MO TU WE TH FR SA SU
-- -- -- -- -- -- -- --
22       01
22          02
22             03
22                04
22                   05
23 06
23    07
23       08
23          09
23             10
23                11
23                   12
```

Как видно из частично представленных результатов, каждый день недели возвращается в отдельной строке. Теперь осталось только сгруппировать дни по неделям и собрать все дни одной недели в одну строку. Чтобы возвратить все дни недели в одной строке, используйте агрегатную функцию MAX и группировку по значениям столбца WK (ISO-номеру недели). Чтобы правильно отформатировать календарь и обеспечить верное расположение дней, упорядочиваем результаты по WK. Окончательный результат приведен ниже:

```
with x(dy,dm,mth,dw,wk)
  as (
select (current_date -day(current_date) day +1 day) dy,
       day((current_date -day(current_date) day +1 day)) dm,
       month(current_date) mth,
       dayofweek(current_date -day(current_date) day +1 day) dw,
       week_iso(current_date -day(current_date) day +1 day) wk
  from t1
 union all
 select dy+1 day, day(dy+1 day), mth,
        dayofweek(dy+1 day), week_iso(dy+1 day)
   from x
  where month(dy+1 day) = mth
)
select max(case dw when 2 then dm end) as Mo,
       max(case dw when 3 then dm end) as Tu,
       max(case dw when 4 then dm end) as We,
       max(case dw when 5 then dm end) as Th,
       max(case dw when 6 then dm end) as Fr,
       max(case dw when 7 then dm end) as Sa,
       max(case dw when 1 then dm end) as Su
  from x
```

```
 group by wk
 order by wk
```

```
MO TU WE TH FR SA SU
-- -- -- -- -- -- --
      01 02 03 04 05
06 07 08 09 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
```

Oracle

В Oracle 9*i* Database начинаем с использования рекурсивного оператора CONNECT BY, с помощью которого получаем столько строк, сколько дней в заданном месяце. В более ранних версиях СУБД CONNECT BY не может таким образом использоваться. Вместо этого можно применить сводную таблицу, например T500, как в решении для MySQL.

Для каждого дня месяца понадобится дополнительно получить следующие данные: порядковый номер дня месяца (под псевдонимом DM), день недели (под псевдонимом DW), текущий месяц (под псевдонимом MTH) и ISO-номер недели для каждого дня месяца (под псевдонимом WK). Результаты, возвращаемые представлением X оператора WITH для первого дня текущего месяца, показаны ниже:

```
select trunc(sysdate,'mm') dy,
       to_char(trunc(sysdate,'mm'),'dd') dm,
       to_char(sysdate,'mm') mth,
       to_number(to_char(trunc(sysdate,'mm'),'d')) dw,
       to_char(trunc(sysdate,'mm'),'iw') wk
  from dual
```

```
DY          DM MT         DW WK
----------- -- -- ---------- --
01-JUN-2005 01 06          4 22
```

Далее пошагово увеличиваем значение DM (перебираем дни месяца) до тех пор, пока не будут получены все дни текущего месяца. Для каждого дня месяца также возвращаем соответствующий ему день недели и ISO-номер недели, в которую попадает данный день. Результаты частично показаны ниже (полная дата для каждого дня приведена для ясности):

```
with x
  as (
select *
  from (
select trunc(sysdate,'mm')+level-1 dy,
       to_char(trunc(sysdate,'mm')+level-1,'iw') wk,
       to_char(trunc(sysdate,'mm')+level-1,'dd') dm,
       to_number(to_char(trunc(sysdate,'mm')+level-1,'d')) dw,
       to_char(trunc(sysdate,'mm')+level-1,'mm') curr_mth,
```