

SQLで移動平均を算出

2020/01/08

平均価格納行がおかしかったので修正。

2020/01/08

職場のTRSHM君に煽られたんでちょっと書く。

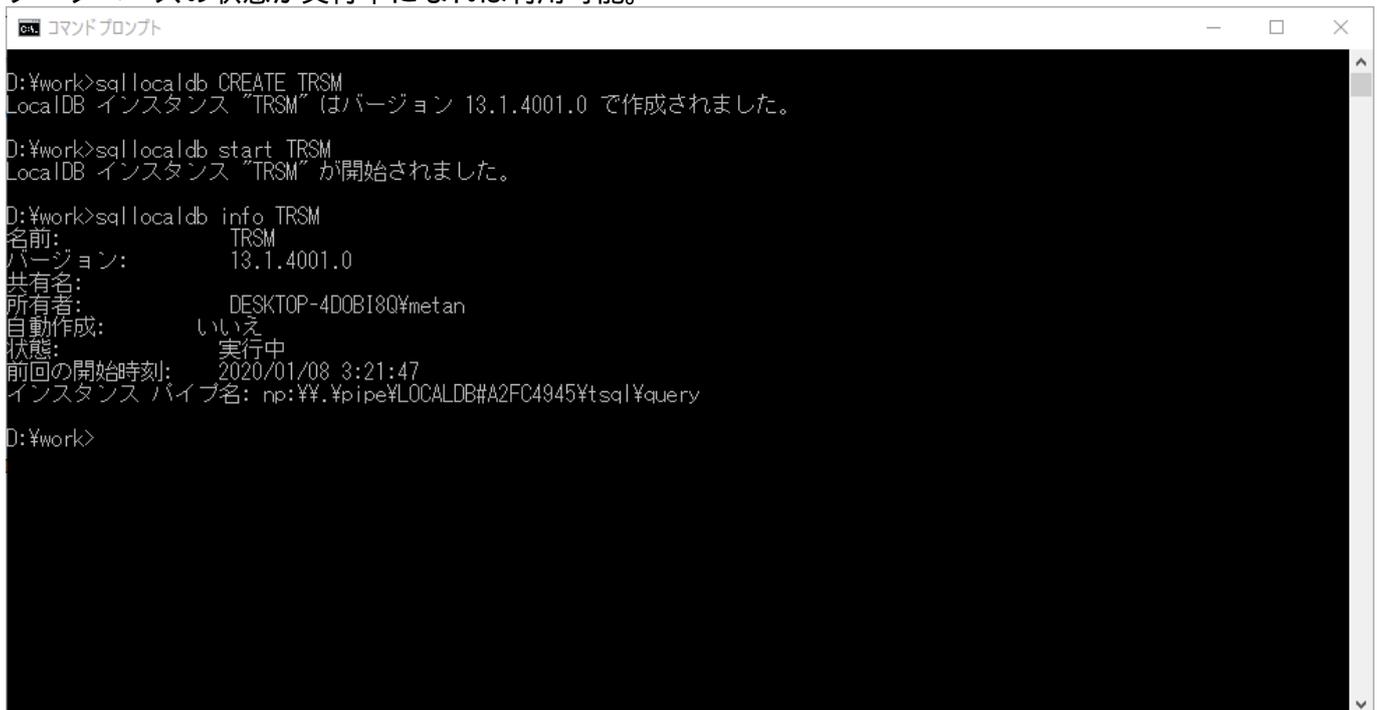
概要

- 直近3日の移動平均を出すサンプルコードを掲示する。
- ウィンドウ関数を利用する。サブクエリを連ねるのはさすがに古い。

データベース準備

Visual Studioをインストールしていれば多分SQLServer LocalDBが使えると思うので、こちらを利用する。

データベースの状態が実行中になれば利用可能。



```
コマンドプロンプト
D:\work>sqllocaldb CREATE TRSM
LocalDB インスタンス "TRSM" (バージョン 13.1.4001.0 で作成されました。

D:\work>sqllocaldb start TRSM
LocalDB インスタンス "TRSM" が開始されました。

D:\work>sqllocaldb info TRSM
名前: TRSM
バージョン: 13.1.4001.0
共有名:
所有者: DESKTOP-4DOB18Q\metan
自動作成: いいえ
状態: 実行中
前回の開始時刻: 2020/01/08 3:21:47
インスタンス パイプ名: np:¥¥.¥pipe¥LOCALDB#\A2FC4945¥tsql¥query

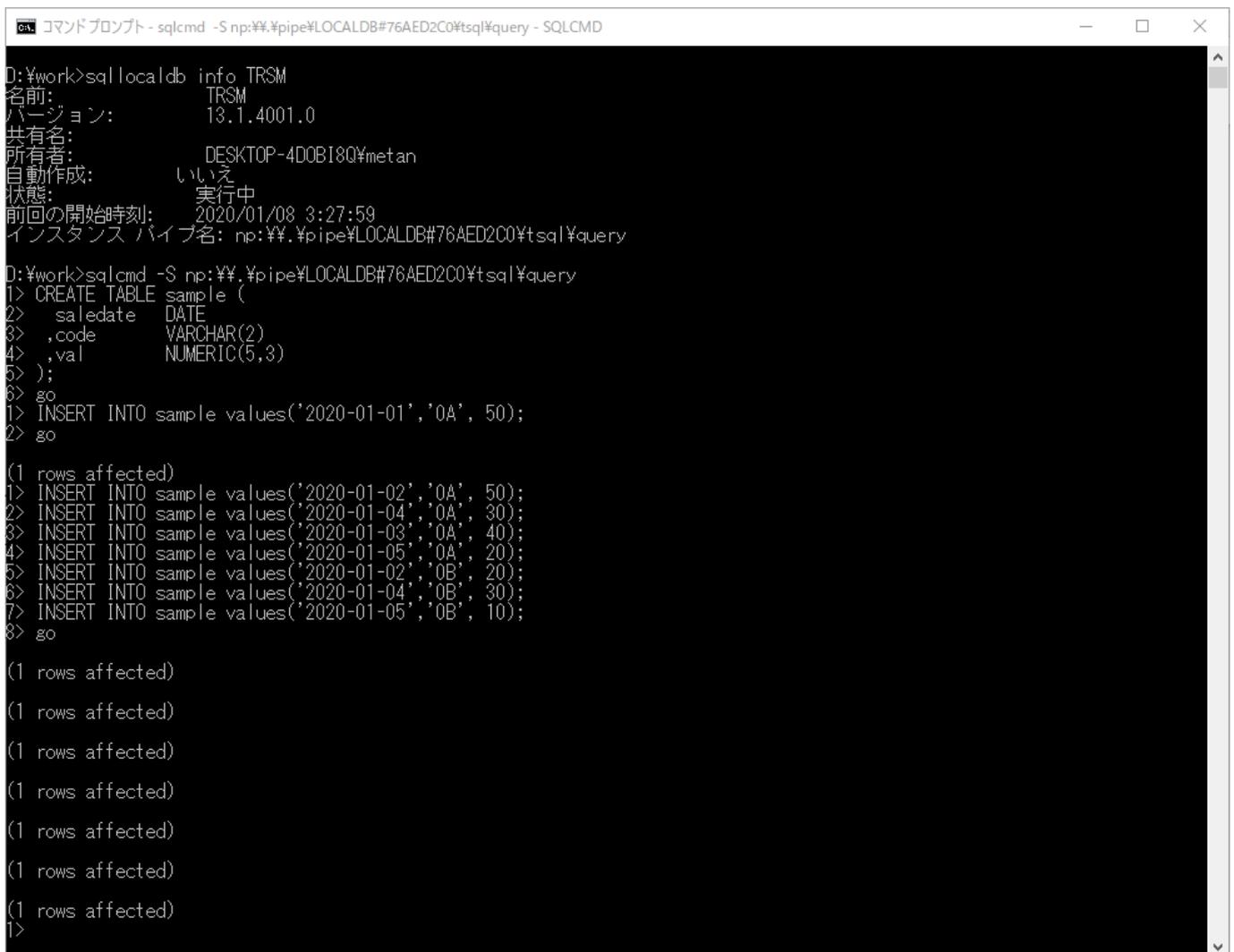
D:\work>
```

テーブルとデータの用意

sqlcmdでインスタンスパイプ名を指定して接続。あとは普通にSQLを実行してデータを定義する。

```
CREATE TABLE sample (
  saledate DATE
, code VARCHAR(2)
```

```
,val      NUMERIC(5,3)
);
GO
INSERT INTO sample VALUES('2020-01-01','0A', 50);
GO
INSERT INTO sample VALUES('2020-01-02','0A', 50);
INSERT INTO sample VALUES('2020-01-04','0A', 30);
INSERT INTO sample VALUES('2020-01-03','0A', 40);
INSERT INTO sample VALUES('2020-01-05','0A', 20);
INSERT INTO sample VALUES('2020-01-02','0B', 20);
INSERT INTO sample VALUES('2020-01-04','0B', 30);
INSERT INTO sample VALUES('2020-01-05','0B', 10);
GO
```



OVER句に対象レコードの範囲を指定する

PARTITION BY句で範囲を決め、ORDER BY句でレコードの並びを決めたのちROWS句で処理対象レコードを明示する。

以下のクエリだと、

- 2つ前のレコード 1つ前のレコード

- 1つ前のレコード 今選択されているレコード
- 今選択されているレコード 1つ後のレコード

の3レコードが処理対象になる。日付でソートして“ n 日前”や“ n 日後”をとれるようにすること。

```
SELECT saledate
      ,code
      ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1
preceding AND 1 following) AS "moved_avg"
FROM sample
GO
```

ただ、単純にこんなのを書くとはまる。

AVG()で指定する項目がNULLだった場合、そのレコードのカウントが入らないので、例えばcode='0A'の2020-01-01の“moved_avg”は33.333ではなく50.000になってしまう。

※NULL, 50, 50 となるので、 $100 \div 3$ ではなく $100 \div 2$ になる。

code='0B'なんか歯抜けがあるのでこれも正しくない。

```

C:\> コマンドプロンプト - sqlcmd -S np:¥¥.¥pipe¥LOCALDB#76AED2C0¥tsql¥query - SQLCMD
D:¥work>sqlcmd -S np:¥¥.¥pipe¥LOCALDB#76AED2C0¥tsql¥query
1> SELECT * FROM sample ORDER BY code,saledate
2> go
saledate          code val
-----
2020-01-01 0A    50.000
2020-01-02 0A    50.000
2020-01-03 0A    40.000
2020-01-04 0A    30.000
2020-01-05 0A    20.000
2020-01-02 0B    20.000
2020-01-04 0B    30.000
2020-01-05 0B    10.000

(8 rows affected)
1> SELECT saledate
      ,code
      ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1 preceding AND 1 following) as "moved_avg"
4> FROM sample
5> go
saledate          code moved_avg
-----
2020-01-01 0A    50.000000
2020-01-02 0A    46.666666
2020-01-03 0A    40.000000
2020-01-04 0A    30.000000
2020-01-05 0A    25.000000
2020-01-02 0B    25.000000
2020-01-04 0B    20.000000
2020-01-05 0B    20.000000

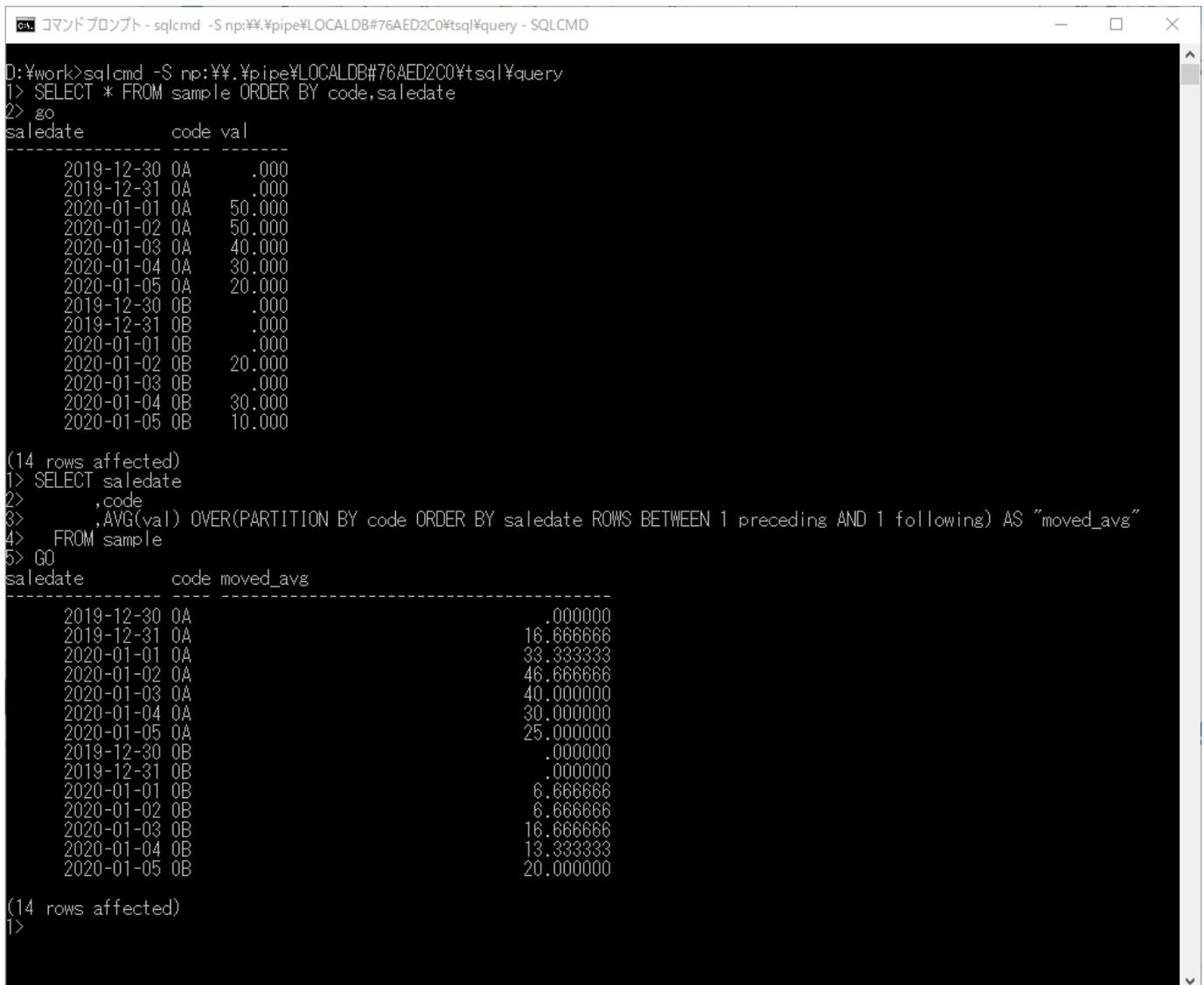
(8 rows affected)
1>

```

NULL値の対応を入れる

欠損レコードを補いNULL値の影響を消し込む。下手な小細工するより早い。

```
INSERT INTO sample VALUES ('2019-12-31', '0A', 0);
INSERT INTO sample VALUES ('2019-12-30', '0A', 0);
INSERT INTO sample VALUES ('2019-12-31', '0B', 0);
INSERT INTO sample VALUES ('2019-12-30', '0B', 0);
INSERT INTO sample VALUES ('2020-01-01', '0B', 0);
INSERT INTO sample VALUES ('2020-01-03', '0B', 0);
GO
SELECT saledate
       ,code
       ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1
preceding AND 1 following) AS "moved_avg"
FROM sample
GO
```



```
コマンドプロンプト - sqlcmd -S np:¥¥.¥pipe¥LOCALDB#76AED2C0¥tsql¥query - SQLCMD
D:¥work>sqlcmd -S np:¥¥.¥pipe¥LOCALDB#76AED2C0¥tsql¥query
1> SELECT * FROM sample ORDER BY code,saledate
2> go
saledate      code val
-----
2019-12-30 0A      .000
2019-12-31 0A      .000
2020-01-01 0A     50.000
2020-01-02 0A     50.000
2020-01-03 0A     40.000
2020-01-04 0A     30.000
2020-01-05 0A     20.000
2019-12-30 0B      .000
2019-12-31 0B      .000
2020-01-01 0B      .000
2020-01-02 0B     20.000
2020-01-03 0B      .000
2020-01-04 0B     30.000
2020-01-05 0B     10.000

(14 rows affected)
1> SELECT saledate
       ,code
       ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1 preceding AND 1 following) AS "moved_avg"
4> FROM sample
5> GO
saledate      code moved_avg
-----
2019-12-30 0A      .000000
2019-12-31 0A     16.666666
2020-01-01 0A     33.333333
2020-01-02 0A     46.666666
2020-01-03 0A     40.000000
2020-01-04 0A     30.000000
2020-01-05 0A     25.000000
2019-12-30 0B      .000000
2019-12-31 0B      .000000
2020-01-01 0B      6.666666
2020-01-02 0B      6.666666
2020-01-03 0B     16.666666
2020-01-04 0B     13.333333
2020-01-05 0B     20.000000

(14 rows affected)
1>
```

ダミーレコードは出したくない

ダミーレコードを出したくないならテーブルを拡張して対応すればいいと思う。
この例だと計算結果の行セットVを作り、ここからダミー行を除いている。

```
DROP TABLE sample
GO
CREATE TABLE sample (
    saledate    DATE
,code         VARCHAR(2)
,val          NUMERIC(5,3)
,disp        VARCHAR(1)
);
GO
INSERT INTO sample VALUES ('2020-01-01', '0A', 50, 'Y');
INSERT INTO sample VALUES ('2020-01-02', '0A', 50, 'Y');
INSERT INTO sample VALUES ('2020-01-04', '0A', 30, 'Y');
INSERT INTO sample VALUES ('2020-01-03', '0A', 40, 'Y');
INSERT INTO sample VALUES ('2020-01-05', '0A', 20, 'Y');
INSERT INTO sample VALUES ('2020-01-02', '0B', 20, 'Y');
INSERT INTO sample VALUES ('2020-01-04', '0B', 30, 'Y');
INSERT INTO sample VALUES ('2020-01-05', '0B', 10, 'Y');
INSERT INTO sample VALUES ('2019-12-31', '0A', 0, 'N');
INSERT INTO sample VALUES ('2019-12-30', '0A', 0, 'N');
INSERT INTO sample VALUES ('2019-12-31', '0B', 0, 'N');
INSERT INTO sample VALUES ('2019-12-30', '0B', 0, 'N');
INSERT INTO sample VALUES ('2020-01-01', '0B', 0, 'N');
INSERT INTO sample VALUES ('2020-01-03', '0B', 0, 'N');
GO
WITH V AS (
    SELECT saledate
           ,code
           ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1
preceding AND 1 following) AS "moved_avg"
           ,disp
    FROM sample
)
SELECT saledate
       ,code
       ,moved_avg
FROM V
WHERE disp='Y'
GO
```

```

C:\ コマンドプロンプト - sqlcmd -S np:pipe\LOCALDB#\76AED2C0\tsql\query - SQLCMD
1> SELECT * FROM sample ORDER BY code,saledate
2> go
saledate      code val      disp
-----
2019-12-30 0A      .000 N
2019-12-31 0A      .000 N
2020-01-01 0A     50.000 Y
2020-01-02 0A     50.000 Y
2020-01-03 0A     40.000 Y
2020-01-04 0A     30.000 Y
2020-01-05 0A     20.000 Y
2019-12-30 0B      .000 N
2019-12-31 0B      .000 N
2020-01-01 0B      .000 N
2020-01-02 0B     20.000 Y
2020-01-03 0B      .000 N
2020-01-04 0B     30.000 Y
2020-01-05 0B     10.000 Y

(14 rows affected)
1> WITH V as (
2>   SELECT saledate
3>         ,code
4>         ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1 preceding AND 1 following) as "moved_avg"
5>         ,disp
6>   FROM sample
7> )
8> SELECT saledate
9>        ,code
10>       ,moved_avg
11>   FROM V
12>  WHERE disp='Y'
13> go
saledate      code moved_avg
-----
2020-01-01 0A      33.333333
2020-01-02 0A      46.666666
2020-01-03 0A      40.000000
2020-01-04 0A      30.000000
2020-01-05 0A      25.000000
2020-01-02 0B       6.666666
2020-01-04 0B      13.333333
2020-01-05 0B      20.000000

(8 rows affected)
1>

```

もし先頭日と最終日を除きたければ行セットVからその日付を除けばいい。

```

WITH V AS (
  SELECT saledate
        ,code
        ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1
preceding AND 1 following) AS "moved_avg"
        ,disp
  FROM sample
)
SELECT saledate
      ,code
      ,moved_avg
FROM V
WHERE disp='Y'
      AND saledate BETWEEN '2020-01-02' AND '2020-01-04'
GO

```

```
コマンドプロンプト - sqlcmd -S np:¥¥.¥pipe¥LOCALDB#76AED2C0¥tsql¥query - SQLCMD
D:¥work>sqlcmd -S np:¥¥.¥pipe¥LOCALDB#76AED2C0¥tsql¥query
1> SELECT * FROM sample ORDER BY code,saledate
2> go
saledate      code val      disp
-----
2019-12-30 0A      .000 N
2019-12-31 0A      .000 N
2020-01-01 0A     50.000 Y
2020-01-02 0A     50.000 Y
2020-01-03 0A     40.000 Y
2020-01-04 0A     30.000 Y
2020-01-05 0A     20.000 Y
2019-12-30 0B      .000 N
2019-12-31 0B      .000 N
2020-01-01 0B      .000 N
2020-01-02 0B     20.000 Y
2020-01-03 0B      .000 N
2020-01-04 0B     30.000 Y
2020-01-05 0B     10.000 Y

(14 rows affected)
1> WITH V AS (
2>   SELECT saledate
3>         ,code
4>         ,AVG(val) OVER(PARTITION BY code ORDER BY saledate ROWS BETWEEN 1 preceding AND 1 following) AS "moved_avg"
5>         ,disp
6>   FROM sample
7> )
8> SELECT saledate
9>        ,code
10>       ,moved_avg
11> FROM V
12> WHERE disp='Y'
13>       AND saledate BETWEEN '2020-01-02' AND '2020-01-04'
14> go
saledate      code moved_avg
-----
2020-01-02 0A      46.666666
2020-01-03 0A      40.000000
2020-01-04 0A      30.000000
2020-01-02 0B       6.666666
2020-01-04 0B     13.333333

(5 rows affected)
1>
```

database, SQL, 不足レコード補填, SQLServer, 技術資料

From:

<https://wiki.hgotoh.jp/> - 努力したWiki

Permanent link:

<https://wiki.hgotoh.jp/documents/database/sql-0011>

Last update: **2023/04/14 02:32**

