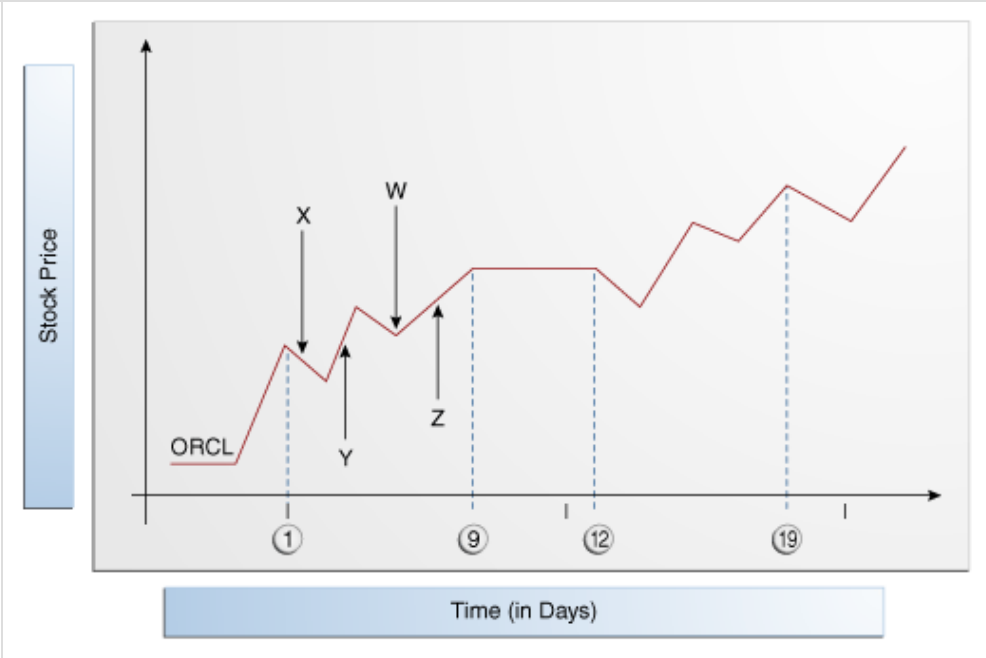
Oracle SQL - CEP语法梳理

实时处理中自然少不了CEP这个重量级功能。其实已经有不少成熟的框架实现了CEP，如Github siddhi和EsperTech，但他们使用了特有的语法来做到开发者的友好交互。那么，把CEP的功能加入到万能的SQL中，是不是很有意思呢?

详见Oracle文档，本篇博客语法解释及示例均来自官方文档。

速览

先来一个简单的例子，下面是一个股票实时折线图。



在这个图中，我们需要实时捕获X-Y-W-Z（跌-涨-跌-涨）的趋势，拿到趋势起始点和终止点。此时，SQL如下，出现了很多不常见的SQL语法对吧?
接下来我们逐个揭晓。

```
SELECT
    T.A,
    T.lastZ
FROM
    STREAM
MATCH_RECOGNIZE (
    MEASURES
        A,
        last(Z) as lastZ
```

```
    PATTERN(A W+ X+ Y+ Z+)
    DEFINE
        W as W.price < prev(W.price),
        X as X.price > prev(X.price),
        Y as Y.price < prev(Y.price),
        Z as Z.price > prev(Z.price)
) as T
```

MATCH_RECOGNIZE

作为CEP的标识符而存在，通常使用 **SELECT ... FROM S MATCH_RECOGNIZE (....) as T WHERE ...** 来表示一个CEP的操作。

MEASURES

生成T的指标，指标中的事件可以选择**DEFINE**中已经出现过的事件。可以使用示例中的Z.price也可以使用如sum等UDAP函数。

PATTERN

示例中的 **(A W+ X+ Y+ Z+)** 表示了严格的事件顺序，在这里表示A后面可以跟随N个W事件，last W后跟随N个X事件，last X后跟随N个Y事件，last Y后跟随N个Z事件。

贪婪	非贪婪	含义
*	*?	匹配0~N个
+	+?	匹配1~N个
?	??	匹配0,1个

贪婪和非贪婪举例如下，如**(A B* C)**，如果某事件同时满足B和C在DEFINE中的条件，那么此时会将此事件归因为B，若B使用非贪婪*?，则此事件归因为C。
这里还可以采用正则表达式，如**(A B+ | C)**这样的形式。

DEFINE

事件的定义，业务逻辑写在这一步，如 **W as W.price < prev(W.price)**，即W事件需要满足价格低于上一个W事件价格的条件。

PARTITION BY

数据分区，类似窗口函数用法。假设之前的折线图，我们要按照股票行业来划分，SQL可以写成如下形式：

```
PARTITION BY
    A.industry
```

```
MEASURES
    A,
    last(Z) as lastZ
PATTERN(A W+ X+ Y+ Z+)
DEFINE
    W as W.price < prev(W.price),
    X as X.price > prev(X.price),
    Y as Y.price < prev(Y.price),
    Z as Z.price > prev(Z.price)
```

DURATION

在示例中，你不希望曲线过于震荡，你的期望是，在10分钟内只发生W，X，Y，Z的趋势，在趋势完成后，不要再出现其他的转折点。那么此时可以限定10分钟内，不会有其他的事件掺入。

```
MEASURES
    A,
    last(Z) as lastZ
include timer events // 和duration 10配合使用
PATTERN(A W+ X+ Y+ Z+)
duration 10
DEFINE
    W as W.price < prev(W.price),
    X as X.price > prev(X.price),
    Y as Y.price < prev(Y.price),
    Z as Z.price > prev(Z.price)
```

WITHIN

在示例中，你不希望用一天去等待这个趋势，你期望在10分钟之内达到你的条件时，这个趋势才是有效的。注意WITHIN和WITHIN INCLUSIVE两种用法，对边界的包含是不同的。

```
MEASURES
    A,
    last(Z) as lastZ
PATTERN(A W+ X+ Y+ Z+)
DEFINE
    W as W.price < prev(W.price),
    X as X.price > prev(X.price),
    Y as Y.price < prev(Y.price),
    Z as Z.price > prev(Z.price)
```

ALL MATCHES

以股票折线图为例，在X的下降折线中，随意选择一点都满足DEFINE中的条件，此时ALL MATCHES会将他们全都匹配到。如果不设置ALL MATCHES，匹配到后处理程序会自动跳到最后一个事件的位置。