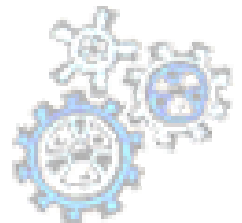


**Semantics of Proposed ANSI Specs  
for  
Event Queries in SQL—  
vs.  
composite events of active DBs**

**Y. Bai, H. Thakkar, C. Zaniolo**

**UCLA**



ISO/IEC JTC1/SC32 WG3:URC-*nnn*  
ANSI NCITS H2-2006-*nnn*

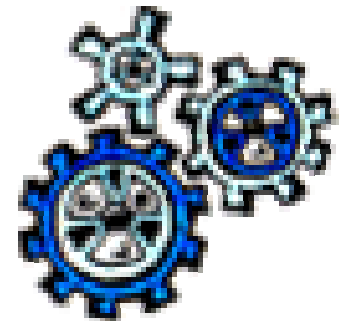
# Pattern Matching in Sequences of Rows

March 2, 2007 Change Proposal  
(for SQL standards)

**Authors:** Fred Zemke (Oracle), Andrew Witkowski (Oracle),  
Mitch Cherniak (Streambase), Latha Colby (IBM)

**Discussion Forum—the Tom Kyte blog:**

<http://tkyte.blogspot.com/2007/04/so-in-your-opinion.html>



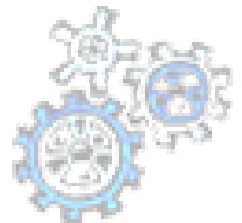
# SQL-TS

- ⌘ An SQL extension for complex events [Reza Sadri's thesis 2001—circa]
- ⌘ **Basic Idea:** use **regular expressions** for defining/detecting complex events
- ⌘ **Optimization** of FSA-based implementation based on a generalization of the Knuth, Morris and Pratt string-search algorithm.

E.g. if we search for:

a b c d a f

and we fail at f we should not restart the search from b--we can restart from the 2<sup>nd</sup> occurrence of a.



# Match\_Recognize (MR)

For instance:

$A^*$  — 0 or more matches of event  $A$

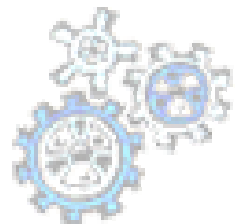
$C^+$  — 1 or more matches of event  $C$

$D ?$  — 0 or 1 match of event  $D$

$\{ n \}$  — exactly  $n$  matches

$\{ n, m \}$  — between  $n$  and  $m$  (inclusive) matches

More ...



```

SELECT a_symbol,
       a_tstamp,      /* start time */
       a_price,       /* start price */
       max_c_tstamp, /* inflection time */
       last_c_price,  /* low price */
       max_f_tstamp, /* end time */
       last_c_price,  /* end price */
       Matchno
FROM Ticker MATCH_RECOGNIZE (
  PARTITION BY Symbol
  ORDER BY Tstamp
  MEASURES A.Symbol AS a_symbol,
           A.Tstamp AS a_tstamp,
           A.Price AS a_price,
           MAX (C.Tstamp) AS max_c_tstamp,
           LAST (C.Price) AS last_c_price
           MAX (F.Tstamp) AS max_f_tstamp
  MATCH_NUMBER AS matchno

```

ONE ROW PER MATCH

AFTER MATCH SKIP PAST LAST ROW

MAXIMAL MATCH

PATTERN (A B C\* D E\* F+)

DEFINE /\* A defaults to True, matches  
any row \*/

B AS (B.price < PREV(B.price)),

C AS (C.price <= PREV(C.price)),

D AS D.Price > PREV(D.price)),

E AS (E.Price >= PREV(E.Price)),

F AS (F.Price >= PREV(F.price) AND  
F.price > A.price))



*Define the pattern and the conditions which must be satisfied in each state of the pattern*

*No condition on A*

```

SELECT a_symbol,
       a_tstamp,      /* start time */
       a_price,       /* start price */
       max_c_tstamp, /* inflection time */
       last_c_price,  /* low price */
       max_f_tstamp, /* end time */
       last_c_price,  /* end price */
       Matchno

```

```

FROM Ticker MATCH_RECOGNIZE (
  PARTITION BY Symbol
  ORDER BY Tstamp
  MEASURES A.Symbol AS a_symbol,
           A.Tstamp AS a_tstamp,
           A.Price AS a_price,
           MAX (C.Tstamp) AS max_c_tstamp,
           LAST (C.Price) AS last_c_price
           MAX (F.Tstamp) AS max_f_tstamp
  MATCH_NUMBER AS matchno

```

**ONE ROW PER MATCH**

**AFTER MATCH SKIP PAST LAST ROW**

**MAXIMAL MATCH**

**PATTERN (A B C\* D E\* F+)**

**DEFINE /\* A defaults to True, matches any row \*/**

**B AS (B.price < PREV(B.price)),**

**C AS (C.price <= PREV(C.price)),**

**D AS D.Price > PREV(D.price),**

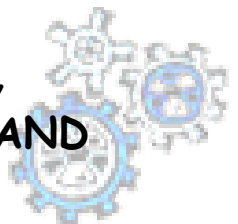
**E AS (E.Price >= PREV(E.Price)),**

**F AS (F.Price >= PREV(F.price) AND  
F.price > A.price))**

```

{ ONE ROW | ALL ROWS } PER MATCH
{ MAXIMAL | INCREMENTAL } MATCH
AFTER MATCH SKIP { TO NEXT ROW |
                  PAST LAST ROW |
                  TO LAST<variable> |
                  TO FIRST <variable>
}

```



```

SELECT a_symbol,
       a_tstamp,      /* start time */
       a_price,      /* start price */
       max_c_tstamp, /* inflection time */
       last_c_price, /* low price */
       max_f_tstamp, /* end time */
       last_c_price, /* end price */
       Matchno
FROM Ticker MATCH_RECOGNIZE (
  PARTITION BY Symbol
  ORDER BY Tstamp
  MEASURES
  A.Symbol AS a_symbol,
  A.Tstamp AS a_tstamp,
  A.Price AS a_price,
  MAX (C.Tstamp) AS max_c_tstamp,
  LAST (C.Price) AS last_c_price
  MAX (F.Tstamp) AS max_f_tstamp
  MATCH_NUMBER AS matchno

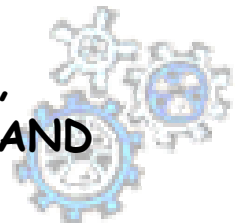
```

**Measures:  
Naming and  
renaming**

```

ONE ROW PER MATCH
AFTER MATCH SKIP PAST LAST ROW
MAXIMAL MATCH
PATTERN (A B C* D E* F+)
DEFINE /* A defaults to True, matches
any row */
B AS (B.price < PREV(B.price)),
C AS (C.price <= PREV(C.price)),
D AS D.Price > PREV(D.price)),
E AS (E.Price >= PREV(E.Price)),
F AS (F.Price >= PREV(F.price) AND
F.price > A.price))

```



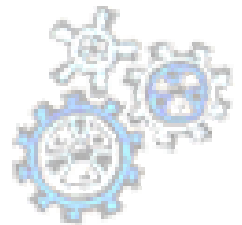
# Proposed Standards

## ⌘ More features:

- ☑ Windows
- ☑ Alternate Patterns (disjunction)
- ☑ All permutations
- ☑ Greedy vs. reluctant matching modes,
- ☑ Etc. ...

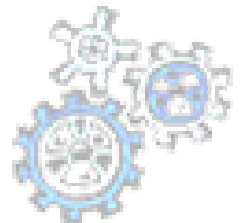
## ⌘ My First Impression:

- ☑ Very Expressive—even if we only use basic constructs
- ☑ Basic semantics easy to understand and use
- ☑ A bit overdone—raising some questions on semantics, implementation, and optimization issues.



# Composite Events for Active DB Rules

- ⌘ Interesting work and interesting semantics
- ⌘ But did not make into SQL standards ...
  - ☒ triggers often viewed as expensive and complex
  - ☒ Even MQ publish/subscribe did not get much traction
- ⌘ Query-oriented approach offers two important benefits:
  - ☒ Same query language on both stream data and stored data (i.e., same for push\* and pull\*\*)
  - ☒ Query optimization (actually optimizations since the techniques used are often different)
  - ☒ Ability to capture advanced semantics—such as Snoop's “chronicle”—a research issue.



# The End

☑ Thank you !

☑ Discussion?

