Esper JDBC

Interoperability for the Esper CEP engine
Esper JDBC

- **Server + Client building blocks**
  - Server side JDBC enablement: Esper JDBC (server behavior)
  - Client side JDBC enablement: Esper JDBC driver (single jar file, ~100KB)
  - Esper JDBC driver is « Type 4 » (full Java)

- **Java Database Connectivity (JDBC)**
  - A set of industry defined standards and API (Java Specification)
  - To enable interop between Java and external *systems of records* like a Database, but also like a CEP server

- **No proprietary vendor lockin code on the client side**
  - Drivers are loaded by name
  - The API is fully standardized
    
    ```java
    import java.sql.*;
    ```
  - Esper JDBC is JDBC 3.0 / JSR-54 / SQL-92 compliant (partial)

- **Very large set of knowledge sources available**
Esper JDBC

- Turns Esper into a JDBC compliant server
  - Get an on-demand view on the data continuously computed by Esper
  - Remotely or Locally
- Leverage well known paradigms
- Ensure interoperability with existing tools and infrastructure

<table>
<thead>
<tr>
<th>Concept</th>
<th>How</th>
<th>Esper</th>
<th>RDBMS analogy</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-demand query</td>
<td><code>select ... from ...</code></td>
<td>Named Window</td>
<td>Table</td>
</tr>
<tr>
<td>SQL-92 compliance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>On-demand computation</td>
<td><code>call ...</code></td>
<td>EPL Statement result (Iterator)</td>
<td>Stored Procedure</td>
</tr>
<tr>
<td>Namespace</td>
<td><code>catalog</code></td>
<td>Engine instance URI</td>
<td>Schema</td>
</tr>
</tbody>
</table>
Sample Esper JDBC use cases

- On-demand query for analysis & troubleshooting
- Point and click integration with any reporting tool

Esper JDBC (server endpoint)

- Event Stream connectors & adapters
- Historical data access layer
- Historical data

- Statements
- Named Windows
- Continuous processing
- Esper engine
- Event Query & Causality Pattern Language
- Core container

High-speed high-volume real-time data streams

Esper JDBC (client driver)

Output adapters

Esper: Lightweight ESP/CEP container

Copyright 2006 – 2019 EsperTech Inc.
Esper JDBC Server and Client

- Esper (server) must start an Esper JDBC server endpoint
  - IP, Port (defaults to 8450), session idle timeout, threads

```java
JDBCEndpointConfiguration config = new JDBCEndpointConfiguration();
config.setListenPort(8450);
config.setSessionIdleTimeout(60); // 60-second idle timeout
JDBCEndpoint endpoint = new JDBCEndpoint(config);
endpoint.start();
```

- Esper JDBC driver
  - JDBC Driver and URL or DataSource

```java
Class.forName("com.espertech.esper.jdbc.remote.EPLRemoteJdbcDriver");
String url = "jdbc:esper:remote:127.0.01:8450"; // hostname:port
Connection connection = DriverManager.getConnection(url);

// or using Esper JDBC Data Source
EPLRemoteDataSource remoteDataSource = new EPLRemoteDataSource();
remoteDataSource.setServer("myhostname");
remoteDataSource.setPortNumber(8450);
Connection connection = remoteDataSource.getConnection();
```
Example: Named Window over JDBC

- **Esper Named Window**
  - A continuous view on a stream
  - Esper JDBC enables SQL-92 select
    
    ```
    select … from [Named Window] where ….
    ```

    // Esper EPL (server side)
    ```
    create window TickSummary.win:time(60 sec)
    as select symbol, price from Tick
    ```

    // JDBC (client side)
    ```
    // import java.sql*
    String query = "select * from TickSummary where symbol = ‘GOOG’";
    PreparedStatement stmt = connection.prepareStatement(query);
    ResultSet rs = stmt.executeQuery();
    //... code to interrogate the result set
    … rs.next();
    … rs.getDouble("price");
    ```
Example: EPL Statement results over JDBC

- **Esper EPL Statement**
  - A continuous query on real-time streams
  - Esper JDBC enables SQL-92 stored procedure call
    call [EPL Statement name]

```java
// Esper EPL (server side)
select symbol, avg(price) as avp from Tick group by symbol
// assume statement name is avgPriceBySymbol

// JDBC (client side)
// import java.sql*
String epl = "avgPriceBySymbol";
PreparedStatement stmt = connection.prepareStatement(epl);
ResultSet rs = stmt.executeQuery();
//... code to interrogate the result set
... rs.next();
... rs.getDouble("avp");
```
A large bookstore search engine

Esper CEP used to extract continuous KPI of search count, average latency, most searched items over the last minutes etc.

Steps

Simple query over any JDBC compliant tool

Integration of KPI in end-user visual friendly reports
Example Named Windows

create window SearchRequests.win:time(1 min) as select * from SearchRequest

Example EPL Statements

select serviceName, count(*)
   from SubmittedWorkEvent.win:time(60 sec) group by serviceName
   output every 1 seconds

select serviceName, instanceId, avg(msecClockTime), count(*)
   from CompletionEvent.win:time(60 sec) group by serviceName, instanceId
   output every 1 seconds

select serviceName, avg(msecClockTime), count(*)
   from CompletionEvent.win:time(60 sec) group by serviceName
   output every 1 seconds

select queueName, avg(depth)
   from QueueReportEvent.win:time(60 sec) group by queueName
   output every 1 seconds
Thank you
http://www.espertech.com
info@espertech.com