Result Set Collection Decorator Requirements Specification

1. Scope

1.1 Overview

Quite often, reporting applications require a result set of raw data as well as a set of extra information and behavior to fully describe and use the data. This component provides a simple and robust interface to associate both related information and necessary behavior directly to a result set instance.

This component is intended entirely as added functionality around a retrieval mechanism; it is not intended to affect any update logic.

1.2 Logic Requirements

1.2.1 Compatibility

The component must be compatible with the TCS Result Set Collection.

1.2.2 Result Set Expiration

One key piece of data is result set expiration. Many result sets have a limited life-time; pricing, scores, rankings, and other such data have a limited usable lifetime. By directly associating an expiration date with the result set, cache, user presentation, and other operations become simpler to manage.

The expiration date generation process must be arbitrary. The component must support fixed time (configurable length from creation) expiration directly, but must allow for other algorithms (potentially data dependant).

1.2.3 Data Association

The component must be capable of associating arbitrary data to the result set at the set, row, and column levels.

1.2.3.1 Set Associated Data

The only explicitly required set-level data are the criteria used to generate this result set. The format of this data is up to the designer. At minimum, the criteria stored must be capable of recreating a SQL style WHERE clause. The intention of this requirement, however, is to ease presentation to the user (e.g. “These are the filters used to create this table: …”).

1.2.4 Behavior Association

The component must allow the retrieval of cells to be altered by an arbitrary ordered list of behaviors. This altered cell retrieval should be optional; raw data must still be available. The only strictly required behavior is cell-mapping, described in 1.2.4.1.

The designer is responsible for the general behavior interface.

1.2.4.1 Mapping

For this release of the component, there is one critical behavior: mapping. The component must provide an interface to allow each cell to be mapped from its “raw” value to a new value. There should not be a limitation on the source and destination types; for example, it should be possible to map an integer to a string, or a string to an integer.

Map miss behavior (source value not mapped) is up to the discretion of the designer.

As an example, the user might come up with 3 mappings from the raw value:

Message Type -> Importance Ranking
Importance Ranking -> Status Ranking
Status Ranking -> Status Name

The user should be able to create one behavior that aggregates all mappings, or a behavior for each mapping. The chain of behaviors should start with the raw value from the table, and each subsequent behavior should take the output from the previous behavior as its input.

1.3 Required Algorithms

No specific algorithms are required.

1.4 Example of the Software Usage

A simple reporting application caches result sets for 30 minutes after retrieval. These results may be output to a web page, or to an Excel document. Both formats indicate the filters used to generate the results. Further, certain columns (e.g. item type) need to be mapped from an integer to a string value.

1.5 Future Component Direction

Future direction may include more native behaviors such as formatting or filtering.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

No GUI required.

2.1.2 External Interfaces

The component does not need to literally use the Decorator design pattern. However, the full functionality of the Result Set Collection must be retained and available to consumers.

2.1.3 Environment Requirements

- Development language: Java1.4
- Compile target: Java1.4

2.1.4 Package Structure

com.topcoder.sql.resultsdecorator

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

- The fixed length expiration period must be set through configuration, as well as through the component API.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

JDBC

3.2.2 TopCoder Software Component Dependencies:

- Result Set Collection
- Configuration Manager

**Please review the TopCoder Software component catalog for existing components that can be used in the design.

3.2.3 Third Party Component, Library, or Product Dependencies:

None.

3.2.4 QA Environment:

- RedHat Linux 7.1
- Windows 2003

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines.

3.4 Required Documentation

3.4.1 Design Documentation

- Use-Case Diagram
- Class Diagram
- Sequence Diagram
- Component Specification

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the ‘Documentation’ tab of Poseidon.