Objectivity Inc.

Assist Query Builder MRD

Version 1.0

Revision History

Date	Version	Description	Author
1/27/2011	1.0	Initial	Lenny Hoffman

Table of Contents

MRD

1.Introduction

The purpose of this document is to collect, analyze and define high-level needs and features of the *Assist Query Builder*. It focuses on the capabilities needed by the stakeholders, and the target users, and why these needs exist. The details of how the Assist Query Builder fulfils these needs are detailed in Assist Query Builder SRD.

1.1.Definitions, Acronyms and Abbreviations

Query Builder:

A graphical user interface (GUI) that facilitates the development of a query expression.

1.2.References

The query builder interfaces with the operators registered with mechanisms introduced by the Enhance Object Qualification (EOQ) project:

http://devwiki.objy.com/wiki/index.php/Enhanced_Object_Qualification

1.3.Overview

Assist already has a query builder, but it is crude, it is not easy to use and most of all does not support the flexible operator support added by the EOQ project.

Despite its crudeness, the current Assist query builder gets a lot of use by users looking to verify that data expected to exist does indeed exist and looking to learn about and view their data independent of their application, especially during application development when it is not available for that purpose.

As data size and complexity have increased, query has fast become a top feature sought from Objectivity by prospects. A query builder that showcases the query capability Objectivity provides today and is flexible enough to extend to showcase new query power and flexibility as it is added will go a long way toward making it clear to the ever extending marketplace seeking query capability what we can do for them.

2.Feature Requirements

2.1.Problems

2.1.1.The current query builder does not adjust to the current set of available operators.

There only operators it knows about are the ones it is hard coded for.

2.1.2.New operators added by Objectivity have to be hard coded in.

This means that there is a lag between our adding new built in operators and their getting supported by Assist.

2.1.3.Operators added by users are not supported.

We have given users the important ability to add their own custom operators, and they will naturally want to be able to use them in Assist, but they cannot.

2.1.4.The newly added literal types are not supported.

This means many useful queries are not supportable.

2.1.5.The new variable capability is not supported.

Variable support solves ease of use and performance problems for applications, but at least they have had the workaround of using string substitution, but that is not available to users of Assist. This problem becomes exaggerated once Problem 122 is solved, as many saved queries will be in a sense a template for a general query whose values should be filled out at use time.

2.1.6.The current query builder is hard to use.

This makes it harder to visualize and edit a query.

2.1.7.The query builder does not allow for simple typing in of a query.

For simple queries, simply typing them in is much easier than having to use the builder.

2.1.8.The query builder does not support drag and drop and requires a create deal of clicking on the right spot.

It is not a graphical display, rather it is a faked out textual line that one has to click on in the right spot to ensure the insert occurs in the right spot, which has been observed to be awkward.

2.1.9.There is no way to save and manage built queries.

It can be a lot of work creating a query and not being able to save it means having to recreate it for each time it is to be used.

2.2.Goals

2.2.1.Provide support for all currently registered operators no matter the source.

This means solving Problems 117, 117.1 and 117.2.

Power

As we and customers add operators query power is being increased and provided immediate and thorough access to those operators in Assist makes that power available.

Ease

Productivity

For both ease and productivity – by automatically detecting and presenting for use all operators including user defined ones, users don't have to find and use a workaround when wanting to build a query involving one or more of them.

2.2.2.Provide support for the new literal types and variable capability.

This means solving Problems 118 and 119.

Power

These new capabilities add power to queries, as they can do more than before.

2.2.3.Make it easy to create and edit a query.

This means at minimum solving Problems 120, 120.1 and 120.2.

Ease

2.2.4. *Make it easy to comprehend even complex queries.*

Expression tree visualization in addition to the traditional string representation is one way to achieve this goal.

Ease

2.2.5. *Provide users the capability to save and retrieve queries.*

This means at minimum solving Problem 122.

Ease

3.Relationship to other features

3.1.Part of an optional feature? *****

None.

3.2.Other features required.

None.

3.3.Other features requiring this one.

None.

4.Positioning

4.1.Competitors that have this capability:

Almost all RDBMS systems come with some type of query builder, but even more significant are the number of 3rd party options available for SQL query builders.

4.2.Customers that require this capability:

All benefit.

4.3.Revenue at risk or which could be won:

The query builder is a big part of us being able to demonstrate or users demonstrating for themselves that they can execute the type of queries they need to, thus helping secure new business.

4.4.When is this required?

As soon as possible given other priorities.

5.Extent.

5.1.Languages that must support this capability:

NA.

5.2.Platforms that must support this capability:

All that Eclipse supports.

6.Applicable Standards

None.