

have the ability to create custom User-Defined Functions or StoredProcedures in the database to be used by the CQR engines and are forced to use existing stored procedures. For existing stored procedures, these may not be normally used as a unified CRUD set and shaping/naming may be different between each stored procedure. In general, existing stored procedures cannot be altered because other legacy applications will be using them. This may result in the user needing to write SQL to manipulate the results. The user may also want to use SQL to massage data when mapping significantly de-normalized tables.

**[0209]** Custom SQL. In this scenario, the user does not have a procedure on the server to implement CRUD operations, but desires to perform some type of advanced behavior, so the user is allowed to craft arbitrary SQL with some restrictions. Examples of this scenario include: custom shaping; calling scalar UDFs/UDT methods; and Inserting/Updating/Deleting data in multiple tables or tables other than the table that serves as the BasedOn for the CustomTable.

**[0210]** Custom Table Format

**[0211]** 1. CustomTables Element For consistency with other RSD structures, the CustomTables element represents a container element for zero or more CustomTable types. CustomTables is a child of the Schema element. The implication of this is that custom tables are referenced in the Mapping file exactly like physical tables.

**[0212]** 2. CustomTable Element

**[0213]** The CustomTable element is exposed to mapping as if it is a table, but under the covers it allows customization so that Insert, Delete, Update and Query commands can be overridden to come from various DB structures such as stored procedures, UDFs, or inline SQL statements as outlined below.

**[0214]** CustomTables fall into two main categories: BasedOn and Procedure Abstraction. BasedOn CustomTables are based on a physical table or view and generate one or more of their Commands automatically according to the definition of the BasedOn structure. Automatically-generated commands behave as if they were executed directly against the BasedOn structure. BasedOn CustomTables can still override individual commands. Procedural Abstraction CustomTables do not have any automatically generated commands and must have their Columns defined explicitly. The Columns serve as an abstraction for binding FieldMaps to parameters and/or Result columns.

Tag	Card	Comment
<u>Attributes</u>		
Name	Req	String representing the name of the CustomTable. Must conform to the structure naming and uniqueness rules (i.e., cannot share a 3-part name with any other structure in the RSD file). This name is referenced in a case sensitive manner to be consistent with other relational structures.

-continued

Tag	Card	Comment
<u>Sub-Elements</u> The sub-elements are constrained by the following content model:		
BasedOn	0-1	BasedOn references an existing Table or View. The BasedOn serves as the basis for the CustomTable's columns, relationships, and is used to auto-generate commands where appropriate. The relational structure that the CustomTable is BasedOn is resolved using one, two, or three part names.
Columns	0-1	Container elements for explicitly defined Columns in the Custom Table. By definition these columns are simply abstractions for procedure parameters or result columns so they only allow a name and a type to be specified.
Condition	0+	This is meant to be a filter over the set that is exposed by the CustomTable. Condition can be used independently of the QueryCommand or in conjunction with the QueryCommand. If the predicate specified in the Condition can be composed with the QueryCommand, the CustomTable can be generated on the server, however in particular cases (Inline Commands, Stored Procedures) the predicate may be applied on the client.
QueryCommand	0-N	Command instance for querying the source. Multiple QueryCommand scenario is multiple StoredProcedures or UDFs taking different parameter types but returning the same result set (e.g., sp_GetCustById, sp_GetCustByName, etc.).
InsertCommand	0-1	Command instance for inserting data into the source.
UpdateCommand	0-1	Command instance for updating data of the source.
DeleteCommand	0-1	Command instance for deleting data from the source.
CustomKey	0-1	This is the key that is used to uniquely identify relational instances. This is a logical key, and overrides any CustomKey defined by the structure that the Custom Table is BasedOn. If a CustomKey is not defined and the Custom Table is BasedOn a table with a PrimaryKey the CustomKey will automatically inherit the PrimaryKey.

**[0215]** 3. BasedOn Structure

<u>Attributes</u>		
Tag	Card	Comment
Name	Req	String representing the name of the relational structure that the CustomTable is BasedOn. BasedOn can only reference a Table or View using a 1, 2, or 3-part name. Since Tables and Views share a common namespace within a schema it is not necessary to have an additional type attribute.