**ExxonMobil Chemical Company**

**Database Design Specification**

**Version 1.10**

**March 11, 2016**

**Prepared by:**

**ILS Automation Inc.**

REVISION HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 1.0 | May 27, 2014 | PH | Initial draft |
| 1.1 | October 28, 2014 | PH | Updated recipe tables |
| 1.2 | November 10, 2014 | PH | Final check for recipe tables |
| 1.3 | January 13, 2015 | PH | Added Diagnostic Toolkit tables |
| 1.4 | February 20, 2015 | PH | Updated DtQuantOutput, removed DtConsoleSubscription, Updated unique constraints. Added unit tables |
| 1.5 | April 20, 2015 | PH | Revised core post, unit, and console tables. |
| 1.6 | May 11, 2015 | PH | Added Delivery section |
| 1.7 | June 3, 2015 | PH | Added new CheckpointTimestamp to QueueMaster table, added ER diagram for common tables, documented TkWriteLocation table. |
| 1.8 | December 13, 2015 | PH | Added DtSQCDiagnosis table. |
| 1.9 | February 23, 2016 | PH | Added SfcRecipeDataKey tables. Moved UUID and DiagramUUID from the DtDiagnosisEntry table up to the DtFinalDiagnosis table. |
| 1.10 | March 11, 2016 | PH | Added ConsoleName to the TkConsole table. Updated documentation for the common tables. There were no changes to the table design. Added grade to the LtHistory table. |

**Table Of Contents**

1 Introduction 5

2 Delivery 6

3 Common Tables and Views 7

3.1 Common Infrastructure Tables 7

3.1.1 TkPost Table 7

3.1.2 TkUnit Table 8

3.1.3 TkConsole Table 8

3.1.4 TkWriteLocation Table 9

3.1.5 QueueMaster Table 9

3.1.6 QueueDetail Table 9

3.1.7 QueueMessageStatus Table 10

3.2 Lookup Tables 11

3.2.1 LookupType Table 11

3.2.2 Lookup Table 11

3.3 Association Tables 12

3.3.1 TkAssociationType Table 12

3.3.2 TkAssociation Table 12

3.4 Unit Parameter Tables 13

3.4.1 TkUnitParameter Table 13

3.4.2 TkUnitParameterBuffer Table 13

3.5 Engineering Unit Translation Tables 14

3.5.1 Units Table 14

3.5.2 UnitAliases Table 15

3.6 Miscellaneous Tables 15

3.6.1 RoleTranslation Table 15

4 Recipe Toolkit Tables and Views 16

4.1 Core Recipe Tables 16

4.1.1 RtRecipeFamily 17

4.1.2 RtGradeMaster 17

4.1.3 RtGradeDetail 18

4.1.4 RtValueDefinition 18

4.2 Download Log Tables 21

4.2.1 RtDownloadMaster 21

4.2.2 RtDownloadDetail 21

4.3 SQC Tables 22

4.3.1 RtSQCParameter 22

4.3.2 RtSQCLimit 23

4.3.3 RtGain 23

4.3.4 RtGainGrade 23

4.3.5 RtEventParameter 23

4.3.6 RtEvent 24

4.4 Miscellaneous Tables 24

4.4.1 RtAdhocCatalog 24

4.4.2 RtAllowedFlyingSwitch 24

4.4.3 RtWatchDog 24

5 Lab Data Tables and Views 26

5.1 Core Tables 26

5.1.1 LtValue Table 26

5.1.2 LtHistory Table 27

5.1.3 LtDisplayTable Table 27

5.1.4 LtPHDValue Table 28

5.1.5 LtDCSValue Table 28

5.1.6 LtLocalValue Table 28

5.1.7 LtHDAInterface Table 29

5.2 Limit Tables 29

5.2.1 LtLimit Table 29

5.3 Selector Tables 30

5.3.1 LtSelector Table 30

5.3.2 LtSelectorDetail Table 30

5.3.3 LtSelectorType Table 30

6 Diagnostic Toolkit Tables and Views 31

6.1 Entity Relationship Diagram 31

6.2 Tables 33

6.2.1 DtApplication Table 33

6.2.2 DtFamily Table 33

6.2.3 DtFinalDiagnosis Table 34

6.2.4 DtSQCDiagnosis Table 35

6.2.5 DtQuantOutput Table 36

6.2.6 DtRecommendationDefinition Table 38

6.2.7 DtDiagnosisEntry Table 38

6.2.8 DtRecommendation Table 39

7 Sequential Control Toolkit Tables and Views 40

7.1 Entity Relationship Diagram 40

7.2 SfcControlPanelMsgs Table 40

7.3 SfcSessions Table 41

7.4 SfcRecipeDataKeyMaster Table 41

7.5 SfcRecipeDataKeyDetail Table 41

# Introduction

The new Ignition-based toolkits are tightly integrated with a database. A common design pattern is that where the previous platform defined a class definition the new platform will define a table. Instances of the class correspond to records in the table.

This document describes the design of the common database used by the toolkits at all of the sites. This is commonly referred to as the XOM database.

The sites also have the freedom to design site-specific tables that will be contained in a site specific database. An example of this is the UIR functionality which varies greatly from site to site, therefore each site will define the database requirements that meet their needs.

Finally, there is also a database dedicated solely to tag history. This database is commonly referred to as “XOMhistory”

# Delivery

This section describes the delivery process for the common database. While the schema is common for all sites, during the migration process the schema is populated with site-specific data. The delivery process will be to use standard facilities in SQL\*Server to generate a script for each site that creates the tables and populates them with site specific data. Refer to the installation manual for more details.

The process of creating a site specific script that contains both the schema and the data is:

1. Open SQL\*Server
2. Initialize tables that contain transient data by running *sweeper.sql* in the database folder.
3. Right-click on the XOM database, select ***Tasks***, then ***Generate Scripts***
4. If the Introduction page is displayed press Next
5. Select “Select specific database objects”, then select Tables and Views, press Next
6. Press Advanced.
7. Change the following settings:
   1. Script Extended Properties = False
   2. Script USE DATABASE = False
   3. Type of data to script = Schema and Data
   4. Script Indexes = True
8. Specify a filename, the convention is: createXOM*site*, i.e. createXOMvistalon
9. Press Next
10. Press Next again
11. Check file into SVN in the site migration folder.

The process of creating the script that contains just the schema is:

1. Open SQL\*Server
2. Right-click on the XOM database, select ***Tasks***, then ***Generate Scripts***
3. If the Introduction page is displayed press Next
4. Select “Select specific database objects”, then select Tables and Views, press Next
5. Press Advanced.
6. Change the following settings:
   1. Script Extended Properties = False
   2. Script USE DATABASE = False
   3. Type of data to script = Schema
   4. Script Indexes = True
7. Specify a filename: createXOM.sql
8. Press Next
9. Press Next again
10. Check file into SVN.

# Common Tables and Views

This section describes common tables and views.

## Common Infrastructure Tables

This section describes tables that are common to the several of the toolkits and utilities.



### TkPost Table

This table contains a definition of a post. A post corresponds to an operator logged into a workstation. The name of the post is the same as the username the operator will use to log in to the workstation.

| Column | Description | Datatype |
| --- | --- | --- |
| PostId | System defined id for this post. | Integer, PK |
| Post | The name of the post | Varchar(50), UK, NN |
| MessageQueueId | Id of the queue that will be used for this post. | Integer, FK |

### TkUnit Table

This table defines a unit. A unit is a somewhat vague concept, but it corresponds to something physical.

| Column | Description | Datatype |
| --- | --- | --- |
| UnitId | The name of the Ignition role, i.e., AE, Operator. | Varchar(50), PK, NN |
| UnitName | The name of the unit | Varchar(50), UK, NN |
| UnitPrefix | Prefix to be used when certain abbreviation and concatenations are used. This was relevant in the old system to help define a set of recipe tables for a unit. (I’n not sure if this is used in the new system). | Varchar(50) |
| unitAlias | An alias for the unit. (I’m not sure where this was used in the old platform and if it will be used in the new application) | Varchar(50) |

### TkConsole Table

This table lists the windows that will be opened for a post when the operator logs in to the system. The main window that is anchored to the left side is often referred to as the console, but any number of windows or consoles can be opened.

| Column | Description | Datatype |
| --- | --- | --- |
| ConsoleId | System defined id for this record. | integer, PK |
| PostId | Id of the post for this window. | Integer, FK, NN |
| WindowName | The full path to the Vision window that will be displayed for this console. | Varchar(100), UK, NN |
| ConsoleName | The name of the console as it appears in the pull-down console menu. The common XOM project is configured with all consoles for all sites. The data in this table defines which of those are relevant for an individual site. Irrelevant consoles will be deleted by the client startup script. | Varchar(100), NN, UK |
| Priority | Used for ordering the windows | Integer |

### TkWriteLocation Table

This table defines the OPC servers and their scan classes. This table should reflect what is configured in Ignition. There is not a mechanism to update Ignition from the contents of this table

| Column | Description | Datatype |
| --- | --- | --- |
| WriteLocationId | System defined id for this write location. | Int, PK |
| Alias | A common name for this write location which may correspond to an alias used in the old recipe toolkit and throughout the old application. | Varchar(max), UK, NN |
| ServerName | The name of the OPC Connection in Ignition. | Varchar(max), NN |
| ScanClass | The name of the scan class in Ignition. | Varchar(max), NN |

### QueueMaster Table

This table defines queue instances. It contains one record for each queue.

| Column | Description | Datatype |
| --- | --- | --- |
| QueueId | A system defined id | Integer, PK, NN |
| QueueKey | A unique key for this queue that will be used during insert and fetches. (A preferred name for this column would be Key but that is a reserved word) | Varchar(50), UK, NN |
| Title | The title of the queue used in the user interface. | Varchar(100), NN |
| CheckpointTimestamp | This is used to mark a certain point in time that serves as way to distinguish between historic and current messages. This is most often used by the sequential control toolkit at the beginning of an operation. | Datetime |

### QueueDetail Table

This table contains the contents of a queue. There is one record for each entry in the queue.

| Column | Description | Datatype |
| --- | --- | --- |
| Id | A system defined id | Onteger, PK, NN |
| QueueId | The id of the queue for this message. Foreign key to the QueueMaster table. | Integer, FK, NN |
| Timestamp | The timestamp, generally the system time of when the message was inserted. | Datetime, NN |
| StatusId | The id of the status, which determines the background color, of the message in the UI. Foreign key to the QueueMessageStatus table. | Integer, NN, FK |
| Message | The text body of the entry. | Varchar(2000), NN |

### QueueMessageStatus Table

This table contains the mapping from message status to colors used in the queue user interface. The status/color mapping is global and applies to all queues in the application. Changes made in this table will be applied in the application the next time the queue view is opened.

| Column | Description | Datatype |
| --- | --- | --- |
| StatusId | A system defined id | Integer, PK, NN |
| MessageStatus | The status of the message. | Varchar(15), NN |
| Color | The background color that will be used in the queue user interface. The naming convention for colors is the generally accepted color names, although it is possible RGB values can also be used. | Varchar(15), NN |

## Lookup Tables

This section describes tables that are used in the lookup table facility. There are numerous different lookups used throughout the application. Rather than define a separate table for each of the different lookup types, this generic framework is provided.



### LookupType Table

This table defines the constants necessary for the linear transformation from one unit to another.

| Column | Description | Datatype |
| --- | --- | --- |
| LookupTypeCode | A user defined code that defines the lookup type. A code is used rather than a system id to make maintaining the lookup table somewhat easier | Varchar(15), NN, PK |
| LookupTypeName | The name of the unit. | Varchar(64), UK, NN |
| LookupTypeDescription | Optional description of the units | Varchar(2000) |

### Lookup Table

This table defines lookup values.

| Column | Description | Datatype |
| --- | --- | --- |
| LookupId | System assigned unique id. | Int, PK, NN |
| LookupTypeCode | Defines the type of the lookup. | Varchar(15), UK1, NN |
| LookupName | The lookup value that will be displayed in the dropdown. | Varchar(50), UK1, NN |
| LookupDescription | An optional description of the lookup | Varchar(500) |
| Active | Specifies if the value should be included in the dropdown list. This is used rather than deleting the lookup record in order to prevent foreign key integrity constraints. | Bit, NN |

## Association Tables

This section describes tables that are used to define an association between entities. In a generic sense this replaces the relation and connection constructs in G2.



### TkAssociationType Table

This table defines the association types.

| Column | Description | Datatype |
| --- | --- | --- |
| AssociationTypeId | Unique system assigned id | Int, NN, PK |
| AssociationType | The name of the association. | Varchar(64), UK, NN |

### TkAssociation Table

This table defines the association types. Associations can be directed by paying attention to the source and sink.

| Column | Description | Datatype |
| --- | --- | --- |
| AssociationId | Unique system assigned id | Int, NN, PK |
| Source | Th |  |
| Sink |  |  |
| AssociationTypeId | The type of the association. | Varchar(64), UK, NN |

## Unit Parameter Tables

This section describes tables that are used to define and implement UnitParameter tags. A unit parameter is basically a global data entity that has built in filtering in the form of averaging the last n raw values. There is a UnitParameter UDT that defines Unit Parameter behavior and an instance of that UDT should exist for every row in this table.



### TkUnitParameter Table

This table defines the unit parameter.

| Column | Description | Datatype |
| --- | --- | --- |
| UnitParameterId | Unique system assigned id | Int, NN, PK |
| UnitParameterTagName | The name of the unit parameter UDT instance. | Varchar(150), UK, NN |

### TkUnitParameterBuffer Table

This table implements the circular buffer used to store the history of values for a unit parameter. The implementation of the circular key is managed by the change handler on the unit parameter UDT. It bumps the BufferIndex and updates the appropriate cell in the circular buffer.

| Column | Description | Datatype |
| --- | --- | --- |
| UnitParameterId | Unique system assigned id | Int, NN, PK |
| BufferIndex | A pointer to the end of the circular queue. | Int, NN, PK |
| RawValue | The value | float |

## Engineering Unit Translation Tables

This section describes tables that are used in the common engineering unit translation utility.



### Units Table

This table defines the constants necessary for the linear transformation from one engineering unit to another.

| Column | Description | Datatype |
| --- | --- | --- |
| Id | Unique system assigned id | Int, NN, PK |
| Name | The name of the unit. | Varchar(64), UK, NN |
| isBaseUnit | A base unit defines the reference that all other units of the same type are with respect to. There should always be exactly one base per type. | bit, NN |
| Type | The type of the unit, i.e, length, weight, temperature, etc. | Varchar(64), NN |
| Description | Optional description of the units | Varchar(2000) |
| M | The slope variable in the y=mx+b | Float |
| B | The y-intercept in y=mx+b | Float |

### UnitAliases Table

This table contains aliases that are used for the situation where multiple names may be used for the same engineering unit, i.e., degC, DegCelcius, DegCentigrade.

| Column | Description | Datatype |
| --- | --- | --- |
| Id | System assigned unique id. | Int, PK, NN |
| Alias | An equivalent name for a unit defined in the Units table. | Varchar(64), UK, NN |
| Name | The name of a unit in the Units table. | Varchar(64), NN |

## Miscellaneous Tables

This section defines miscellaneous tables.

### RoleTranslation Table

This table contains a translation from the Windows user roles defined by ExxonMobils IT policy and logical roles used in Ignition. The reason for this table is that there isn’t a single role for AE that would be good for all of the ExxonMobil sites. Furthermore, a G-Line AE should not have access to a Vistalon site. Unfortunately, there is not a way to use this translation in any component level security configurations.

| Column | Description | Datatype |
| --- | --- | --- |
| IgnitionRole | The name of the Ignition role, i.e., AE, Operator. | Varchar(50), PK, NN |
| WindowsRole | The name of the Windows role | Varchar(50), UK, NN |
| QueueId | Id of the queue that will be used for this console. | Integer, FK, NN |

# Recipe Toolkit Tables and Views

This section describes the tables that pertain to the Recipe Data toolkit. It DOES NOT include the recipe data database tables, but only the tables required to implement the recipe data functionality.

## Core Recipe Tables

The relationship between the core tables that make up the recipe database is shown below:



### RtRecipeFamily

This table corresponds to the class definition recipe-cell-contents in the old platform. In the Vistalon application, the records in the database correspond to the named instances RLA3, RLA3\_TWR, and VFU. Records in this table are manually inserted and configured via SQL\*Server. Because these records are very static, there are only a couple of rows, and they are not likely to change, there is not a GUI to configure them.

| Column | Description | Datatype |
| --- | --- | --- |
| RecipeFamilyId | System defined id that uniquely identifies a recipe family | Integer, PK |
| RecipeFamilyName | Identifies the recipe family. | Varchar(50), UK, NN |
| RecipeUnitPrefix | Not sure if this is used | Varchar(50), NN |
| RecipeNameAlias | Not sure if this is used | Varchar(50), NN |
| PostId | The post that can download or view this recipe. | Integer, FK, NN |
| ConfirmDownload | Specifies if the operator must confirm the recipe download. | Bit |
| CurrentRecipeGrade | The id of the currently selected recipe. This is dynamically updated by the recipe download system. | Varchar(256), NN |
| CurrentRecipeVersion | The version of the currently selected recipe. This is dynamically updated by the recipe download system. | Integer |
| Status | The status of the recipe download action. This is dynamically updated by the recipe download system. | Varchar(50) |
| Timestamp | The time that this record was last updated. This is dynamically updated by the recipe download system. | Datetime |

### RtGradeMaster

This table holds the list of grades available for processing units. Its primary key, *recipeFamilyId*,*grade*,*version* corresponds to a family-grade or family-product combination. It also contains a flag marking the active version. The application guarantees that at most only one version is active for any unit-grade combination.

| Column | Description | Datatype |
| --- | --- | --- |
| RecipeFamilyId | The recipe family | Integer, PK |
| Grade | The grade | Varchar(50), PK |
| Version | The version | Integer, PK |
| Timestamp | Time the recipe was created | Datetime, NN |
| Active | There can only be one active version for a Grade. The active version will be the one that is automatically downloaded. | Bit |

### RtGradeDetail

Rows in this table hold settings for a single recipe parameter. The primary key for the table is a combination of *RecipeFamilyId*, *Grade,* *Version* and *ValueId*

| Column | Description | Datatype |
| --- | --- | --- |
| RecipeFamilyId | The recipe family | Integer, PK |
| Grade | The grade | Varchar(50), PK |
| Version | The version | Integer, PK |
| ValueId |  | Integer, NN |
| RecommendedValue | The recommended value to be downloaded to the DCS tag or Ignition tag. Unless changed by the operator, the PEND value in the Operator spreadsheet will be the Recommend value. Note that a quantity (number) or text entry is allowed. Text values can be downloaded to DCS tags that support such features. | Varchar(max) |
| LowLimit | The lowest value allowed to be entered into the Recommend column by a user of dbManager or the PEND column by a user of the Console Operator spreadsheet. | Varchar(max) |
| HighLimit | The highest value allowed to be entered into the Recommend column by a user of dbManager or the PEND column by a user of the Console Operator spreadsheet. | Varchar(max) |

### RtValueDefinition

This table defines all of the parameters that are included in the recipe for a unit. It defines and orders recipe parameters for a unit. When retrieving parameters for a recipe, the definitions should ORDER BY *presentationOrder*.

| Column | Description | Datatype |
| --- | --- | --- |
| ValueId | This uniquely identifies a recipe database | Varchar(10), PK |
| RecipeFamilyId | Identifies the recipe family. | Integer, FK, NN |
| PresentationOrder | Defines the order in which parameter records will appear when the recipe is viewed. The order is important because this is the order of download to the DCS or Ignition tag as the recipe is instantiated. In the old system the order was manually set, in the new system this value is automatically adjusted as rows are dragged and dropped on the GUI. | Integer, NN |
| Description |  | Varchar(max) |
| StoreTag | An Ignition tag, either a memory tag or an OPC tag, intended to receive the data. Tags are created dynamically within the Recipe Toolkit, as needed. Use a tag name of the form DTA(nn) or PVC(nn) for parameters which are arrays, where nn is the number of the element. For example CAF001:DTA(13) in the DCS would be CAF001.DTA(13) in the recipe. | Varchar(max) |
| CompareTag | Either a DCS tag or Ignition tag that holds a current setting shown to an operator, whereas the StoreTag holds the ultimate target which may be reachable only after some process delay. In the case of a process delay, the StoreTag value may be a target stored to a data word and used by an application to set a process SP after some delay and the CompareTag value would be the process SP that will be eventually set equal to the store value. In many cases both the StoreTag and CompareTag are the same DCS tag or Ignition tag.  Note: If you specify a compare value but not a store value the Recipe Toolkit will ignore the compare value. If you specify a store value but not a compare value the Recipe Toolkit will set the compare value to the same tag\_parameter as the store value. | Varchar(max) |
| ChangeLevel | ChangeLevel: Defines whether the operator does or does not have access to the PEND value. (NOTE: There is no PEND value in any table) The following entries are valid for the change level.  **OC**: Operator (anyone using operator mode in Ignition) can see and change the PEND value.  **AE**: Operator can see PEND value but cannot change the value.  **EO**: OC can neither see nor change the PEND value.  **CC**: Comment line that can be used to provide a visual separation between row entries or logical grouping of rows.  No StoreTag or CompareTag is entered.  If a PEND value is entered, the value is shown appended to the end of the Description in the Ignition spreadsheet. Normally, the PEND value is left blank. | Varchar(max) |
| ModeAttribute | Certain settings in a DCS require that a controller be put into a mode attribute setting so that an application external to the DCS can download a value to the controller. If the controller requires such a setup, the tag attribute to set is filled in this field. For example, a DCS could require that a controller mode attribute be put to PROGRAM to accept values. The Recipe Toolkit should in an ideal scenario place the controller to the correct setting and return the controller to the previous setting when the value is downloaded.  Note: At this time, the operator is required to have the controller in the proper control mode (computer, cascade, etc.) so that a value such as a setpoint can be successfully downloaded. For example, if a SP is downloaded to a controller in MAN with PV tracking enabled, the SP store would fail. | Varchar(max) |
| ModeValue | Defines the value (text or number) to be downloaded to ModeAttribute. | Varchar(max) |
| WriteLocation | An alias for the system where the recipe value will be written. | Varchar(max) |

## Download Log Tables

This section describes the tables that log a recipe download from the recipe database to the DCS. Even though these tables are populated from the on-line control application, they are also visible from the DB Manager application.

### RtDownloadMaster

This table contains a record for each recipe download. This table corresponds to the class definition *recipe-cell-contents* in the old platform. In the Vistalon application, the records in the database correspond to the named instances RLA3, RLA3\_TWR, and VFU.

| Column | Description | Datatype |
| --- | --- | --- |
| MasterId | System defined id | Integer, PK |
| RecipeFamilyId | The recipe family | Integer, FK, NN |
| Grade | The grade being downloaded | Varchar(50), NN |
| Version | The version of the recipe | Integer, NN |
| DownloadStartTime | The time that the download started | Datetime, NN |
| DownloadEndTime | The time that the download finished | Datetime |
| Status | “Success” or “Failure”. Success if all tags that were scheduled to be downloaded were successfully written | Varchar(50) |
| TotalDownloads | # of tags that were scheduled to be written. | Integer |
| PassedDownloads | # of tags that were successfully written | Integer |
| FailedDownloads | # of tags that were NOT successfully written | Integer |

### RtDownloadDetail

This table stores the individual record of each tag write that is attempted including the tag, value, and the error if it failed.

| Column | Description | Datatype |
| --- | --- | --- |
| DetailId | System defined id | Integer, PK |
| MasterId | The id of the download session. This column allows NULL values in order to be able to record tag writes that occur outside the scope of a recipe download. (I’m not sure if this is necessary) | Integer, FK |
| Timestamp | Time of the download | Datetime, NN |
| Tag | The tag name | Varchar, NN |
| OutputValue | The value that was written. | float, NN |
| Success | True is successfully written, False otherwise. | bit, NN |
| StoreValue | The value of the STORE tag before the download | Float, NN |
| CompareValue | The value of the COMPARE tag before the download | Float, NN |
| RecommendedValue | The value recommended from the recipe. | Float, NN |
| Reason | The reason (free text) for downloading a PEND value that is different from the Recommend value. After the operator enters a reason, the operator can then change the PEND value. | Varchar(2000) |
| Error | The reason that the write failed, NULL if it succeeded | Varchar(2000) |

## SQC Tables

There are a collection of tables were historically contained in an SQC database. Sites will only implement the tables relevant to that site, rarely are all tables implemented.



### RtSQCParameter

Defines the SQC parameters for a unit. The primary key is *parameterId*. The list of parameters is not consistent across units, but it is for every grade for a unit.

| Column | Description | Datatype |
| --- | --- | --- |
| ParameterId | System defined id | Integer, PK |
| RecipeFamilyId | Id of the family this parameter applies to | Integer, NN |
| Parameter | The parameter name | Varchar(50), NN |

### RtSQCLimit

Upper and lower SQC limits are stored for the various parameters. by *initId* , *grade*, and *parameterId*..

| Column | Description | Datatype |
| --- | --- | --- |
| ParameterId | Id of the SQC parameter | Integer, PK |
| Grade | The grade being downloaded | Varchar(50), NN |
| Version | The version of the recipe | Integer, NN |
| UpperLimit | The upper SQC limit | Float |
| LowerLimit | The lower SQC limit | Float |

### RtGain

This table defines the gain parameters that apply to a specific unit.

| Column | Description | Datatype |
| --- | --- | --- |
| ParameterId | System defined id | Integer, PK |
| RecipeFamilyId | Id of the family this parameter applies to | Integer, NN |
| Parameter | The name of the gain parameter | Varchar(50), NN |

### RtGainGrade

This table contains the gain values for a specific grade parameter pair.

| Column | Description | Datatype |
| --- | --- | --- |
| ParameterId | System defined id | Integer, PK |
| Grade | The grade that this gain applies to | Varchar(50), NN |
| Gain | The gain value | Float |

### RtEventParameter

This table defines the event parameters that apply to a specific unit.

| Column | Description | Datatype |
| --- | --- | --- |
| ParameterId | System defined id | Integer, PK |
| RecipeFamilyId | Id of the family this parameter applies to | Integer, NN |
| Parameter | The name of the event parameter | Varchar(max), NN |

### RtEvent

This table contains the event values for a specific grade parameter pair.

| Column | Description | Datatype |
| --- | --- | --- |
| ParameterId | System defined id | Integer, PK |
| Grade | The grade being downloaded | Varchar(50), NN |
| Value | The event value | Float, NN |

## Miscellaneous Tables

This section describes the remaining tables that are generally considered part of the recipe toolkit.

### RtAdhocCatalog

This table lists the adhoc database tables that are custom for the site. Putting a record in this table with a valid table definition in the database will allow the generic table editor to work in *DBManager*.

| Column | Description | Datatype |
| --- | --- | --- |
| TableName | System defined id | Integer, PK |

### RtAllowedFlyingSwitch

This table defines the allowed flying switch transitions.

| Column | Description | Datatype |
| --- | --- | --- |
| Id | System generated key | Integer, PK |
| CurrentGrade | The current grade | Varchar(50), NN |
| NextGrade | The next grade | Varchar(50), NN |

### RtWatchDog

I have no idea how data gets into this table or what the data means.

| Column | Description | Datatype |
| --- | --- | --- |
| Observation | System defined id | Integer, PK |
| Timestamp |  | Datetime, NN |

# Lab Data Tables and Views

This section describes the tables that pertain to the Lab Data toolkit. The relationship between the Lab Data tables is shown below



## Core Tables

The core Lab Data table are discussed below.

### LtValue Table

This table defines a lab measurement.

| Column | Description | Datatype |
| --- | --- | --- |
| ValueId | System defined id | Integer, PK, NN |
| ValueName | The name of the measurement | Varchar(50), UK, NN |
| Description | Optional description of the measurement | Varchar() |
| DisplayDecimals | The number of decimals to display in the user interface for this measurement | Integer, NN |
| DisplayTableId | The id of the table that this measurement will be displayed in | Integer, FK |
| LastHistoryId | Id of the most recent value for this measurement | Integer, FK |
| UnitId | Id of the unit that is associated with this lab data. | Integer, FK, NN |

### LtHistory Table

This table contains a history of all of the laboratory measurements.

| Column | Description | Datatype |
| --- | --- | --- |
| HistoryId | System defined Id | Integer, PK, NN |
| ValueId | Id of the measurement | Integer, FK, NN |
| RawValue | The value of the measurement | Float, NN |
| SampleTime | The time that the sample was taken and sent to the lab | Datetime, NN |
| ReportTime | The time that the value was reported. | Datetime, NN |
| Grade | The grade that was running at the time the sample was taken. | Varchar(50) |

### LtDisplayTable Table

This table defines the organization of the screens that the operator uses to view lab data.

| Column | Description | Datatype |
| --- | --- | --- |
| DisplayTableId | System defined Id | Integer, PK, NN |
| PostId | Id of the post where this will be displayed |  |
| DisplayTableTitle |  |  |
| DisplayPage |  |  |
| DisplayOrder |  |  |
| DisplayFlag |  |  |
| OldTableName |  |  |

### LtPHDValue Table

For a measurement that is received from the PHD data source, this table defines the information necessary to acquire the measurement.

| Column | Description | Datatype |
| --- | --- | --- |
| PHDValueId | System defined Id | Integer, PK, NN |
| ValueId | The id of the measurement | Integer, NN, FK |
| ItemId | The item-id of tag in PHD for this measurement | Varchar(50), NN |
| InterfaceId | The id of the PHD interface | Integer, NN, FK |

### LtDCSValue Table

For a measurement that is received from a DCS data source, this table defines the information necessary to acquire the measurement.

| Column | Description | Datatype |
| --- | --- | --- |
| DCSValueId | System defined Id | Integer, PK, NN |
| ValueId | The id of the measurement | Integer, NN, FK |
| ItemId | The item-id of tag in PHD for this measurement | Varchar(50), NN |
| WriteLocationId | The id of the interface/scan class to use for reading this value | Integer, NN, FK |

*Should this table contain information about where to store the data?*

### LtLocalValue Table

This table defines measurements that will ALWAYS be entered manually. It also defines the location in PHD where the value will be written.

| Column | Description | Datatype |
| --- | --- | --- |
| LocalValueId | System defined Id | Integer, PK, NN |
| ValueId | The id of the measurement | Integer, NN, FK |
| ItemId | The item-id of tag in PHD where the value will be stored after it is manually entered. | Varchar(50), NN |
| InterfaceId | The id of the PHD interface | Integer, NN, FK |

### LtHDAInterface Table

This table contains a definition of the PHD interfaces that support OPC-HDA to acquire the lab measurements.

| Column | Description | Datatype |
| --- | --- | --- |
| InterfaceId | System defined Id | Integer, PK, NN |
| InterfaceName | Name of the HDA interface in Ignition | Varchar(50), NN |

## Limit Tables

This following tables have to do with lab data limits.

### LtLimit Table

This table contains limit information. It has columns for validity, SQC, and release limits. They all allow NULL values but it is assumed that at least one pair of limits is not null.

| Column | Description | Datatype |
| --- | --- | --- |
| LimitId |  | Integer, PK, NN |
| ValueId | Id of the corresponding measurement | Integer, NN, FK |
| LimitTypeId |  | Int, FK |
| LimitSourceId |  | Int, FK |
| UpperValidityLimit |  | Float |
| LowerValidityLimit |  | Float |
| UpperSQCLimit |  | Float |
| LowerSQCLimit |  | Float |
| UpperReleaseLimit |  | Float |
| LowerReleaseLimit |  | Float |
| Target |  | Float |
| StandardDeviation |  | Float |
| RecipeParameterName |  | Varchar(100) |
| OPCUpperItemId |  | Varchar(50) |
| OPCLowerItemId |  | Varchar(50) |
| OPCWriteLocationId |  | Int, FK |

## Selector Tables

Selector tables are used to switch the source where measurements are taken from. This generally is used to account for some change in the physical configuration of the unit. The power of selectors is that it allows the other toolkits (diagnostic and sequential control) to look at a consistent sent of lab objects independent of the physical configuration of the unit. For example, the various toolkits reference a lab data object named Mooney-Lab-Data even though the source of the data is different depending on whether the reactor configuration is single or series. The tables described below provide a mechanism that provides a single controlling factor, i.e., SingleRxGrade, SeriesRxGrade, SplitFeedGrade, etc. If the logic is multi-dimensional, such as reactor configuration and flash drum usage, then the logic is best handled in a Python script rather.

### LtSelector Table

This table contains the definition of a selector.

| Column | Description | Datatype |
| --- | --- | --- |
| SelectorId | System defined Id | Integer, PK, FK, NN |
| SelectorName | The name of the selector | Varchar(50), NN, UK |
| Description | Optional description of the selector | Varchar(1000) |

### LtSelectorDetail Table

This table contains the detail of a selector configuration.

| Column | Description | Datatype |
| --- | --- | --- |
| DetailId | System defined Id | Integer, PK, NN |
| SelectorId | The id of the selector (Series, Single, etc) | Integer, NN, FK |
| SelectorTypeId | The id of the attribute (itemId, Interface) | Integer, NN, FK |
| TargetId | The id of the lab value that will be updated | Integer, NN, FK |
| TargetTextValue | TargetTextValue | Varchar(50), NN |

### LtSelectorType Table

This table contains the type of attribute that is being configured, i.e., item-id or interface Id.

| Column | Description | Datatype |
| --- | --- | --- |
| SelectorTypeId | System defined Id | Integer, PK, NN |
| SelectorTypeName | The attribute that is being configured. | Varchar(50), NN |

# Diagnostic Toolkit Tables and Views

This section describes the tables that pertain to Diagnostic toolkit.

## Entity Relationship Diagram

The diagram shown below shows the tables used in the Diagnostic Toolkit and shows the relationship between tables.



## Tables

The tables used by the diagnostic toolkit are described below.

### DtApplication Table

This table defines the diagnostic applications present in the project. There is one record for each application. Records in this table correspond to applications in the Diagnostic Toolkit resource tree in Ignition. The data is maintained by configuring (adding, deleting, updating) applications in Ignition Designer. Although this table needs to be completely configured for the Application to be functional, the columns allow NULL values so that a new application row can be inserted as applications are created / imported.

| Column | Description | Datatype |
| --- | --- | --- |
| ApplicationId | A unique system generated id. | Integer, PK |
| ApplicationName | A unique name for the application. | Varchar(250), NN, UK |
| Description | An optional description of the application. | Varchar(2000) |
| MessageQueueId | The key to use for the message queue for any post message blocks in the application. | Integer |
| IncludeInMainMenu | Flag to indicate if this application will be displayed in some pull down menu. (I’m not sure if there is a pull-down menu in the new system) | Bit |
| GroupRampMethodId | Not sure if this is used | Integer |
| UnitId | The id of the unit that this application is associated with. The post can be found using the unit. | Integer |

### DtFamily Table

This table defines a family of problems. There is one record for each family. Records in this table correspond to families in the Diagnostic Toolkit resource tree in Ignition. The data is maintained by configuring (adding, deleting, updating) families in Ignition Designer.

| Column | Description | Datatype |
| --- | --- | --- |
| FamilyId | A unique system generated id. | Integer, PK |
| FamilyName | A unique name for the family. | Varchar(250), UK1 |
| ApplicationId | The application that the family belongs to. | Integer, FK,UK1 |
| FamilyPriority | The priority of this family, the higher the number the more important the family. | Float, NN |
| Description | Option description of the family | Varchar(2000) |

### DtFinalDiagnosis Table

This table defines a final diagnosis

| Column | Description | Datatype |
| --- | --- | --- |
| FinalDiagnosisId | A unique system generated id. | Integer, PK |
| FinalDiagnosisName | A unique name | Varchar(250), UK1 |
| FamilyId | The id of the family that this diagnosis belongs to. | Integer, FK, UK1 |
| FinalDiagnosisPriority | The priority of this final diagnosis, the higher the number the more important the final diagnosis | Float, NN |
| CalculationMethod | The name of the Python calculation method, including the full path. | Varchar(1000) |
| PostTextRecommendation | Not sure how this is used | Bit, NN |
| TextRecommendationCallback | Optional callback for getting a dynamic text recommendation. (Typically, the text of a recommendation is static) | Varchar(1000) |
| RefreshRate | Interval in seconds that the recommendations will be automatically refreshed | Int, NN |
| TextRecommendation | Text that will be used for the recommendation when the textRecommendationCallback is null. | Varchar(1000) |
| Active | Flag used by the recommendation manager to indicate that this final diagnosis is currently in the list of highest priorities and is being acted upon. | Bit, NN |
| Explanation |  | Varchar(1000) |
| TrapInsignificantRecommendations | Specifies if a very small recommendation should be displayed in the setpoint spreadsheet that is displayed to the operator. | Bit |
| LastRecommendationTime | The time that the last recommendation was made for this Final Diagnosis | Datetime |
| TimeOfMostRecentRecommendation  Implementation | The time that a recommendation was last acted upon (downloaded or not downloaded). | Datetime |
| UUID | The UUID of the final diagnosis. | Varchar(100) |
| DiagramUUID | The UUID of the diagram containing the final diagnosis. | Varchar(100) |

### DtSQCDiagnosis Table

This table defines a SQC diagnosis.

| Column | Description | Datatype |
| --- | --- | --- |
| SQCDiagnosisId | A unique system generated id. | Integer, PK |
| SQCDiagnosisName | A unique name | Varchar(250), UK1 |
| FamilyId | The id of the family that this diagnosis belongs to. | Integer, FK, UK1 |
| Status | The current status of the SQC diagnosis. Possible values are Active, Inactive, Unknown | Varchar(50), NN |
| BlockId | The id of the Block in the Block Language Toolkit. This is used as part of the SQC plotting utility where the SQC diagnosis is the entry point to the utility and then the parameters and setting from the upstream SQC observation blocks are discovered. | Varchar(50) |

### DtQuantOutput Table

This table defines a Quant Output.

| Column | Description | Datatype |
| --- | --- | --- |
| QuantOutputId | System defined primary key | Integer, PK |
| QuantOutputName | A unique name for this Quant Output | Varchar(1000), UK1, NN |
| ApplicationId | Id of the application that contains this quant output. This is used to populate the list of available QuantOutputs when specifying the Quant Outputs touched by a Final Diagnosis | Integer, FK, UK1 |
| TagPath | The full path to the OPC tag governed by this Quant Output. The tag provider must be included in the path. | Varchar(1000), NN |
| MostNegativeIncrement | Self-explanatory | Float, NN |
| MostPositiveIncrement | Self-explanatory | Float, NN |
| MinimumIncrement | Self-explanatory | Float, NN |
| SetpointHighLimit | Self-explanatory | Float, NN |
| SetpointLowLimit | Self-explanatory | Float, NN |
| FeedbackMethodId | Method to use when multiple moves are recommended for the same output. Legal choices are: Most Positive, Most Negative, Average, or Simple sum. This is not case sensitive. | Varchar(50), NN |
| IncrementalOutput | Specifies if the calculated recommendations will be treated as Absolute or Incremental changes. | Bit, NN |
| OutputLimitedStatus | Automatically set by the recommendation engine. | Varchar(50) |
| OutputLimited | Automatically set by the recommendation engine. If True then the output has been limited. | Bit |
| OutputPercent | The percent of the recommended move that will be used. The percent will be less than 100% if the output is bound or if another output is bound and vector clams are enabled. Automatically set by the recommendation engine. | Float |
| FeedbackOutput | The raw output from the recommendations. Automatically set by the recommendation engine. | Float |
| FeedbackOutputManual | A manually entered output value. Automatically set by the recommendation engine. | Float |
| FeedbackOutputConditioned | The final validated output value. Automatically set by the recommendation engine. | float |
| ManualOverride | Automatically set by the recommendation engine. | Bit |
| Active | Automatically set by the recommendation engine. | Bit |
| CurrentSetpoint | The current value of the tag at the time the recommendation was made. Automatically set by the recommendation engine. | Float |
| FinalSetpoint | If Incremental, then the current setpoint + the recommendation, if Absolute, then the recommendation. Automatically set by the recommendation engine. | Float |
| DisplayedRecommendation | Used for display purposes only, this is what is shown in the setpoint spreadsheet. Automatically set by the recommendation engine. | Float |

### DtRecommendationDefinition Table

This table defines the list of Quant Outputs touched by a final Diagnosis. The data in this table is updated when a final diagnosis is edited in Ignition Designer.

| Column | Description | Datatype |
| --- | --- | --- |
| RecommendationDefinitionId | System defined primary key | Integer, PK |
| FinalDiagnosisId | Identifies a Final Diagnosis | Integer, FK, NN |
| QuantOutputId | Identifies a Quant Output | Integer, FK, NN |

### DtDiagnosisEntry Table

This table defines a dynamic diagnosis entry. A record is inserted every time a Final Diagnosis becomes true. The contents of this table are displayed in the diagnosis queue.

| Column | Description | Datatype |
| --- | --- | --- |
| DiagnosisEntryId | System defined primary key | Integer, PK |
| FinalDiagnosisId | Id of the final diagnosis. | Integer, FK |
| Status | The status of the entry | Varchar(50), NN |
| Timestamp | Timestamp when the record was created, which is the same as when the final diagnosis became true. | Datetime, NN |
| Grade | The grade that was running at the time the final diagnosis became true. | Varchar(50), NN |
| TextRecommendation | The text describing the diagnosis. | Varchar(1000), NN |
| RecommendationStatus | The status of a recommendation. Possible values: MADE, NOT-MADE, RESCINDED | Varchar(50), NN |
| ManualMove | Entered from the recommendation map | Bit, NN |
| ManualMoveValue | Entered from the recommendation map | Float, NN |
| RecommendationMultiplier | Entered from the recommendation map | Float, NN |
| RecommendationErrorText | Error description if an error is encountered during automated processing. Not sure if this is used. | Varchar(1000) |

### DtRecommendation Table

This table defines a recommendation for a specific output in response to a diagnosis entry. Records in this table are inserted and deleted dynamically when the state of Final Diagnosis changes.

| Column | Description | Datatype |
| --- | --- | --- |
| RecommendationId | System generated primary key | Integer, PK |
| RecommendationDefinitionId | Identifies the quant output that this recommendation is for. | Integer, FK |
| DiagnosisEntryId | The Diagnosis Entry that this recommendation was made for. | Integer, FK |
| Recommendation | The active recommendation, initially the auto recommendation but will be overwritten with the manual recommendation. | Float, NN |
| AutoRecommendation | The automatically calculated recommendation. | Float, NN |
| ManualRecommendation | A manually entered recommendation, this is normally NULL until the recommendation is manually edited by the operator. | Float |
| AutoOrManual | Initially AUTO but updated to MANUAL when a manual recommendation is entered. | Varchar(50), NN |

# Sequential Control Toolkit Tables and Views

This section describes the tables that pertain to the Sequential Control toolkit.

## Entity Relationship Diagram

This table contains the messages sent to the SFC control panels.



## SfcControlPanelMsgs Table

This table contains the messages sent to the SFC control panels.

| Column | Description | Datatype |
| --- | --- | --- |
| id |  | PK, varchar(36)NN |
| chartRunId |  | Varchar(36), NN |
| Message |  | Varchar(256) NN |
| createTime |  | Datetime |
| ackRequired |  | Bit |
| ackTimedOut |  | Bit |
| ackTime |  | Datetime |

## SfcSessions Table

This table is defines the SFC charts that are currently running.

| Column | Description | Datatype |
| --- | --- | --- |
| userName |  | Integer, PK, NN |
| chartName |  | Varchar(64), NN |
| chartRunId |  | Varchar(64), NN |
| Status |  | Varchar(16), NN |
| Operation |  | Varchar(64) |
| startTime |  | Datetime |
| lastChangeTime |  | Datetime |

## SfcRecipeDataKeyMaster Table

This table defines the keys that are used for “keyed” array and matrix recipe data. At Vistalon, there is a single key named VISTALON-MONOMER-KEY. The use of recipe data keys allows the elements of various recipe data arrays and matrices to be accessed using symbolic values rather than hard-coded indexes.

| Column | Description | Datatype |
| --- | --- | --- |
| KeyId | A system defined integer. | Integer, PK, NN |
| KeyName | The user defined name of the key | Varchar(50), NN |

## SfcRecipeDataKeyDetail Table

This table defines the values for a key and maps them to a specific array index. For a specific key the indexes should be 0 based. There is nothing in the table definition that ensures that the key is zero based and uses consecutive values. The validation of the index values is left to the user interface. The table does enforce appropriate uniqueness.

| Column | Description | Datatype |
| --- | --- | --- |
| KeyId | A foreign key into the master table | Integer, FK, NN |
| KeyValue | The symbolic name used to access an element of an array. | Varchar(64), NN |
| KeyIndex | The index to use for this value to access the array or matrix. | Varchar(64), NN |