**How to Define Criteria in Data Validation Module**

* Define QC query (See [Example Multiple Criteria QC Query](#example_mult_field_qc_query) below)
  + Step 1: Define the Data Validation Criteria using spreadsheet template (ex: <https://docs.google.com/a/noaa.gov/spreadsheets/d/1SQa21eVycE1V3PJ_0PoPJ7CBijl2vifz4JIFwj_QBpQ/edit?usp=sharing>)
    - \*\*Note: All important columns used when implementing the QC criteria in the Data Validation Module are "I" through "O" and the column headers are referenced throughout this document.
    - Define the purpose of the given data validation criteria in "QC Error Name (DVM\_ERROR\_TYPES.ERR\_TYPE\_NAME)" column (e.g. what condition you are identifying in the data as problematic).
    - Define the "Validation Type (DVM\_ERROR\_TYPES.ERR\_SEVERITY\_ID)" value based on the severity type of given data validation issue (e.g. FATAL, WARN, etc.)
    - Define the "Validation Scope (DVM\_ERROR\_TYPES.DATA\_STREAM\_ID)" value based on the data stream/import method being used. There may be cases where a separate set of additional data validation criteria is needed depending on the method used to enter the data (e.g. XML import module vs. APEX RPL data entry) and this flexibility has been implemented in this foreign key field. This field is used to filter data validation criteria when executed on a given data stream. Typically each data stream will have its own Validation Scope defined for the global QC queries that will be implemented on all existing data within a given data stream (e.g. RPL, UL, FOT, Tracking, etc.).
      * Since the XML Import Module and Data Validation Module were developed together the data validation criteria for the RPL data set was split between the two Validation Scopes: RPL and XML to allow the XML data validation criteria to be excluded if the data was not imported using the XML import module.
      * Each new data stream type defined in the spreadsheet will require a corresponding record in the DVM\_DATA\_STREAMS table. Each DVM\_DATA\_STREAMS will also need to define a DATA\_STREAM\_PAR\_TABLE value that corresponds to the parent table name for the given data stream.
    - Logically group the data validation criteria into similar categories (e.g. Vessel Trips, Trip Events, Set Catch) and attempt to implement each category with as few QC Views as possible. These different groups of validation criteria will be compiled into separate QC Views.
    - Define the "QC Query Field (DVM\_ERROR\_TYPES.IND\_FIELD\_NAME)" with a unique name within a given parent QC Object. This will be the View field name that will indicate if the given data validation condition has been identified in the result set data from the given QC Object.
      * This is defined as an indicator field in the corresponding QC View object.
  + Step 2: Develop the QC Query
    - Multiple data validation criteria can be implemented in a single QC query for efficiency purposes based on the type of data validation issue that is being evaluated. (e.g. a general trip event QC query could check for both blank activity code and a blank school association code).
    - Best practice: develop foundational Oracle Views for each main database table entity in the given data stream. These foundational Views should join all reference tables to allow the given main entity to be queried easily including all reference record values (e.g. SPT\_TRIP\_EVT\_V). The QC Views should be developed directly from the foundational Oracle Views (e.g. SPT\_QC\_TRIP\_EVT\_V).
    - Business Rules:
      * The parent table always needs to be included in the QC View to provide context for the given data validation issue so it can be easily identified and resolved (e.g. identifying the set catch record without providing information about the fishing trip or trip event is not very useful to a data manager).
        + For example the foundational views are as follows: Vessel Trip (SPT\_RPL\_PTA\_HEADER\_V), Trip Event (SPT\_TRIP\_EVT\_V), and Set Catch (SPT\_CATCH\_V). When developing a QC query for the Set Catch all three Views should be used and the fields that help to identify the given catch record including its parent record values should be included in the result set.
      * The parent table's primary key must be included in the result set of the given QC Object (e.g. for RPL fishing trips the VESS\_TRIP\_ID field must be included in the result set). This field will be used to filter out which parent record is being validated on a given execution and the absence of this field will cause the module to break.
      * Each indicator field defined in the spreadsheet must be included in the result set with the corresponding column alias name (e.g. INV\_DB\_DEP\_ARR\_DTM) by the QC View that indicates the presence/absence of a given data validation issue. The QC query result of each indicator field is a calculation that evaluates to ‘Y’ when the condition is true (indicates a data validation issue) or ‘N’ when the condition is false (does not indicate a data validation issue). These indicator fields are the mechanism used by the data validation framework to identify instances of various error types in the QC View results.
        + \*\*Note: If a given indicator field defined in the spreadsheet does not have a corresponding column alias in the corresponding QC View the validation issues will not be identified correctly.
        + \*\*Note: In special cases when XML\_QC\_OBJ\_ID is NULL this is the constant name that is used to refer to the a global error type (e.g. the generic DB\_ERROR)
      * Each calculated indicator field returned in the result set that indicates an error (based on [indicator field expression example](#example_mult_field_qc_query) in yellow) should also be implemented on the WHERE clause of the query as well with OR operators between each indicator field expression (show in blue) so that the only rows that satisfy at least one of the validation criteria are returned by the query so that each individual data validation issue can be logged separately.
        + If there is only one data validation criteria that is implemented in a given QC query then the ON/WHERE clause will identify all records that satisfy the given data validation criteria and have a static formula for the given indicator field since it will be true for all returned rows (see [Example Single Criteria QC Query](#example_single_field_qc_query) below)
      * Each non-indicator field that is included in the result set should have a purpose in describing/providing context to the given validation issue (e.g. Departure Date, Vessel Name, etc.). These fields will be used in the ERR\_TYPE\_COMMENT\_TEMPLATE to generate a custom error message based on the context of the record value(s).
      * QC objects and associated Error Types can be enabled/disabled by updating the QC\_OBJ\_ACTIVE\_YN and ERR\_TYPE\_ACTIVE\_YN field values respectively. Note that for existing records that already have PTA error types defined will ignore these field values, they will only affect new records.
    - Update the spreadsheet to populate the "QC Query Error Message (SPT\_ERROR\_TYPES.ERR\_TYPE\_COMMENT\_TEMPLATE)" column with the template text for the given error type. This is the template for the specific error description that exists for the specific error condition. This field should contain placeholders in the form: [PLACEHOLDER] where PLACEHOLDER is the corresponding field name in the result set that will have its placeholder replaced by the corresponding result set field value. This value can be NULL only in special cases when XML\_QC\_OBJ\_ID is NULL.
      * All of the placeholders in the error type comment template must be included in the result set of the corresponding QC Object otherwise the missing fields will have the corresponding placeholders replaced with a blank instead of the field value
      * Example error comment template value can be found [here](#err_template). In this example the RPL\_ORIG\_OB\_FISH\_WT\_CHR field must be included in the field list of the result set for the error comment template to be generated correctly. Each data validation issue identified during the data validation process will be inserted as a separate DVM\_ERRORS record with an ERROR\_DESCRIPTION value defined as the error comment template with all placeholders replaced with the runtime values of the QC query results. This is how the context of how to identify the location of the error is communicated to data management staff.
    - Update spreadsheet to populate the "QC Query (DVM\_ERROR\_TYPES.QC\_OBJECT\_ID)" column with the corresponding QC Oracle View name developed to identify the given data validation issue type.
      * This field value will be used to assign the primary key value of the corresponding DVM\_QC\_OBJECTS record
  + Step 3: Register the QC Object into the Data Validation Module by entering records into the database
    - Execute the DDL to define the QC View Objects
    - Define View/Field comments for the QC object
      * Best practice: use Excel formulas to generate the DDL necessary to define comments on the different columns of a given View object. Many of the View fields' comments can be pulled directly from the underlying base table's field comments using the VLOOKUP function. Example can be found [here](../DB_DDL_helper.xlsx) on the "View Comments" worksheet.
    - Add the DVM\_QC\_OBJECTS record to the database
      * OBJECT\_NAME is the name of the corresponding QC View
      * QC\_SORT\_ORDER is the relative order that the QC Views will be executed in for a given data stream when the data validation criteria is evaluated
      * QC\_OBJ\_ACTIVE\_YN should be 'Y' if the given QC Object is enabled and 'N' if the object should be excluded from being evaluated on new records.
    - Add the DVM\_ERROR\_TYPES records to the database for each indicator field in the query that has been implemented
      * Use the values in the spreadsheet to populate the values of the records in the SPT\_ERROR\_TYPES records in the fields indicated in the column headers in parentheses.
      * ERR\_TYPE\_ACTIVE\_YN should be 'Y' if the given Error Type is enabled and 'N' if the Error Type should be excluded from being evaluated on new records.
      * ERR\_TYPE\_DESC can be populated with a description of the Error Type but this is optional
  + Step 4: Verify that all of the error type records' error comment template's defined field references have corresponding field values in the given QC View object.
    - This can be done using the DVM\_QC\_MSG\_MISS\_FIELDS\_V View object, all rows returned will contain a comma-delimited list of missing QC View field references in the MISSING\_VIEW\_FIELDS field
* Configure parent table in Data Validation Module
  + Create the required parent error table column in the parent table (defined in DVM\_DATA\_STREAMS.DATA\_STREAM\_PAR\_TABLE for the given data stream) by modifying the provided scripts by defining the :table\_name, :uk\_name, :fk\_name bind variables based on the data model on the target database instance.
  + \*\*Note: code and detailed information can be found here: [configure\_parent\_table.sql](../SQL/scripts/configure_parent_table.sql)

**Examples Section:**

**Example Multiple Criteria QC Query:**

CREATE OR REPLACE VIEW

SPT\_QC\_TRIP\_OB\_FISH\_V

AS SELECT

SPT\_TRIP\_OB\_FISH\_V.TRIP\_DISP\_NAME,

SPT\_TRIP\_OB\_FISH\_V.RPL\_ORIG\_OB\_FISH\_WT\_NUM,

SPT\_TRIP\_OB\_FISH\_V.RPL\_ORIG\_OB\_FISH\_WT\_CHR,

SPT\_TRIP\_OB\_FISH\_V.OB\_FISH\_WT\_MT,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_VESS\_NAME,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_REG\_NUM,

SPT\_RPL\_PTA\_HEADER\_V.PTA\_VESS\_NAME,

SPT\_RPL\_PTA\_HEADER\_V.FORMATTED\_ARRIVAL\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.FORMATTED\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_REG\_NUM,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_ID,

SPT\_RPL\_PTA\_HEADER\_V.WT\_UOM\_CONV\_FACTOR\_FROM,

SPT\_TRIP\_OB\_FISH\_V.OB\_FISH\_ID,

(RPL\_ORIG\_OB\_FISH\_WT\_NUM \* WT\_UOM\_CONV\_FACTOR\_FROM) CONV\_ORIG\_OB\_FISH\_WT\_NUM,

(CASE WHEN RPL\_ORIG\_OB\_FISH\_WT\_CHR IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM >= 0 AND RPL\_ORIG\_OB\_FISH\_WT\_NUM <> CONVERT\_STR\_TO\_NUMBER\_FN(RPL\_ORIG\_OB\_FISH\_WT\_CHR) THEN 'Y' ELSE 'N' END) MIS\_OB\_FISH\_WT,

(CASE WHEN RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NOT NULL AND OB\_FISH\_WT\_MT IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM >= 0 AND OB\_FISH\_WT\_MT >= 0 AND OB\_FISH\_WT\_MT <> ROUND(RPL\_ORIG\_OB\_FISH\_WT\_NUM \* WT\_UOM\_CONV\_FACTOR\_FROM, 5) THEN 'Y' ELSE 'N' END) MIS\_CONV\_OB\_FISH\_WT,

(CASE WHEN OB\_FISH\_WT\_MT IS NOT NULL AND OB\_FISH\_WT\_MT < 0 THEN 'Y' ELSE 'N' END) INV\_DB\_OB\_FISH\_WT\_MT,

(CASE WHEN RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM < 0 THEN 'Y' ELSE 'N' END) INV\_DB\_OB\_FISH\_WT\_NUM,

(CASE WHEN RPL\_ORIG\_OB\_FISH\_WT\_CHR IS NULL THEN 'Y' ELSE 'N' END) BLANK\_RPL\_OB\_FISH\_WT\_CHR,

(CASE WHEN RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NULL THEN 'Y' ELSE 'N' END) BLANK\_DB\_OB\_FISH\_WT\_NUM,

(CASE WHEN OB\_FISH\_WT\_MT IS NULL THEN 'Y' ELSE 'N' END) BLANK\_DB\_OB\_FISH\_WT\_MT

FROM

SPT\_TRIP\_OB\_FISH\_V

INNER JOIN SPT\_RPL\_PTA\_HEADER\_V

ON

SPT\_TRIP\_OB\_FISH\_V.VESS\_TRIP\_ID = SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_ID

WHERE

(RPL\_ORIG\_OB\_FISH\_WT\_CHR IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM >= 0 AND RPL\_ORIG\_OB\_FISH\_WT\_NUM <> CONVERT\_STR\_TO\_NUMBER\_FN(RPL\_ORIG\_OB\_FISH\_WT\_CHR))

OR (RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NOT NULL AND OB\_FISH\_WT\_MT IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM >= 0 AND OB\_FISH\_WT\_MT >= 0 AND OB\_FISH\_WT\_MT <> ROUND(RPL\_ORIG\_OB\_FISH\_WT\_NUM \* WT\_UOM\_CONV\_FACTOR\_FROM, 5))

OR (OB\_FISH\_WT\_MT IS NOT NULL AND OB\_FISH\_WT\_MT < 0)

OR (RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NOT NULL AND RPL\_ORIG\_OB\_FISH\_WT\_NUM < 0)

OR (RPL\_ORIG\_OB\_FISH\_WT\_CHR IS NULL)

OR (RPL\_ORIG\_OB\_FISH\_WT\_NUM IS NULL)

OR (OB\_FISH\_WT\_MT IS NULL)

ORDER BY SPT\_RPL\_PTA\_HEADER\_V.FORMATTED\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_ID,

SPT\_TRIP\_OB\_FISH\_V.TRIP\_DISP\_NAME

;

**Example Single Field QC Query:**

CREATE OR REPLACE VIEW SPT\_QC\_DUP\_TRIP\_CATCH\_V AS

SELECT

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_ID,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_NUM,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.FORMATTED\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_VESS\_NAME,

SPT\_RPL\_PTA\_HEADER\_V.PTA\_VESS\_NAME,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_REG\_NUM,

SPT\_RPL\_PTA\_HEADER\_V.FORMATTED\_ARRIVAL\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_REG\_NUM,

SPT\_TRIP\_EVT\_V.VESS\_TRIP\_EVT\_ID,

SPT\_TRIP\_EVT\_V.RPL\_ORIG\_EVT\_DATE,

SPT\_TRIP\_EVT\_V.RPL\_ORIG\_ACT\_CODE,

UPPER(SPT\_CATCH\_V.SPP\_FAO\_CODE) SPP\_FAO\_CODE,

UPPER(SPT\_CATCH\_V.RPL\_ORIG\_CATCH\_SPP) RPL\_ORIG\_CATCH\_SPP,

SPT\_CATCH\_V.RET\_CATCH\_YN,

SPT\_CATCH\_V.SIZE\_CLASS\_LABEL,

COUNT(\*) NUM\_DUP\_SPP\_RET\_DISC,

'Y' INV\_DB\_DUP\_SET\_CATCH\_SPP,

FORMATTED\_TRIP\_EVT\_START\_DTM,

FORMATTED\_TRIP\_EVT\_END\_DTM,

ACT\_CODE,

ACT\_NAME

FROM SPT\_RPL\_PTA\_HEADER\_V

INNER JOIN SPT\_TRIP\_EVT\_V

ON SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_ID = SPT\_TRIP\_EVT\_V.VESS\_TRIP\_ID

INNER JOIN SPT\_CATCH\_V

ON SPT\_CATCH\_V.CATCH\_TRIP\_EVT\_ID = VESS\_TRIP\_EVT\_ID

INNER JOIN (SELECT UPPER(SPP\_FAO\_CODE) SPP\_FAO\_CODE, CATCH\_TRIP\_EVT\_ID, RET\_CATCH\_YN, CATCH\_SIZE\_CLASS\_ID FROM SPT\_CATCH\_V WHERE CATCH\_WT\_MT > 0 group by CATCH\_SIZE\_CLASS\_ID, UPPER(SPP\_FAO\_CODE), CATCH\_TRIP\_EVT\_ID, RET\_CATCH\_YN having count(\*) > 1) DUP\_EVT\_CATCH\_SPP

ON

DUP\_EVT\_CATCH\_SPP.SPP\_FAO\_CODE = UPPER(SPT\_CATCH\_V.SPP\_FAO\_CODE) AND DUP\_EVT\_CATCH\_SPP.CATCH\_TRIP\_EVT\_ID = SPT\_CATCH\_V.CATCH\_TRIP\_EVT\_ID AND DUP\_EVT\_CATCH\_SPP.RET\_CATCH\_YN = SPT\_CATCH\_V.RET\_CATCH\_YN

AND (DUP\_EVT\_CATCH\_SPP.CATCH\_SIZE\_CLASS\_ID = SPT\_CATCH\_V.CATCH\_SIZE\_CLASS\_ID OR (DUP\_EVT\_CATCH\_SPP.CATCH\_SIZE\_CLASS\_ID IS NULL AND SPT\_CATCH\_V.CATCH\_SIZE\_CLASS\_ID IS NULL))

GROUP BY

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_ID,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_NUM,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_TRIP\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.FORMATTED\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_DEPART\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_VESS\_NAME,

SPT\_RPL\_PTA\_HEADER\_V.PTA\_VESS\_NAME,

SPT\_RPL\_PTA\_HEADER\_V.VESS\_REG\_NUM,

SPT\_RPL\_PTA\_HEADER\_V.FORMATTED\_ARRIVAL\_DTM,

SPT\_RPL\_PTA\_HEADER\_V.RPL\_ORIG\_REG\_NUM,

SPT\_TRIP\_EVT\_V.VESS\_TRIP\_EVT\_ID,

SPT\_TRIP\_EVT\_V.RPL\_ORIG\_EVT\_DATE,

SPT\_TRIP\_EVT\_V.RPL\_ORIG\_ACT\_CODE,

UPPER(SPT\_CATCH\_V.SPP\_FAO\_CODE),

SPT\_CATCH\_V.RET\_CATCH\_YN,

UPPER(RPL\_ORIG\_CATCH\_SPP),

SPT\_CATCH\_V.SIZE\_CLASS\_LABEL,

FORMATTED\_TRIP\_EVT\_START\_DTM,

FORMATTED\_TRIP\_EVT\_END\_DTM,

ACT\_CODE,

ACT\_NAME

;

**ERR\_TYPE\_COMMENT\_TEMPLATE example:**

The Fish Onboard Weight for "[TRIP\_DISP\_NAME]" reported on the RPL ([RPL\_ORIG\_OB\_FISH\_WT\_CHR]) is different than the unconverted fish onboard weight stored in the database ([RPL\_ORIG\_OB\_FISH\_WT\_NUM]) for the Vessel (RPL: [RPL\_ORIG\_VESS\_NAME], Reg Num: [RPL\_ORIG\_REG\_NUM]) (History: [PTA\_VESS\_NAME], Reg Num: [VESS\_REG\_NUM]) trip (VESS\_TRIP\_ID: [VESS\_TRIP\_ID]) that departed on [FORMATTED\_DEPART\_DTM] and arrived [FORMATTED\_ARRIVAL\_DTM]