# Centralized Cruise Database DVM Testing Documentation

## Overview:

The Centralized Cruise Database (CCD) was developed to manage cruise information for PIFSC. The Data Validation Module (DVM) was implemented to perform automated data Quality Control (QC) on the CCD to help ensure the quality of the data. There are two different methods for test case verification: SQL and PL/SQL. The standard method for defining formal, repeatable test cases for the DVM and verifying them are defined in this document. All SQL queries should be executed on the CEN\_CRUISE schema. This SOP requires the process to be completed using Oracle SQL Developer.

\*\*Note: The automated test cases require these scripts to be executed on a development/test instance. DVM rules and data will be purged from the database, to avoid data loss do not execute this on a production database.

## Resources:

* [CCD Documentation](../../Centralized%20Cruise%20Database%20-%20Technical%20Documentation.docx)
* [PL/SQL Coding Conventions](../../Centralized%20Cruise%20Database%20-%20PLSQL%20Coding%20Conventions.docx)
* [CCD DVM Test Cases](Centralized%20Cruise%20Database%20CCD%20DVM%20Test%20Cases.xlsx)

## Definitions:

* Test Case Definitions: This [excel file](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\Centralized%20Cruise%20Database%20CCD%20DVM%20Test%20Cases.xlsx) is used to define all formal test cases
* (SQL verification only) Verification Templates: These excel files (<verification_templates>) are used to list the individual test cases in a given category that are defined in the Test Case Definitions. The templates use excel formulas to verify that the results of a given script execution match the verified results.
* Verification Exports: These files ([verification\_templates\automated](verification_templates/automated)) are used to verify the results of a given test script match the verified results using a file comparison tool to streamline the process.

## Test Case Verification SOP:

* Setup Test Cases:
  + Purge CCD and DVM records from the database
    - Execute [delete\_all\_DVM\_recs.sql](../../../SQL/queries/delete_all_DVM_recs.sql)
    - Execute [delete\_ref\_data.sql](../../../SQL/queries/delete_ref_data.sql)
  + Load test data
    - Execute the corresponding test CCD data and DVM rule loading scripts
* Automated Verification:
  + (SQL verification only) Export the data from the database after the DVM has been executed on the test data and compare it to the Verification Exports.
    - SOP:
      * Execute the corresponding DVM test script(s)
      * Generate the data reports (execute the [validation issue verification query](#validation_issue_verification_query) for the given test case category and export the results in a .csv file with the specified naming convention)
      * Open a diff tool (e.g. WinMerge) and compare the exported query results (e.g. category\_1\_DVM\_issue\_verification20200423.csv for a report generated on 4/23/2020) with the corresponding Verification Export (e.g. [category\_1\_DVM\_issue\_verification.csv](verification_templates/automated/category_1_DVM_issue_verification.csv)) in the [verification\_templates\automated](verification_templates/automated) folder
        + If the two files' content matches exactly then the test cases have been verified successfully
  + (PL/SQL verification only)
    - SOP:
      * Execute the corresponding Test Cases Setup scripts
      * Clear the "script output" window
      * Execute the corresponding DVM test script
      * Copy the content in "script output" and save as a temporary text file (e.g. category\_4\_script\_output\_20200716.txt)
      * Open a diff tool (e.g. WinMerge) and compare the saved script output with the corresponding Verification Export (e.g. [category\_4\_script\_output\_verification.txt](verification_templates/automated/category_4_script_output_verification.txt)) in the [verification\_templates\automated](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\verification_templates\automated) folder
        + If the two files' content matches exactly then the test cases have been verified successfully

## Test Case Definition SOP:

* Update the Test Case Definitions in the [CCD DVM Test Cases](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\Centralized%20Cruise%20Database%20CCD%20DVM%20Test%20Cases.xlsx) workbook to add the expected results for the new test cases in the corresponding section based on the type of test case
* SQL Verification:
  + Update the corresponding <verification_templates> file(s) to add the expected result for the new test cases
    - Description: These Verification Templates translate the individual test cases in a given category defined in the Test Case Definitions into their corresponding query results so they can be compared with the query results from subsequent script executions. These template files contain excel formulas to compare the expected verified results with the actual results of a given script execution.
    - SOP:
      * Update the corresponding test data loading script (e.g. [category\_1\_load\_test\_data.sql](SQL/category_1_load_test_data.sql)) and DVM rule script (e.g. [category\_1\_load\_DVM\_rules.sql](SQL/category_1_load_DVM_rules.sql)) to load database records necessary to setup the test case conditions that can be used to verify that the expected outcome was produced.
        + [Cruise\_Leg\_DDL\_DML\_generator](../../Cruise_Leg_DDL_DML_generator.xlsx) can be used to generate the DML to load the DVM test data records in to the corresponding data loading script
        + \*\*Note: do not include any database fields in the verification queries that have a random element like primary key values or date/time values that depend on when the script was executed otherwise the automated test case verification approach will not work properly.
      * Update the corresponding DVM test script(s) (e.g. [category\_3\_exec\_DVM.sql](SQL/category_3_exec_DVM.sql)) to execute the new test cases and update the database accordingly.
      * (For new test case categories only) Define a naming convention for the Verification Template and Verification Export
      * Execute the SQL query for the given test case category and export the results in a .csv file with the specified naming convention for the Verification Export
      * Copy the exported data from the .csv file into the "Database Export" worksheet of the corresponding verification template.
      * Open the "verification" worksheet and search for the "false" value specifying "values" in the "Look in" option, update the excel formulas as necessary. Confirm there are no matches found, if so then the test cases have been successfully verified
  + Replace the corresponding Verification Export .csv file in the [verification\_templates\automated](verification_templates/automated) folder and include it in the version control commit.
* PL/SQL Verification:
  + Update the corresponding script [Verification Export](verification_templates/automated) file to add the expected result for the new test cases
    - Description: These Verification Export files contain the script output for the PL/SQL verification test cases.
    - SOP:
      * Update the corresponding test data loading script (e.g. [category\_1\_load\_test\_data.sql](SQL/category_1_load_test_data.sql)) and DVM rule script (e.g. [category\_1\_load\_DVM\_rules.sql](SQL/category_1_load_DVM_rules.sql)) to load database records necessary to setup the test case conditions that can be used to verify that the expected outcome was produced.
        + [Cruise\_Leg\_DDL\_DML\_generator](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\Cruise_Leg_DDL_DML_generator.xlsx) can be used to generate the DML to load the DVM test data records in to the corresponding data loading script
        + \*\*Note: do not include any database fields in the verification queries that has a random element like primary key values or date/time values that depend on when the script was executed otherwise the automated test case verification approach will not work properly.
      * Update the corresponding DVM test script(s) (e.g. [category\_4\_exec\_DVM.sql](SQL/category_4_exec_DVM.sql)) to include the PL/SQL code to execute the new test cases and produce the desired script output
      * (For new test case categories only) Define a naming convention for the Verification Export
      * Copy the "script output" produced by the DVM test script and save it as a text file with the specified naming convention for the Verification Export
        + Use a diff tool to verify that the script output matches the expected results defined in the corresponding script [verification export](verification_templates/automated) file
  + Include the new/modified Verification Export file in the version control commit to replace the previous version (if any).
* Update documentation (if necessary) and commit changes to the version control system.

## Test Case Types:

* DVM Tests
  + Description: The [test cases workbook](Centralized%20Cruise%20Database%20CCD%20DVM%20Test%20Cases.xlsx) lists instances of each validation criteria test case for each test case category that is implemented in the data validation module (DVM) for the CCD.
    - The columns in each test case category worksheet are based on the information that is necessary to verify the test cases
  + Category 1 Cases (SQL verification):
    - Description: These test cases verify that each type of DVM validation issue is identified successfully
    - To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute the individual scripts listed below for this test case category: [category\_1\_exec\_all\_scripts.sql](SQL/category_1_exec_all_scripts.sql)
      * Load test data:
        + Test data: [category\_1\_load\_test\_data.sql](SQL/category_1_load_test_data.sql)
        + DVM rules: [category\_1\_load\_DVM\_rules.sql](SQL/category_1_load_DVM_rules.sql)
      * DVM test script: [batch\_DVM\_script.sql](../../../SQL/queries/batch_DVM_script.sql)
    - Validation Issues:
      * Execute the [Validation Issue Verification Query](#validation_issue_verification_query)
      * Verification Files:
        + Template: [category\_1\_DVM\_issue\_verification.xlsx](verification_templates/category_1_DVM_issue_verification.xlsx)
        + Export: [category\_1\_DVM\_issue\_verification.csv](verification_templates/automated/category_1_DVM_issue_verification.csv)
  + Category 2 Cases (SQL verification):
    - Description: These test cases verify that existing validation issues are removed when the underlying validation issues are resolved by updating the record values
    - To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute all individual scripts listed below for this test case category: [category\_2\_exec\_all\_scripts.sql](SQL/category_2_exec_all_scripts.sql)
      * Load test data:
        + Test data: [category\_1\_load\_test\_data.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_1_load_test_data.sql)
        + DVM rules: [category\_1\_load\_DVM\_rules.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_1_load_DVM_rules.sql)
      * DVM test scripts:
        + Execute the DVM: [batch\_DVM\_script.sql](../../../SQL/queries/batch_DVM_script.sql)
        + Resolve the data issues: [category\_2\_data\_updates.sql](SQL/category_2_data_updates.sql)
        + Re-execute the DVM: [batch\_DVM\_script.sql](../../../SQL/queries/batch_DVM_script.sql)
    - Validation Issues:
      * Execute the [Validation Issue Verification Query](#validation_issue_verification_query)
        + Template: [category\_2\_DVM\_issue\_verification.xlsx](verification_templates/category_2_DVM_issue_verification.xlsx)
        + Export: [category\_2\_DVM\_issue\_verification.csv](verification_templates/automated/category_2_DVM_issue_verification.csv)
  + Category 3 Cases (SQL verification):
    - Description: These test cases verify that multiple data streams that share one or more QC objects can be validated separately or concurrently
    - To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute all individual scripts listed below for this test case category: [category\_3\_exec\_all\_scripts.sql](SQL/category_3_exec_all_scripts.sql)
      * Load test data:
        + Test data: [category\_1\_load\_test\_data.sql](SQL/category_1_load_test_data.sql)
        + DVM rules: [category\_3\_load\_DVM\_rules.sql](SQL/category_3_load_DVM_rules.sql)
      * DVM test script: [category\_3\_exec\_DVM.sql](SQL/category_3_exec_DVM.sql)
    - Validation Issues:
      * Execute the [Validation Issue Verification Query](#validation_issue_verification_query)
      * Verification Files:
        + Template: [category\_3\_DVM\_issue\_verification.xlsx](verification_templates/category_3_DVM_issue_verification.xlsx)
        + Export: [category\_3\_DVM\_issue\_verification.csv](verification_templates/automated/category_3_DVM_issue_verification.csv)
  + Category 4 Cases (PL/SQL verification)
    - These test cases verify the different error conditions that can be feasibly tested are handled correctly
      * A list of DVM error codes that are tested in this category for the DVM\_PKG package can be found in the [DVM repository](mailto:git@gitlab.pifsc.gov:centralized-data-tools/data-validation-module.git) in the [Business Rules List](https://gitlab.pifsc.gov/centralized-data-tools/data-validation-module/-/blob/master/docs/DVM%20-%20Business%20Rules.xlsx) where the "Scope" column values are "DVM Processing Errors"
    - Test cases setup
      * To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute all individual scripts listed below to setup the test cases for this test case category: [category\_4\_exec\_all\_scripts.sql](SQL/category_4_exec_all_scripts.sql)
        + Load test data:

Test data: [category\_1\_load\_test\_data.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_1_load_test_data.sql)

DVM rules: [category\_1\_load\_DVM\_rules.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_1_load_DVM_rules.sql)

* + - * + Execute the DVM: [batch\_DVM\_script.sql](../../../SQL/queries/batch_DVM_script.sql)
    - DVM test script: [category\_4\_exec\_DVM.sql](SQL/category_4_exec_DVM.sql)
    - Validation Issues:
      * Verification Export: [category\_4\_script\_output\_verification.txt](verification_templates/automated/category_4_script_output_verification.txt)
  + Category 5 Cases (SQL verification)
    - Description: These test cases verify custom cruise database business logic that any cruise that has an associated cruise leg that overlaps with another cruise leg in the same cruise or for the same vessel is validated the DVM automatically validates the overlapping cruise legs in the insert, update, and delete cruise leg use cases.
      * CCD\_DVM\_PKG.EXEC\_DVM\_CRUISE\_OVERLAP\_SP procedure is used when a new cruise leg is inserted (for more information see CR-DB-012 in the [business rules](../../Centralized%20Cruise%20Database%20-%20Business%20Rule%20List.xlsx))
      * CCD\_CRUISE\_PKG.DELETE\_LEG\_OVERLAP\_SP procedure is used when an existing cruise leg is deleted (for more information see CR-DB-014 in the [business rules](../../Centralized%20Cruise%20Database%20-%20Business%20Rule%20List.xlsx))
      * There is no feasible procedure that can handle the update use case so custom PL/SQL must be used to determine the overlaps before and after the update is made and re-evaluate the DVM on those cruises (for more information see CR-DB-013 in the [business rules](../../Centralized%20Cruise%20Database%20-%20Business%20Rule%20List.xlsx))
    - To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute the individual scripts listed below for this test case category: [category\_5\_exec\_all\_scripts.sql](SQL/category_1_exec_all_scripts.sql)
      * Load test data:
        + Test data: [category\_5\_load\_test\_data.sql](SQL/category_5_load_test_data.sql)
        + DVM rules: [category\_1\_load\_DVM\_rules.sql](SQL/category_1_load_DVM_rules.sql)
      * DVM test script: [category\_5\_exec\_DVM.sql](SQL/category_5_exec_DVM.sql)
    - Validation Issues:
      * Execute the [Validation Issue Verification Query](#validation_issue_verification_query)
      * Verification Files:
        + Template: [category\_5\_DVM\_issue\_verification.xlsx](verification_templates/category_5_DVM_issue_verification.xlsx)
        + Export: [category\_5\_DVM\_issue\_verification.csv](verification_templates/automated/category_5_DVM_issue_verification.csv)
  + Category 6 Cases (SQL verification)
    - These test cases verify that when validation rules change over time the appropriate validation rule sets are defined/deactivated, associated with the corresponding cruise parent records, and the active validation rules at a given time are processed with the expected results
    - To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute all individual scripts listed below for this test case category: [category\_6\_exec\_all\_scripts.sql](SQL/category_6_exec_all_scripts.sql)
      * Load test data:
        + Test data: [category\_1\_load\_test\_data.sql](SQL/category_1_load_test_data.sql)
        + DVM rules: [category\_3\_load\_DVM\_rules.sql](SQL/category_3_load_DVM_rules.sql)
      * DVM test script: [category\_6\_exec\_DVM.sql](SQL/category_6_exec_DVM.sql)
    - Validation Issues:
      * Execute the [Validation Issue Verification Query](#validation_issue_verification_query)
      * Verification Files:
        + Template: [category\_6\_DVM\_issue\_verification.xlsx](verification_templates/category_6_DVM_issue_verification.xlsx)
        + Export: [category\_6\_DVM\_issue\_verification.csv](verification_templates/automated/category_6_DVM_issue_verification.csv)
    - Validation Rules:
      * Execute the [Validation Rule Verification Query](#validation_rule_verification_query)
      * Verification Files:
        + Template: [category\_6\_DVM\_rule\_verification.xlsx](verification_templates/category_6_DVM_rule_verification.xlsx)
        + Export: [category\_6\_DVM\_rule\_verification.csv](verification_templates/automated/category_6_DVM_rule_verification.csv)
    - PTA Validation Rules:
      * Execute the [PTA Validation Rule Verification Query](#PTA_validation_rule_verification_query)
      * Verification Files:
        + Template: [category\_6\_DVM\_PTA\_rule\_verification.xlsx](verification_templates/category_6_DVM_PTA_rule_verification.xlsx)
        + Export: [category\_6\_DVM\_PTA\_rule\_verification.csv](verification_templates/automated/category_6_DVM_PTA_rule_verification.csv)
  + Category 7 Cases (SQL verification)
    - These test cases verify the invalid DVM configuration QC checks that can be feasibly tested are identified correctly
      * A list of DVM configuration criteria that are tested in this test case category for the DVM\_PKG package can be found in the [DVM repository](mailto:git@gitlab.pifsc.gov:centralized-data-tools/data-validation-module.git) in the [Business Rules List](https://gitlab.pifsc.gov/centralized-data-tools/data-validation-module/-/blob/master/docs/DVM%20-%20Business%20Rules.xlsx) where the "Scope" column values are "DVM Configuration QC"
      * \*\*Note: after processing this test case category all DVM rules and DVM data will be purged
      * \*\*Note: All DVM test scripts for this test case category must be executed or the data model will have been changed by the first DVM test script
    - Test cases setup
      * To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute all individual scripts listed below to setup the test cases for this test case category: [category\_7\_exec\_all\_scripts.sql](SQL/category_7_exec_all_scripts.sql)
        + Load test data:

Test data: [category\_1\_load\_test\_data.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_1_load_test_data.sql)

DVM rules: [category\_3\_load\_DVM\_rules.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_3_load_DVM_rules.sql)

* + - DVM test script: [category\_7\_exec\_DVM\_1.sql](SQL/category_7_exec_DVM_1.sql)
    - Validation Issues:
      * Execute the [DVM Configuration QC Verification Query](#DVM_Configuration_QC_Verification)
      * Verification Files:
        + Template: [category\_7\_DVM\_config\_error\_verification\_1.xlsx](verification_templates/category_7_DVM_config_error_verification_1.xlsx)
        + Export: [category\_7\_DVM\_config\_error\_verification\_1.csv](verification_templates/automated/category_7_DVM_config_error_verification_1.csv)
    - DVM test script: [category\_7\_exec\_DVM\_2.sql](SQL/category_7_exec_DVM_2.sql)
    - Validation Issues:
      * Execute the [DVM View Configuration QC Verification Query](#DVM_View_Configuration_QC)
      * Verification Files:
        + Template: [category\_7\_DVM\_config\_error\_verification\_2.xlsx](verification_templates/category_7_DVM_config_error_verification_2.xlsx)
        + Export: [category\_7\_DVM\_config\_error\_verification\_2.csv](verification_templates/automated/category_7_DVM_config_error_verification_2.csv)
    - Verify the view does not exist exception
      * Execute the [DVM Configuration QC Verification Query](#DVM_Configuration_QC_Verification)
      * Verify the ORA-20220 error is reported and indicates that CCD\_QC\_CRUISE\_TEMP\_V does not exist
    - Revert DVM/CCD data model changes and delete all DVM rules and data: [category\_7\_exec\_DVM\_3.sql](SQL/category_7_exec_DVM_3.sql)
    - Validation Issues:
      * Execute the [DVM Configuration QC Verification Query](#DVM_Configuration_QC_Verification)
      * Verification Files:
        + Template: [category\_7\_DVM\_config\_error\_verification\_3.xlsx](verification_templates/category_7_DVM_config_error_verification_3.xlsx)
        + Export: [category\_7\_DVM\_config\_error\_verification\_3.csv](verification_templates/automated/category_7_DVM_config_error_verification_3.csv)
  + Category 8 Cases (PL/SQL verification)
    - These custom Cruise DVM procedure test cases verify the different error conditions that can be feasibly tested are handled correctly
      * A list of DVM error codes that are tested in this category for the CCD\_DVM\_PKG package can be found in the CCD [Business Rules List](../../Centralized%20Cruise%20Database%20-%20Business%20Rule%20List.xlsx) where the "Scope" column values are "CCD Custom DVM Errors"
    - Test cases setup
      * To streamline the test case verification process a single script was compiled to purge the CCD and DVM data as well as execute all individual scripts listed below to setup the test cases for this test case category: [category\_8\_exec\_all\_scripts.sql](SQL/category_8_exec_all_scripts.sql)
        + Load test data:

Test data: [category\_1\_load\_test\_data.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_1_load_test_data.sql)

DVM rules: [category\_1\_load\_DVM\_rules.sql](file:///C:\Users\Jesse.Abdul\Documents\Version%20Control\Git\centralized-cruise-database\docs\test%20cases\DVM_PKG\SQL\category_1_load_DVM_rules.sql)

* + - DVM test script: [category\_8\_exec\_DVM.sql](SQL/category_8_exec_DVM.sql)
    - Validation Issues:
      * Verification Export: [category\_8\_script\_output\_verification.txt](verification_templates/automated/category_8_script_output_verification.txt)

## Appendix:

* Validation Issue Verification Query:

select cruise\_name, LEG\_NAME\_CD\_LIST, iss\_severity\_code, iss\_type\_name, iss\_type\_desc, ISS\_DESC, IND\_FIELD\_NAME from CCD\_CRUISE\_SUMM\_ISS\_V order by cruise\_name, iss\_severity\_code, iss\_type\_name, TO\_CHAR(iss\_desc);

* Validation Rule Verification Query:

select rule\_set\_active\_yn, rule\_data\_stream\_code, iss\_type\_name, ind\_field\_name, iss\_severity\_code, iss\_type\_desc FROM dvm\_rule\_sets\_v order by data\_stream\_code, rule\_set\_id, ind\_field\_name;

* PTA Validation Rule Verification Query:

Select cruise\_name, leg\_name\_cd\_list, rule\_data\_stream\_code, iss\_type\_name, ind\_field\_name, iss\_severity\_code, iss\_type\_desc from CCD\_CRUISE\_DVM\_RULES\_V order by cruise\_name, data\_stream\_code, ind\_field\_name;

* DVM Configuration QC Verification Query:

Select ERR\_SOURCE, ERR\_MSG FROM DVM\_STD\_QC\_ALL\_RPT\_V ORDER BY ERR\_SOURCE, ERR\_MSG;

* DVM View Configuration QC Verification Query:

select ERR\_SOURCE, ERR\_MSG from DVM\_STD\_QC\_VIEW\_V ORDER BY ERR\_SOURCE, ERR\_MSG;