



# New York State Department of Health

## Test Plan Cover

Track Name  
Release #  
Sprint #

Plan Management
1
4

Approved By	Date	Signature/Initials	Sign Off
Larry Toole	8/27/2012	L.T.	(Email attached in SharePoint > Plan Management > Testing > Approval Sheet)

Resource Name	% Committed
Nilesh Patel	100%
Pragati Shrestha	100%

Assumptions	Clarifications/Risks	Dependencies	Comments	Out of Scope

Test Plan Version #	Date	Reviewed by	Comments/Updates
0.1	8/22/2012	Parampreet Sidana	Initial Draft
0.2	8/24/2012	Larry Toole	
0.3	8/27/2012		Incorporated the changes as suggested by Larry Toole. - Defined the prototypes in user stories - Added missing steps for PM_R1_SP3_US37.1.1_TC002 - Added missing pre-conditions for test cases for user story 37.1.1



# New York State Department of Health

## Test Plan

### Overview

The test plan is designed as a baseline to guide the test team in identifying the scope to be tested, what is to be included in the testing effort, what tools will be used, the break down of the test cases and identifying risk areas, assumptions and dependencies.

Epic	User Story Name	Comments
6907	174: As the Plan Management Sprint team, I want to create the data model so that I can accurately store plan information in the Exchange.(Technical Story)	

User Story Name	Type of Test	Test Scenario	Test Case Number &Name	Test Complexity	To be included in Regression?	Manual or Automated	Testing Phase
174: As the Plan Management Sprint team, I want to create the data model so that I can accurately store plan information in the Exchange.(Technical Story)	Data Validation	ISSUER_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC001_Verify that the ISSUER_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	PLAN_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC002_Verify that the PLAN_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	PLAN_SERVICE_AREA_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC003_Verify that the PLAN_SERVICE_AREA_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	PLAN_ACCREDITATION_DTL table with valid data elements and records	PM_R1_SP3_US174_TC004_Verify that the PLAN_ACCREDITATION_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	PLAN_PREMIUM_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC005_Verify that the PLAN_PREMIUM_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	FORMULARY_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC006_Verify that the FORMULARY_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	PLAN_BENEFIT_SERVICES_REF Table with valid data elements and records	PM_R1_SP3_US174_TC007_Verify that the PLAN_BENEFIT_SERVICES_REF table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	PLAN_BENEFIT_SERVICES_COST_SHARING_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC008_Verify that the PLAN_BENEFIT_SERVICES_COST_SHARING_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	SBC_SCENARIO_RESULTS_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC009_Verify that the SBC_SCENARIO_RESULTS_DTL table exists in the database	2: Medium Complexity	No	Manual	Functional Testing
	Data Validation	SERVICE_AREA_REF Table with valid data elements and records	PM_R1_SP3_US174_TC010_Verify that the SERVICE_AREA_REF table exists in the database	2: Medium Complexity	No	Manual	Functional Testing

	Data Validation	ZIP_SERVICE_AREA_REF Table with valid data elements and records	PM_R1_SP3_US174_TC011_Verify that the Zip_Service_Area table exists in the database	2: Medium Complexity	No	Manual	Funtional Testing
	Data Validation	PM_MASTER_DATA with valid data elements and records	PM_R1_SP3_US174_TC012_Verify that the PM_MASTER_DATA table exists in the database	2: Medium Complexity	No	Manual	Funtional Testing
	Data Validation	PROVIDER_NETWORK_PHYSICIAN_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC013_Verify that the PROVIDER_NETWORK_PHYSICIAN_DTL table exists in the database	2: Medium Complexity	No	Manual	Funtional Testing
	Data Validation	PROVIDER_NETWORK Ancillary_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC014_Verify that the PROVIDER_NETWORK Ancillary_DTL table exists in the database	2: Medium Complexity	No	Manual	Funtional Testing
	Data Validation	PLAN_QUALITY_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC015_Verify that the PLAN_QUALITY_DTL table exists in the database	2: Medium Complexity	No	Manual	Funtional Testing
	Data Validation	CHIP_PREMIUM_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC016_Verify that the CHIP_PREMIUM_DTL table exists in the database	2: Medium Complexity	No	Manual	Funtional Testing

Exceptions	Test Types	To be Included in Regression	Complexities of TC	Manual or Automated
	Functional / Positive GUI Negative Data Validation Interfaces Compatibility	Yes No Does not Apply	1: High Complexity 2: Medium Complexity 3: Low Complexity	Manual Automated Both



# New York State Department of Health

## Test Cases & Defects

User Story Name	Total No of test cases written	Comments
174: As the Plan Management Sprint team, I want to create the data model so that I can accurately store plan information in the Exchange.(Technical Story)	16	

User Story Name	Test Pre Conditions	Test Scenario	Test Case Number & Name	Step number	Test Step name	Expected Results	Test Case/Step Result			Defect ID	Defect Status
							IE 7	IE 8	IE 9		
174: As the Plan Management Sprint team, I want to create the data model so that I can accurately store plan information in the Exchange.(Technical Story)	The ISSUER_DTL table must be present in the database	ISSUER_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC001_Verify that the ISSUER_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					
				2	Run the query to verify the data types and field size of Issuer table: DESC ISSUER_DTL	All the column names with their data types and field size from the ISSUER_DTL table are displayed: ISSUER_ID, ISSUER_LEGAL_NAME, Alpha-Numeric, X(220) FEDERAL_IDENTIFIER, Alpha-Numeric, X(9) NAIC_COMPANY_CODE, Alpha-Numeric, 9(5) NAIC_GROUP_CODE, Alpha-Numeric, 9(5) HHS_ISSUER_ID, Numeric, 9(5) ISSUER_HOLDING_COMPANY_NAME, Alpha-Numeric, X(220) ISSUER_ADDRESS_1, Alpha-Numeric, X(50) ISSUER_ADDRESS_2, Alpha-Numeric, X(50) ISSUER_ADDRESS_3, Alpha-Numeric, X(50) ISSUER_ADDRESS_CITY, Alpha-Numeric, X(25) ISSUER_ADDRESS_STATE, Alpha-Numeric, X(2) ISSUER_ADDRESS_ZIP, Numeric, X(5) ISSUER_WEBSITE, Alpha-Numeric, X(100) ISSUER_STATE, Alpha-Numeric, X(2) ISSUER_THIRD_PARTY_FILER, Boolean, X(1) ISSUER_PHONE, Numeric, X(15) ISSUER_PHONE_EXTENSION, Numeric, X(5) ISSUER_STATE_ID, Alpha-Numeric, X(50) ISSUER_CONSUMER_FACING_WEBSITE, Alpha-Numeric, X(100) INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20)					
				3	Run the query to verify the number of records in Issuer table equals the records obtained from Data Source: Select Count (*) from ISSUER_DTL	The function must return the number of records in ISSUER_DTL table					

				4	Run the query to verify that primary key, "ISSUER_ID and ISSUER_SENT_TIMESTAMP" has unique values: Select ISSUER_ID, ISSUER_SENT_TIMESTAMP, Count (*) From Issuer_DTL Group By ISSUER_ID, ISSUER_SENT_TIMESTAMP Having Count (*) > 1	The query should not return any records					
				5	Run the query to verify the relationship between ISSUER_DTL and PLAN_DTL tables: Select Issuer_DTL.ISSUER_ID, ISSUER_DTL.ISSUER_LEGAL_NAME, Plan_DTL.PLAN_MARKETING_NAME From ISSUER_DTL INNER JOIN Plan_DTL ON Issuer_DTL.ISSUER_ID = Plan_DTL.ISSUER_ID ORDER BY ISSUER_LEGAL_NAME	The function must return all the Issuer with different Plans attached to them					
	The Plan Table must be present in the database	PLAN_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC002_Verify that the PLAN_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					
				2	Run the query to verify the data types and field size of PLAN_DTL table: DESC PLAN_DTL	All the column names with their data types and field size from the PLAN_DTL table are displayed: ISSUER_ID, Alpha-Numeric, X(50) "Plan_ID, Alpha-Numeric, X(50) " PLAN_MARKETING_NAME, Alpha-Numeric, X(220) PLAN_HIOS_PRODUCT_ID, Alpha-Numeric, X(50) PLAN_METAL_LEVEL, Alpha-Numeric, X(15) PLAN_HSA_ELIGIBLE, Boolean, X(1) PLAN_CHILD_ONLY_OFFERING, Boolean, X(1) PLAN_PRODUCT_TYPE, Alpha-Numeric, X(5) PLAN_STND_ALONE_DNTL_PLN_ID, Alpha-Numeric, X(50) PLAN_STND_ALONE_DNTL_PROD_TYPE, Alpha-Numeric, X(5) PLAN_SUM_OF_BENEFITS_CVRG_URL, Alpha-Numeric, X(100) PLAN_ENROLLMENT_PAYMENT_URL, Alpha-Numeric, X(100) PLAN_NEW_OR_EXISTING_IND, Boolean, X(1) PLAN_ENROLLMENT_OPEN_DATE, Date, PLAN_ENROLLMENT_CLOSE_DATE, Date, PLAN_ADMINISTRATIVE_FEES, Float, 9(12,2) PLAN_ADDL_ADMINISTRATIVE_SPECS, Alpha-Numeric, X(500)" PLAN_PRIM_CARE_PHYSICAL_REQD, Boolean, X(1) PLAN_SELF_DIRECTED_ACCOUNT, Boolean, X(1) PLAN_MEDICAL_RECORDS_CVRG, Boolean, X(1) PLAN_OUT-OF-COUNTRY_COVERAGE, Boolean, X(1) PLAN_OUT_OF_SERVICEAREA_CVRG, Boolean, X(1) PLAN_NATIONAL_NETWORK, Boolean, X(1) PLAN_AV_CAL_OUTPUT_NUMBER, Float, 9(4,1)					
				3	Run the query to verify the number of records in PLAN_DTL table equals the records obtained from Data Source: Select Count (*) from PLAN_DTL	The function must return the number of records in Plan table					
				4	Run the query to verify that primary key, "PLAN_ID and PLAN_YEAR" has unique values: Select PLAN_ID, PLAN_YEAR, Count (*) From PLAN_DTL Group By PLAN_ID, PLAN_YEAR Having Count (*) > 1	The query should not return any records					
	The Plan_Service_Area table must be present in the database	PLAN_SERVICE_AREA_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC003_Verify that the PLAN_SERVICE_AREA_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					

				2	Run the query to verify the data types and field size of PLAN_SERVICE_AREA_DTL: DESC PLAN_SERVICE_AREA_DTL	All the column names with their data types and field size from the PLAN_SERVICE_AREA_DTL table are displayed: PLAN_ID, Alpha-Numeric, X(50) PLAN_YEAR, Numeric, X(4) PLAN_SERVICE_AREA, Alpha-Numeric, X(3) PLAN_COUNTY_APPROVAL_STATUS (Only in Staging), Boolean, X(1), TRUE, N PLAN_REJECTION_COMMENTS (Only in Staging), Alpha-Numeric, X(500), TRUE, N INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20) UPDATE_DATE, Date.						
				3	Run the query to verify the number of records in PLAN_SERVICE_AREA_DTL equals the records obtained from Data Source: Select Count (*) from PLAN_SERVICE_AREA_DTL	The function must return the number of records in PLAN_SERVICE_AREA_DTL table						
				4	Run the query to verify the relationship between PLAN_SERVICE_AREA_DTL and PLAN_DTL tables: Select PLAN_SERVICE_AREA_DTL.PLAN_ID, PLAN_SERVICE_AREA_DTL.Service.Service_Area_Code, From PLAN_SERVICE_AREA_DTL INNER JOIN Plan ON PLAN_SERVICE_AREA_DTL.PLAN_ID = PLAN_DTL.PLAN_ID ORDER BY PLAN_ID	The function must return plans allocated with different service areas						
	PLAN_ACCREDITATION_DTL table must be present in the database	PLAN_ACCREDITATION_DTL table with valid data elements and records	PM_R1_SP3_US174_TC004_Verify that the PLAN_ACCREDITATION_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database						
				2	Run the query to verify the data types and field size of PLAN_ACCREDITATION_DTL table: DESC PLAN_ACCREDITATION_DTL	All the column names with their data types and field size from the Plan_Accreditation table are displayed: Plan_ID, Alpha-Numeric, X(50) PLAN_YEAR, Numeric, 9(4) ACCREDITATION_ID, Numeric, X(50) ACCREDITATION_ENTITY_NAME, Alpha-Numeric, X(220) ACCREDITATION_STATUS, Boolean, X(1) ACCREDITATION_MARKET_TYPE, Alpha-Numeric, X(20) ACCREDITATION_PRODUCT, Alpha-Numeric, X(5) ACCREDITATION_SUB_ID, Alpha-Numeric, X(50) INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20) UPDATE_DATE, Date,						
				3	Run the query to verify the number of records in Plan_Accreditation table equals the records obtained from Data Source: Select Count (*) from PLAN_ACCREDITATION_DTL	The function must return the number of records in PLAN_ACCREDITATION_DTL table						
				4	Run the query to verify that primary key, "ACCREDITATION_ID" has unique values: Select ACCREDITATION_ID, Count (ACCREDITATION_ID) From PLAN_ACCREDITATION_DTL Group By ACCREDITATION_ID Having Count (ACCREDITATION_ID) > 1	The query should not return any records						

				5	Run the query to verify the relationship between PLAN_ACCREDITATION_DTL and PLAN_DTL tables: Select PLAN_DTL.PLAN_MARKETING_NAME, Accreditation.ACCREDITATION_ID, Accreditation.ACCREDITATION_ENTITY_NAME, From PLAN_ACCREDITATION_DTL AS Accreditation INNER JOIN PLAN_DTL ON PLAN_ACCREDITATION_DTL.PLAN_ID = Plan.PLAN_ID ORDER BY PLAN_MARKETING_NAME	The function must return all the Plans with their respective accreditation					
	The PLAN_PREMIUM_DTL Table must be present in the database	PLAN_PREMIUM_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC005_Verify that the PLAN_PREMIUM_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					
				2	Run the query to verify the data types and field size of PLAN_PREMIUM_DTL table: DESC PLAN_PREMIUM_DTL	All the column names with their data types and field size from the PLAN_PREMIUM_DTL table are displayed: Plan_ID, Alpha-Numeric, X(50) Plan_Year, Numeric, 9(4) PLAN_SERVICE_AREA, Alpha-Numeric, X(3) PLAN_PREM_TYPE_OF_SUBSCRIBER, Alpha-Numeric, X(3) PLAN_PREM_REGION_IDENTIFIER, Long, PLAN_PREMIUM_RATES, Float, 9(12,2) PLAN_PREMIUM_EFF_DATE, Date, PLAN_PREMIUM_EXP_DATE, Date, INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20) UPDATE_DATE, Date.					
				3	Run the query to verify the number of records in Plan_Premium_Data table equals the records obtained from Data Source: Select Count (*) from PLAN_PREMIUM_DTL	The function must return the number of records in PLAN_PREMIUM_DTL table					
				4	Run the query to verify that unique key, "Rate_Id" has unique values: Select Rate_Id, Count (Rate_Id) From PLAN_PREMIUM_DTL Group By Rate_Id Having Count (Rate_Id) > 1	The query should not return any records					
				5	Run the query to verify the relationship between PLAN_PREMIUM_DTL and PLAN_DTL tables: Select PLAN_DTL.PLAN_ID, PLAN_DTL.PLAN_MARKETING_NAME, Premium.PLAN_PREMIUM_RATES From PLAN_PREMIUM_DTL AS Premium INNER JOIN Plan ON Premium.PLAN_ID = Plan.PLAN_ID ORDER BY PLAN_ID	The function must return the Plans with different premium rates attached to them					
	The FORMULARY_DTL Table must be present in the database	FORMULARY_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC006_Verify that the FORMULARY_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					

				2	Run the query to verify the data types and field size of FORMULARY_DTL table: DESC FORMULARY_DTL	All the column names with their data types and field size from the FORMULARY_DTL table are displayed: Plan_ID, Alpha-Numeric, X(50) Plan_Year, Numeric, 9(4) FORMULARY_ID, Alpha-Numeric, X(50) FORMULARY_VERSION, Alpha-Numeric, X(3) FORMULARY_NAME, Alpha-Numeric, X(220) FORMULARY_MODEL, Alpha-Numeric, X(2) FORMULARY_URL, Alpha-Numeric, X(100) FORMULARY_EFF_DT, Date, FORMULARY_PROVIDER_URL, Alpha-Numeric, X(100) INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20) UPDATE_DATE, Date						
				3	Run the query to verify the number of records in FORMULARY_DTL table equals the records obtained from Data Source: Select Count (*) from FORMULARY_DTL	The function must return the number of records in FORMULARY_DTL table						
				4	Run the query to verify that unique key, "FORMULARY_ID" has unique values: Select FORMULARY_ID, Count (FORMULARY_ID) From FORMULARY_DTL Group By FORMULARY_ID Having Count (FORMULARY_ID) > 1	The query should not return any records						
				5	Run the query to verify the relationship between FORMULARY_DTL and PLAN_DTL tables: Select PLAN_DTL.Plan_Id, PLAN_DTL.PLAN_MARKETING_NAME, FORMULARY_DTL.FORMULARY_ID, FORMULARY_DTL.FORMULARY_NAME, From FORMULARY_DTL INNER JOIN PLAN_DTL ON FORMULARY_DTL.PLAN_ID = PLAN_DTL.PLAN_ID ORDER BY PLAN_ID	The function must return the Plans with their corresponding Formulary details						
	The PLAN_BENEFIT_SERVICES_REF Table must be present in the database	PLAN_BENEFIT_SERVICES_REF Table with valid data elements and records	PM_R1_SP3_US174_TC007_Verify that the PLAN_BENEFIT_SERVICES_REF table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database						
				2	Run the query to verify the data types and field size of PLAN_BENEFIT_SERVICES_REF table: DESC PLAN_BENEFIT_SERVICES_REF	All the column names with their data types and field size from the Benefit_Services table are displayed: BENEFIT_SERVICE_ID, Alpha-Numeric, X(5), FALSE, Y BENEFIT_SERVICE_NAME, Alpha-Numeric, X(220), FALSE, Y BENEFIT_SERVICE_DESC, Alpha-Numeric, X(2000), FALSE, Y BENEFIT_CATEGORY_NAME, Alpha-Numeric, X(500), FALSE, Y INSERT_BY, INSERT_DATE, Date UPDATE_BY, UPDATE_DATE,						
				3	Run the query to verify the number of records in PLAN_BENEFIT_SERVICES_REF table equals the records obtained from Data Source: Select Count (*) from PLAN_BENEFIT_SERVICES_REF	The function must return the number of records in PLAN_BENEFIT_SERVICES_REF table						
				4	Run the query to verify that unique key, "BENEFIT_SERVICE_ID" has unique values: Select BENEFIT_SERVICE_ID, Count (BENEFIT_SERVICE_ID) From PLAN_BENEFIT_SERVICES_REF Group By BENEFIT_SERVICE_ID Having Count (BENEFIT_SERVICE_ID) > 1	The query should not return any records						



				2	Run the query to verify the data types and field size of SBC_SCENARIO_RESULTS_DTL table: DESC SBC_SCENARIO_RESULTS_DTL	All the column names with their data types and field size from the SBC_SCENARIO_RESULTS_DTL table are displayed: PLAN_ID, Alpha-Numeric, X(50) PLAN_YEAR, Numeric, 9(4) SBC_HAVING_BABY_FINAL_PAYMENT, Float, 9(12,2) SBC_HAVING_BABY_DEDUCTIBLE, Float, 9(12,2) SBC_HAVING_BABY_COPAYMENT, Float, 9(12,2) SBC_HAVING_BABY_COINSURANCE, Float, 9(12,2) SBC_HAV_BABY_CSTMR_TOTAL_COST, Float, 9(12,2) SBC_HAVING_BABY_LIMITS, Float, 9(12,2) SBC_TREAT_BRST_CANCER_FNL_PYMT, Float, 9(12,2) SBC_TREAT_BREAST_CANCER_DED, Float, 9(12,2) SBC_TREAT_BREAST_CANCER_COPYMT, Float, 9(12,2) SBC_TREAT_BRST_CANCER_COINS, Float, 9(12,2) SBC_TRET_BRSTCNCR_CST_TOT_COST, Float, 9(12,2) SBC_TREAT_BREAST_CANCER_LIMITS, Float, 9(12,2) SBC_MANAGING_DIABETES_FNL_PYMT, Float, 9(12,2) SBC_MANAGING_DIABETES_DED, Float, 9(12,2) SBC_MANAGING_DIABETES_COPYMT, Float, 9(12,2) SBC_MANAGING_DIABETES_COINS, Float, 9(12,2) SBC_MGN_DIABETES_CST_TOT_COST, Float, 9(12,2) SBC_MANAGING_DIABETES_LIMITS, Float, 9(12,2) SBC_OTHER_FINAL_PAYMENT, Float, 9(12,2) SBC_OTHER_DEDUCTIBLE, Float, 9(12,2) SBC_OTHER_COPAYMENT, Float, 9(12,2) SBC_OTHER_COINSURANCE, Float, 9(12,2) SBC_OTHER_CUSTOMER_TOTAL_COST, Float, 9(12,2)				
				3	Run the query to verify the number of records in SBC_SCENARIO_RESULTS_DTL table equals the records obtained from Data Source: Select Count (*) from SBC_SCENARIO_RESULTS_DTL Table	The function must return the number of records in SBC_SCENARIO_RESULTS_DTL table				
	The SERVICE_AREA_REF table must be present in the database	SERVICE_AREA_REF Table with valid data elements and records	PM_R1_SP3_US174_TC010_Verify that the SERVICE_AREA_REF table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database				
				2	Run the query to verify the data types and field size of SERVICE_AREA_REF table: DESC SERVICE_AREA_REF	All the column names with their data types and field size from the SERVICE_AREA_REF table are displayed: PLAN_SERVICE_AREA, Alpha-Numeric, X(3) PLAN_SERVICE_AREA_DESC, Alpha-Numeric, X(50) STATE_CD, Alpha-Numeric, X(2)				
				3	Run the query to verify the number of records in SERVICE_AREA_REF table equals the records obtained from Data Source: Select Count (*) from SERVICE_AREA_REF	The function must return the number of records in SERVICE_AREA_REF table				
				4	Run the query to verify that primary key, "PLAN_SERVICE_AREA" has unique values: Select PLAN_SERVICE_AREA, Count (PLAN_SERVICE_AREA) From SERVICE_AREA_REF Group By PLAN_SERVICE_AREA Having Count (PLAN_SERVICE_AREA) > 1	The query should not return any records				
				5	Run the query to verify the relationship between SERVICE_AREA_REF and ZIP_SERVICE_AREA_REF tables: Select Zip.ZIP_CODE, Service.PLAN_SERVICE_AREA, Service.PLAN_SERVICE_AREA_DESC From SERVICE_AREA_REF AS Service INNER JOIN ZIP_SERVICE_AREA_REF AS Zip ON Service.PLAN_SERVICE_AREA = Zip.Service_Area_Code ORDER BY Zip_Code	The function must return zip codes with associated service areas				

				6	Run the query to verify the relationship between SERVICE_AREA_REF and Plan_Service_Area tables: Select Service.PLAN_SERVICE_AREA, Service.Service_County_Name, Plan_Service_Area.Plan_Id From SERVICE_AREA_REF AS Service INNER JOIN Plan_Service_Area ON Service.Service_Area_Code = Plan_Service_Area.Service_Area_Code ORDER BY Plan_Id	The function must return plans allocated with different service areas					
	The Zip_Service_Area Table must be present in the database	ZIP_SERVICE_AREA_REF Table with valid data elements and records	PM_R1_SP3_US174_TC011_Verify that the ZIP_SERVICE_AREA_REF table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					
				2	Run the query to verify the data types and field size of ZIP_SERVICE_AREA_REF table: DESC ZIP_SERVICE_AREA_REF	All the column names with their data types and field size from the ZIP_SERVICE_AREA_REF table are displayed: ZIP_CODE, Numeric, X(5) PLAN_SERVICE_AREA, Alpha-Numeric, X(3)					
				3	Run the query to verify the number of records in ZIP_SERVICE_AREA_REF table equals the records obtained from Data Source: Select Count (*) from ZIP_SERVICE_AREA_REF	The function must return the number of records in ZIP_SERVICE_AREA_REF table					
	The PM_Master_Data Table must be present in the database	PM_MASTER_DATA with valid data elements and records	PM_R1_SP3_US174_TC012_Verify that the PM_MASTER_DATA table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					
				2	Run the query to verify the data types and field size of PM_MASTER_DATA table: DESC PM_MASTER_DATA	All the column names with their data types and field size from the PM_Master_Data table are displayed: MASTER_DATA_CODE, Alpha-Numeric, X(3) MASTER_DATA_VALUE, Alpha-Numeric, X(250) MASTER_DATA_DESC, Alpha-Numeric, X(250) MASTER_DATA_TYPE, Alpha-Numeric, X(3) INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20) UPDATE_DATE, Date					
				3	Run the query to verify that primary key, "MASTER_DATA_CODE" has unique values: Select MASTER_DATA_CODE, Count (MASTER_DATA_CODE) From MASTER_DATA_CODE Group By MASTER_DATA_CODE Having Count (MASTER_DATA_CODE) > 1	The query should not return any records					
				4	Run the query to verify that primary key, MASTER_DATA_VALUE has unique values: Select MASTER_DATA_VALUE, Count (MASTER_DATA_VALUE) From MASTER_DATA_VALUE Group By MASTER_DATA_VALUE Having Count (MASTER_DATA_VALUE) > 1	The query should not return any records					
	PROVIDER_NETWORK_PHYSICIAN_DTL Table must be present in the database	PROVIDER_NETWORK_PHYSICIAN_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC013_Verify that the PROVIDER_NETWORK_PHYSICIAN_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					

				2	Run the query to verify the data types and field size of PROVIDER_NETWORK_PHYSICIAN_DTL table: DESC PROVIDER_NETWORK_PHYSICIAN_DTL	All the column names with their data types and field size from the PROVIDER_NETWORK_PHYSICIAN_DTL table are displayed: Plan_ID, Alpha-Numeric, X(50) PLAN_YEAR, Numeric, 9(4) LAST_NAME, Alpha-Numeric, X(25) FIRST_NAME, Alpha-Numeric, X(15) NATIONAL_PROVIDER_IDENTIFIER, Alpha-Numeric, X(10) LICENSE_NUMBER, Alpha-Numeric, X(8) SITE_NAME, Alpha-Numeric, X(50) ROOM_OR_SUITE, Alpha-Numeric, X(20) STREET_ADDRESS, Alpha-Numeric, X(49) TOWN_CITY, Alpha-Numeric, X(30) STATE, Alpha-Numeric, X(2) SERVICE_AREA_CODE, Alpha-Numeric, X(3) ZIP_CODE, Alpha-Numeric, X(5) ZIP_EXTENSION, Alpha-Numeric, X(4) WHEEL_CHAIR_ACCESSIBILITY, Alpha-Numeric, X(1) PRIMARY_DESIGNATION, Alpha-Numeric, X(1) PRIMARY_SPECIALITY, Alpha-Numeric, X(3) SECONDARY_SPECIALITY, Alpha-Numeric, X(3) BOARDSTATUS_PRIMARY_SPECIALITY, Alpha-Numeric, X(1) BOARDSTATUS_SECNDRY_SPECIALITY, Alpha-Numeric, X(1) GENDER, Alpha-Numeric, X(1) MEDICAID_PANEL_STATUS, Alpha-Numeric, X(1) MEDICARE_PANEL_STATUS, Alpha-Numeric, X(1) CHP_PANEL_STATUS, Alpha-Numeric, X(1) FHP_PANEL_STATUS, Alpha-Numeric, X(1)				
				3	Run the query to verify the number of records in PROVIDER_NETWORK_PHYSICIAN_DTL table equals the records obtained from Data Source: Select Count (*) from PROVIDER_NETWORK_PHYSICIAN_DTL	The function must return the number of records in PROVIDER_NETWORK_PHYSICIAN_DTL				
	The PROVIDER_NETWORK Ancillary_DTL Table must be present in the database	PROVIDER_NETWORK Ancillary_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC014_Verify that the PROVIDER_NETWORK Ancillary_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database				
				2	Run the query to verify the data types and field size of PROVIDER_NETWORK Ancillary_DTL table: DESC PROVIDER_NETWORK Ancillary_DTL	All the column names with their data types and field size from the PROVIDER_NETWORK Ancillary_DTL table are displayed: PLAN_ID, Alpha-Numeric, X(50) PLAN_YEAR, Numeric, 9(4) SITE_NAME, Alpha-Numeric, X(50) Room_Or_Suite, Alpha-Numeric, X(20) ROOM_OR_SUITE, Alpha-Numeric, X(50) TOWN_CITY, Alpha-Numeric, X(30) STATE, Alpha-Numeric, X(2) SERVICE_AREA_CODE, Alpha-Numeric, X(3) ZIP_CODE, Alpha-Numeric, X(5) ZIP_EXTENSION, Alpha-Numeric, X(4) DESIGNATED_SERVICE_CODE, Alpha-Numeric, X(3) NATIONAL_PROVIDER_IDENTIFIER, Alpha-Numeric, X(10) LICENSE_NUMBER, Alpha-Numeric, X(8) PERMANENT_FACILITY_IDENTIFIER, Alpha-Numeric, X(4) AREA_CODE, Alpha-Numeric, X(3) PHONE_NUMBER, Alpha-Numeric, X(7) SERVICE_1, Alpha-Numeric, X(3) SERVICE_2, Alpha-Numeric, X(3) SERVICE_3, Alpha-Numeric, X(3) SERVICE_4, Alpha-Numeric, X(3) SERVICE_5, Alpha-Numeric, X(3) SERVICE_6, Alpha-Numeric, X(3) SERVICE_7, Alpha-Numeric, X(3) SERVICE_8, Alpha-Numeric, X(3) SERVICE_9, Alpha-Numeric, X(3)				

				3	Run the query to verify the number of records in PROVIDER_NETWORK Ancillary_DTL table equals the records obtained from Data Source: Select Count (*) from PROVIDER_NETWORK Ancillary_DTL	The function must return the number of records in PROVIDER_NETWORK Ancillary_DTL table					
	The PLAN_QUALITY_DTL Table must be present in the database	PLAN_QUALITY_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC015_Verify that the PLAN_QUALITY_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					
				2	Run the query to verify the data types and field size of PLAN_QUALITY_DTL table: DESC PLAN_QUALITY_DTL	All the column names with their data types and field size from the Plan_Quality_Data table are displayed: PLAN_ID, Alpha-Numeric, X(50) PLAN_YEAR, Alpha-Numeric, 9(4) RATING, Alpha-Numeric, X(2) DOMAIN, Alpha-Numeric, X(25) SCORE, Alpha-Numeric, X(10) MEASURE, Alpha-Numeric, X(30) RATE, Alpha-Numeric, X(4) LEVEL_SIGNIFICANCE, Alpha-Numeric, X(2) INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20) UPDATE_DATE, Date, SOURCE_SENT_TIMESTAMP, Timestamp.					
				3	Run the query to verify the number of records in PLAN_QUALITY_DTL table equals the records obtained from Data Source: Select Count (*) from PLAN_QUALITY_DTL	The function must return the number of records in PLAN_QUALITY_DTL table					
	The CHIP_PREMIUM_DTL Table must be present in the database	CHIP_PREMIUM_DTL Table with valid data elements and records	PM_R1_SP3_US174_TC016_Verify that the CHIP_PREMIUM_DTL table exists in the database	1	Login the given database with valid User ID and Password	User should be able to login the database					
				2	Run the query to verify the data types and field size of CHIP_PREMIUM_DTL table: DESC CHIP_PREMIUM_DTL	All the column names with their data types and field size from the CHIP_PREMIUM_DTL table are displayed: CHILD_ROW_ID, Numeric, 9(2) INCOME_PERCENT_START, Numeric, 9(3) INCOME_PERCENT_END, Numeric, 9(3) RATE_PER_CHILD, Float, 9(12,2) MAXIMUM_FAMILY_RATE, Float, 9(12,2) RATE_BAND_EFFECTIVE_DATE, Date, RATE_BAND_EXPIRATION_DATE, Date, STATUS, Boolean, X(1) INSERT_BY, Alpha-Numeric, X(20) INSERT_DATE, Date, UPDATE_BY, Alpha-Numeric, X(20) UPDATE_DATE, Date, SOURCE_SENT_TIMESTAMP, Timestamp.					
				3	Run the query to verify the number of records in CHIP_PREMIUM_DTL table equals the records obtained from Data Source: Select Count (*) from CHIP_PREMIUM_DTL	The function must return the number of records in CHIP_Premuim_Data table					





























Test Case/Step Result

Passed  
Failed

Manual or Automated

Manual  
Automated

Not Executed

Both

New  
Open  
Fixed  
Ready to test  
Closed

Written By	Comments























































































































































































