Perceptive Reach

Integrated Reach Database System

(IRDS)

Database Design Specification



Department of Veterans Affairs

July 2015

Version 2.9

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 7/13/2015 | 2.9 | Final Review | Monica Mohler/Paul Bradley |  |  |
| 7/8/2015 | 2.8 | Peer Review | Radina Ivanova |  |  |
| 7/8/2015 | 2.7 | Updated data Dictionary and added content about performance views | Bill Balshem |  |  |
| 6/12/2015 | 2.6 | Final Review | Monica Mohler/Paul Bradley |  |  |
| 6/4/2015 | 2.5 | Peer Review | Radina Ivanova |  |  |
| 6/3/2015 | 2.4 | Updated Data Dictionary, Added 6500 compliance items | Bill Balshem |  |  |
| 5/14/2015 | 2.3 | Final Review | Monica Mohler/Paul Bradley |  |  |
| 5/6/2015 | 2.2 | Peer Review | Radina Ivanova |  |  |
| 5/4/2015 | 2.1 | Updates to Data Dictionary | Bill Balshem |
| 4/13/2015 | 2.0 | Final Review | Monica Mohler/Paul Bradley |
| 4/2/2015 | 1.9 | Peer Review | Radina Ivanova |
| 4/1/2015 | 1.8 | Updated physical Model & Data Dictionary | Bill Balshem |
| 3/11/2015 | 1.7 | Final Review | Monica Mohler/Paul Bradley |
| 3/3/2015 | 1.6 | Peer Review | Radina Ivanova |
| 3/2/2015 | 1.5 | Updates to Data Model & Business Process | Bill Balshem |
| 2/13/2015 | 1.4 | Final Review | Monica Mohler/Paul Bradley |
| 2/12/2015 | 1.3 | Peer Review | Radina Ivanova |
| 02/10/2015 | 1.2 | Updates to Physical Data Model, Logical Data Model & Data Dictionary | Bill Balshem |
| 01/08/2015 | 1.1 | Added Node JS Data API content | Bill Balshem |
| 12/11/2014 | 1.0 | Final Review | Monica Mohler/Paul Bradley |
| 12/10/2014 | .02 | Peer Review | Radina Ivanova |
| 12/09/2014 | .01 | Initial Version | Bill Balshem |

**Table of Contents**

[1. About this document 5](#_Toc424140982)

[2. Introduction 5](#_Toc424140983)

[2.1. Purpose 5](#_Toc424140984)

[2.2. Scope, Approach and Methods 5](#_Toc424140985)

[2.3. Acronyms and Abbreviations 6](#_Toc424140986)

[3. System Overview 7](#_Toc424140987)

[3.1. Business Process 8](#_Toc424140988)

[3.2. System Information 10](#_Toc424140989)

[3.2.1. Hardware Requirements 10](#_Toc424140990)

[3.2.2. Support Software 10](#_Toc424140991)

[3.3. Architecture 11](#_Toc424140992)

[3.3.1. Software Architecture 11](#_Toc424140993)

[3.3.2. Interfaces 12](#_Toc424140994)

[3.3.3. Data Stores 12](#_Toc424140995)

[4. Database Design Decisions 12](#_Toc424140996)

[4.1. Assumptions 13](#_Toc424140997)

[4.2. Issues 13](#_Toc424140998)

[4.3. Constraints 13](#_Toc424140999)

[5. Database Administrative Functions 13](#_Toc424141000)

[5.1. Naming Conventions 13](#_Toc424141001)

[5.2. Database Identification 13](#_Toc424141002)

[5.3. Schema Information 14](#_Toc424141003)

[5.3.1. Description 14](#_Toc424141004)

[5.3.2. Logical Data Model 14](#_Toc424141005)

[5.3.3. Physical Data Model 17](#_Toc424141006)

[5.3.4. Data Dictionary 18](#_Toc424141007)

[5.3.5. Veteran De-Duping Process 30](#_Toc424141008)

[5.3.6. Stored Procedures 31](#_Toc424141009)

[5.3.7. Performance Views 32](#_Toc424141010)

[5.4. Denormalization 32](#_Toc424141011)

[5.5. Performance Improvement 32](#_Toc424141012)

[5.6. Storage 32](#_Toc424141013)

[5.7. Recovery 32](#_Toc424141014)

[6. Database Interfaces 32](#_Toc424141015)

[6.1. Suicide Data Repository (SDR) 32](#_Toc424141016)

[6.2. Perceptive Reach Dashboard 33](#_Toc424141017)

[7. Data Access 33](#_Toc424141018)

[7.1. Role Definitions (System) 34](#_Toc424141019)

[7.2. Role Definitions (Dashboard) 34](#_Toc424141020)

[7.3. Node JS Data APIs 34](#_Toc424141021)

[7.3.1. API listing 35](#_Toc424141022)

[8. Implementation Considerations 36](#_Toc424141023)

[8.1. Large Objects 36](#_Toc424141024)

[8.2. Partitioning 37](#_Toc424141025)

[8.3. Error Processing 37](#_Toc424141026)

# About this document

This document is a “work in progress.” The database will be designed through a series of sprint cycles in collaboration with VA stakeholders and users. The document will be populated with content as the design evolves with each sprint, including all potential data options, configurability rules and options. The early versions of the document will focus on the conceptual design, key business requirements, and the primary end user features related to the design. Future versions will add additional detail as it becomes available.

# Introduction

VA is seeking to expand suicide prevention to include upstream approaches, designed to reduce initiation or escalation of a risk factor. Upstream suicide interventions target individuals or groups who exhibit biological, psychological, or social risk factors that are more prominent among high-risk groups than among the larger population. Understanding the unique needs of our nation’s Veterans and the military culture as it relates to stigma and mental health is important for early intervention. The goal of the Integrated Reach Database System (IRDS) innovation is to promote the general health of the Veteran population and effectively intervene in issues before they escalate in crisis.

The IRDS solution innovates the current process of risk data collection, analysis, and use in effective intervention strategy. The solution will harness the power of large and diverse data stores to aggregate, analyze and identify risk onset as well as reveal previously unidentified at-risk individuals and populations as a holistic and integrated approach.

The IRDS innovation will serve to bolster the three major components of VHA’s Strategic Plan for Suicide Prevention: surveillance, risk and protective factors, and prevention interventions. The IRDS innovation will target antecedent events specific to Veteran populations prior to the onset of risk to mitigate the development of risk.

## Purpose

The purpose of the Database Design Specification is to give detailed documentation of the Data and Database components of the IRDS system.

## Scope, Approach and Methods

This document covers the following items:

* Database Properties and Configuration
* Database Conceptual Design
* Database Objects and Table Schemas
* External Data Sources and Interfaces

NOTE: This is a working document. As access to data sources becomes available and technical requirements are further defined, the document will be updated with details such as, database configuration, indexing and the types of data transformations that will take place when importing data from sources into the IRDS system.

## Acronyms and Abbreviations

**Table 1: Acronyms and Abbreviations**

| Acronym | Term |
| --- | --- |
| API | Application Programming Interface |
| CDW | Corporate Data Warehouse |
| DOB | Date of Birth |
| EDW | Enterprise Data Warehouse |
| ETL | Extract, Transform, Load |
| GB | Gigabyte |
| HDD | Hard Disk Drive |
| ICD | International Classification of Diseases |
| ID | Identification |
| IRDS | Integrated Reach Database System |
| JSON | JavaScript Object Notation |
| KNIME | Konstanz Information Miner |
| MS | Microsoft Server |
| MUMPS | Massachusetts General Hospital Utility Multi-Programming System |
| NDI | National Death Index |
| NODE JS | An open source, cross-platform runtime environment for server-side and networking applications |
| PACER | Public Access to Court Electronic Records |
| POC | Point of Contact |
| PTSD | Post-Traumatic Stress Disorder |
| RAM | Random Access Memory |
| RPC | Remote Procedure Call |
| SDCD | State Death Certificate Data |
| SDR | Suicide Data Repository |
| SFTP | Secure File Transfer Protocol |
| SPAN | Suicide Prevention Applications Network |
| SQL | Structured Query Language |
| SSIS | SQL Server Integration Services |
| SSMS | SQL Server Management Studio |
| SSN | Social Security Number |
| TBD | To Be Determined |
| TBI | Traumatic Brain Injury |
| T-SQL | Transact-SQL |
| VA | Department of Veterans Affairs |
| VAMC | VA Medical Center |
| VBA | Veterans Benefits Administration |
| VCL | Veterans Crisis Line |
| VHA | Veterans Health Administration |
| VISN | Veterans Integrated Service Networks |
| VistA | Veterans Health Information Systems and Technology Architecture |
| VLER | Virtual Lifetime Electronic Record |
| VR&E | Vocational Rehabilitation and Employment |

# System Overview

The main components of the IRDS system are:

* **Reach Database -**  A SQL Server database on the IRDS server that stores data from the various VA sources imported into the system
* **Analytics Risk Model/Surveillance Model** – A program written in R that will run periodically to update a list of high risk factors associated with Veteran suicide. These factors will be persisted in one or more tables in the Reach database. A SQL process will run on a regular basis to monitor Veterans as being high risk for suicide based on factors as determined by the Analytics Risk model
* **Direct Messaging** -. As Veterans are identified by the surveillance model as being high risk, the VA staff will be notified via the direct messaging component of the IRDS system, which leverages the VLER solution adopted by the VA
* **Perceptive Reach Dashboard** – VA staff can login to the dashboard to read messages and review data associated with their regional or functional responsibility within the VA (i.e. region, facility)

## Business Process

1. Data sources will be imported into the staging area of Reach database on IRDS server via SQL Server Integration Services (SSIS) import solution. Each data source will have its own SSIS package (.dtsx). The import solution will transform the data and load it into the appropriate tables in the Reach database. NOTE: For CDW data the staging area will be the IRDS project database (VACI\_IRDS) on the CDW server VHACDWA01. This is where the transformations will take place.
2. By analyzing the data sources, the Perceptive Reach Risk Model will be developed, which will be persisted as a set that of risk factor/coefficient pairs to be stored in the Reach database. Every year or so the Model will be reviewed and possibly enhanced, leading to updates to the coefficient values for the risk factors.factors. On a regular basis (daily, weekly) a SQL Server process will run that will do surveillance against a list of Veterans tracked in the Reach database, against the variables in the Risk Model. A risk score is calculated for each Veteran and that value is placed in the record for that Veteran in the ‘Patient’ table, a master list of all Veterans imported into the IRDS system. After the scoring is complete, at risk Veterans will be identified.
3. For each Veteran identified as a at risk for suicide, for the first time, the system will notify appropriate VA authorized outreach and intervention staff member(s) through a secure message, leveraging the VA Virtual Lifetime Electronic Record (VLER) Health Direct messaging service. In addition, detailed information about each at risk veteran, such as emergency contact and medication data will be imported into the Reach database for the purpose of being viewed in the Perceptive Reach dashboard.
4. An IRDS user opens up the dashboard via a compatible web browser and a client side java component connects to a server side java component(residing on the IRDS Application Server), which queries the Reach (residing on the IRDS Database Server) database for both specific and aggregate data regarding at risk Veterans at the users’ management level (region, state, VISN, VAMC). The query results are passed to the client browser and populated in the web page.

**Figure 1: Business Process Diagram**

## System Information

### Hardware Requirements

**Table 2: Hardware Requirements**

| Hardware Component | Requirements |
| --- | --- |
| Processor | Intel Xeon E5-2600 Family (2670 or 2690), 2.6GHZ |
| RAM/Memory | 64GB |
| Storage/HDD | 500GB |

NOTE: The IRDS is a virtualized solution; the table above is a list of resources required for the virtual solution.

### Support Software

**Table 3: Support Software**

| Product | Version | Purpose |
| --- | --- | --- |
| Microsoft (MS) Windows | 2008 R2 or later | Operating System |
| Microsoft (MS) SQL Server | 2012 | Database platform and data store |
| MS SQL Server Integration Services (SSIS) |  | Data import solution development tool Performs ETL work. |
| Transact SQL (T-SQL) |  | Query language native to SQL Server |
| Attachmate Reflection |  | SFTP software to download data files onto the IRDS server. |
| Python | 2.7 | Programming language. Used for Individual de-duping functionality |
| R | 3.1.2 | Programming language. Program will be developed in R to analyze Reach database data and develop Risk Model and store results in a SQL table. |
| Node JS | 0.10.33 | Tool to develop server side component of the dashboard. The dashboard sends requests to the Node JS, which queries the database and returns the result back to the client browser. |
| Knime | 2.10.4 | Data mining tool. VA staff can connect to the Reach database through Knime to do ad-hoc reporting. |

## Architecture

### Software Architecture

* **Data Imports** – SSIS will be the primary tool for importing external data sources into the IRDS Reach database. For a specific data import, an SSIS package will be developed to

1. Make a connection to the source (SQL table, text file, other)
2. Import the data into a staging area
3. Make the appropriate data transformations (cleaning, standardization)
4. Load the transformed data into the appropriate Reach data store tables

The execution of SSIS packages (.dtsx files) can be automated by scheduling them as a Windows process via SQL Server Agent.

* **Remote Procedure Calls (RPCs) –** The VA uses the Veterans Health Information Systems and Technology Architecture (Vista) system, for managing Veterans health data. Data will be imported into the IRDS system directly from VistA using RPC calls. VistA data is stored against a MUMPS back end, which uses text based files for data storage. For each set of VistA data imported into IRDS:

1. Either a custom RPC will be written (in M) or a currently existing one will be leveraged
2. An automated java process will execute the RPC and return the query results in text format
3. Those results will be stored in a flat file on the IRDS server to be imported into the reach database via an SSIS package

**Figure 2: IRDS Database Software Architecture**

### Interfaces

* **Corporate Data Warehouse (CDW)** – Data warehouse for VHA data, primarily VistA data. Records are stored in SQL Server tables.
* **Suicide Data Repository (SDR)** – Suicide and Mortality Data from four sources are stored on a centralized server in SQL Server tables. Those sources are: Mortality Search Results from the National Death Index (NDI), State Death Certificate Data (SDCD), Veterans Crisis Line (VCL), Suicide Prevention Applications Network (SPAN)
* **Veterans Health Information Systems and Technology Architecture (VISTA)** – Open source enterprise system used by VHA network. Data is stored in a MUMPS back end.
* **VA Enterprise Data Warehouse (EDW)** – Data warehouse that stores VBA data.

NOTE: At this time there is an attempt to include VBA data elements housed at EDW in the IRDS system. It is unclear whether access to the data will be available within timeframe set to develop the IRDS system.

* **Perceptive Reach Dashboard** – Users of the IRDS system will connect through a web based dashboard.
* **Analytics Sandbox** – VA users will be able to run ad-hoc reporting by connecting to the IRDS database using tools such as R or Knime.
* **Data Sources External to VA** – The IRDS system may potentially pull data from one or more outside sources such as LexisNexis or PACER.

### Data Stores

* All data stored in the IRDS system will be captured in SQL Server tables.
* In the event that some data will be accessed via a file format, such as FLAT files, those files will be uploaded to the IRDS server via SFTP and placed in a specified location on the IRDS server file system, to be processed for import.

# Database Design Decisions

The decision was made to use SQL Server as opposed to other data platforms such as MySQL or Microsoft Access because:

* Microsoft SQL Server is approved and is widely used within the VA
* Several of the systems from which data will be imported are stored in SQL Server tables
* The IRDS development team has expertise in SQL Server and has successfully developed and deployed the SDR system into production
* The platform is scalable enough to handle the size and performance requirements that are expected for this system

The initial database will be configured to the default configuration for a SQL server database. As technical requirements evolve, that setup will change to accommodate the updated specifications (i.e., breaking the database file into multiple files).

## Assumptions

There are currently no technical assumptions being made for the Database portion of the IRDS system.

## Issues

There are currently no issues for the Database portion of the IRDS system.

## Constraints

There are currently no constraints for the Database portion of the IRDS system.

# Database Administrative Functions

## Naming Conventions

**Table 4: Database Naming Conventions**

| Type | Guideline |
| --- | --- |
| Reference Tables | Begin with “Ref\_” (ex: Ref\_Gender) |
| Table names | Mountain style (ex: SuicideEvents) |
| Constraints | Begin with constraint abbreviation then underscore (foreign key example: fk\_VetID) |
| Unique Identifier | All tables will have an ID field of type integer – identity. |

## Database Identification

**Table 5: Database Identification**

| Element | Element Name | Meaning |
| --- | --- | --- |
| db\_name | Reach | Production/Master Database |
| db\_name | Reach\_Dev | Development database |
| db\_name | Reach\_Test | Test/Pre-production database |
| db\_path | D:\MSSQL11.MSSQLSERVER\MSSQL\DATA | The full path to the location where the database is stored on the system. |
| db\_file | Reach.mdf | Database filename |
| db\_log | Reach\_log.ldf | Database log file |

## Schema Information

### Description

The Reach database will contain the following schemas:

* .dbo – The MS SQL Server default schema, will contain data imported from VA data sources, such as the VA Suicide Data Repository (SDR). Reference data will be stored here as well, such as a list of VAMC and a list of ICD codes. The risk model output will be persisted into one or more tables in this schema as well.
* SSIS – Will contain tables created during source data import processes, developed using MS SQL Server Integration Services (SSIS). The records created by the last run per SSIS solution (.dtsx) will be persisted in the SSIS schema tables, leaving a footprint available for troubleshooting and debugging
* PRSystem – Tables used by the IRDS dashboard, such as a list of preferences for each dashboard user.
* Staging – This will be the staging area for data imported into IRDS before it is transformed by SSIS import packages and imported into tables described in the .dbo schema

### Logical Data Model

The Reach Database contains the following tables:

Data Tables

**Patient** - The master list of Veterans that will be scored for suicide risk in the IRDS system. This table contains:

* Reach ID - Unique IRDS system ID (all child tables to the Patient table will be linked by Reach ID)
* VA Identifiers - Patient ICN
* Demographics - Name, SSN, DOB
* Current Risk Score and Risk Level
* Date First Identified as High Risk

**RiskFactors** - The Risk Factors table will store risk factors for each Veteran, which will be used to calculate that Veteran's risk score. One record will contain all of the risk factor values for a Veteran. There will be a 1 to 1 relationship between the Patient table and the RiskFactors table.

**ScoreHistory** - Each time risk scores are re-run for the Veterans the old scores will be moved to a ScoreHistory table, so each Veteran score can be mapped over time. The table will have a 1 to many relationship with the Patient table.

**Veteran details tables** - Any data that will be displayed in the IRDS dashboard for high risk Veterans will be stored in one to many child tables to the Patient tables. Ex: There will be an emergency contact table in the Reach database. Any time a Veteran becomes identified as high risk, the emergency contact data for that Veteran will be added to an EmergencyContact table.

Reference Tables

Examples are lists to be used for reporting and normalizing of the data such as a list VAMCs and a list of ICD Codes that contain a diagnosis description associated with each code.

VeteranStation

Each Veteran will be tied to one or more locations in the IRDS system. This way, aggregate details about Veterans will be able to be rolled up to Facility, VISN and state levels in the dashboard. All records in the CDW data have the specific VAMC where a Veteran: had a visit, was prescribed medication, was diagnosed with a condition, etc. This value is stored in the Sta3N field in CDW tables. A list of VAMC’s (Sta3N) for where each Veteran has had some kind of activity in the past 2 years will be stored in the Reach database and accessible to the dashboard when doing data aggregation at a specific level for Veterans.

System Tables

Tables used by the dashboard such as User roles and Preferences (see Data Access).

Tables for 6500 Compliance

To satisfy VA Handbook 6500, item AU3, IRD must meet VA 6500 audit record content requirements, the Reach database will have tables to capture data entry by the users of the IRDS Dashboard: These tables will contain the following elements:

* Patient’s ReachID (if the data entry pertains to a specific Patient with the Reach system)
* Dashboard user’s UserID (from the prsystem.User table)
* Timestamp – Date/time that the user updated the record in the system
* The data entry value entered by the user



**Figure 3: IRDS Reach Database Logical Data Model**

### Physical Data Model



### Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| TABLE NAME | COLUMN NAME | DATA TYPE | NULLS? |
| dbo.Appointments | ReachID | int | N |
| dbo.Appointments | ApptDate | date | Y |
| dbo.Appointments | CancelNoShowCode | varchar(5) | Y |
| dbo.Appointments | PrimarySecondaryStopCodeName | varchar(125) | Y |
| dbo.EmergencyContact | ReachID | int | N |
| dbo.EmergencyContact | NameOfContact | varchar(50) | N |
| dbo.EmergencyContact | StreetAddress1 | varchar(50) | Y |
| dbo.EmergencyContact | StreetAddress2 | varchar(50) | Y |
| dbo.EmergencyContact | StreetAddress3 | varchar(50) | Y |
| dbo.EmergencyContact | City | varchar(50) | Y |
| dbo.EmergencyContact | State | varchar(30) | Y |
| dbo.EmergencyContact | Zip | varchar(50) | Y |
| dbo.EmergencyContact | Zip4 | varchar(50) | Y |
| dbo.EmergencyContact | Phone | varchar(50) | Y |
| dbo.EmergencyContact | PhoneWork | varchar(50) | Y |
| dbo.OutreachStatusHistory | ReachID | int | Y |
| dbo.OutreachStatusHistory | OutReachStatus | smallint | Y |
| dbo.OutreachStatusHistory | UpdateDate | datetime | Y |
| dbo.OutreachStatusHistory | UpdateUser | int | Y |
| dbo.Patient | ReachID | int | N |
| dbo.Patient | LastName | varchar(50) | Y |
| dbo.Patient | FirstName | varchar(50) | Y |
| dbo.Patient | SSN | varchar(50) | Y |
| dbo.Patient | Address1 | varchar(50) | Y |
| dbo.Patient | Address2 | varchar(50) | Y |
| dbo.Patient | Address3 | varchar(50) | Y |
| dbo.Patient | City | varchar(50) | Y |
| dbo.Patient | State | varchar(30) | Y |
| dbo.Patient | Zip | varchar(50) | Y |
| dbo.Patient | Zip4 | varchar(50) | Y |
| dbo.Patient | HomePhone | varchar(50) | Y |
| dbo.Patient | WorkPhone | varchar(50) | Y |
| dbo.Patient | CellPhone | varchar(50) | Y |
| dbo.Patient | MothersMaidenName | varchar(50) | Y |
| dbo.Patient | DateofDeath | date | Y |
| dbo.Patient | EmailAddress | varchar(50) | Y |
| dbo.Patient | Race | smallint | Y |
| dbo.Patient | Gender | char(1) | Y |
| dbo.Patient | PatientICN | varchar(50) | Y |
| dbo.Patient | MaritalStatus | smallint | Y |
| dbo.Patient | DOB | date | Y |
| dbo.Patient | RiskScore | decimal(11,10) | Y |
| dbo.Patient | RiskLevel | smallint | Y |
| dbo.Patient | DateIdentifiedAsAtRisk | date | Y |
| dbo.Patient | MilitaryBranch | smallint | Y |
| dbo.Patient | OutreachStatus | smallint | Y |
| dbo.Patient | ScoringStatus | smallint | Y |
| dbo.Patient | MilitaryEra | smallint | Y |
| dbo.Patient | HighRisk | bit | Y |
| dbo.Patient | SafetyPlanCurrent | bit | Y |
| dbo.PatientDiagnosis | ReachID | int | N |
| dbo.PatientDiagnosis | ICD\_Code | varchar(50) | N |
| dbo.PatientDiagnosis | ICD\_Desc | varchar(255) | Y |
| dbo.PatientDiagnosis | DiagnosisDate | date | Y |
| dbo.PatientMedications | ReachID | int | N |
| dbo.PatientMedications | MedicationName | varchar(100) | N |
| dbo.PatientMedications | Active | bit | Y |
| dbo.PatientRiskFactors | ReachID | int | N |
| dbo.PatientRiskFactors | alprazolam12 | bit | Y |
| dbo.PatientRiskFactors | alprazolam24 | bit | Y |
| dbo.PatientRiskFactors | ami12 | bit | Y |
| dbo.PatientRiskFactors | ami24 | bit | Y |
| dbo.PatientRiskFactors | amp12 | bit | Y |
| dbo.PatientRiskFactors | amp24 | bit | Y |
| dbo.PatientRiskFactors | analgesic12 | bit | Y |
| dbo.PatientRiskFactors | analgesic24 | bit | Y |
| dbo.PatientRiskFactors | anticonvulsant12 | bit | Y |
| dbo.PatientRiskFactors | anticonvulsant24 | bit | Y |
| dbo.PatientRiskFactors | antidep12 | bit | Y |
| dbo.PatientRiskFactors | antidep24 | bit | Y |
| dbo.PatientRiskFactors | antipsy12 | bit | Y |
| dbo.PatientRiskFactors | antipsy24 | bit | Y |
| dbo.PatientRiskFactors | anyattempt1 | bit | Y |
| dbo.PatientRiskFactors | anyattempt12 | bit | Y |
| dbo.PatientRiskFactors | anyattempt18 | bit | Y |
| dbo.PatientRiskFactors | anyattempt2 | bit | Y |
| dbo.PatientRiskFactors | anyattempt24 | bit | Y |
| dbo.PatientRiskFactors | anyattempt3 | bit | Y |
| dbo.PatientRiskFactors | anyattempt6 | bit | Y |
| dbo.PatientRiskFactors | AnyEDvisits\_prior1 | bit | Y |
| dbo.PatientRiskFactors | AnyEDvisits\_prior12 | bit | Y |
| dbo.PatientRiskFactors | AnyEDvisits\_prior18 | bit | Y |
| dbo.PatientRiskFactors | AnyEDvisits\_prior2 | bit | Y |
| dbo.PatientRiskFactors | AnyEDvisits\_prior24 | bit | Y |
| dbo.PatientRiskFactors | AnyEDvisits\_prior3 | bit | Y |
| dbo.PatientRiskFactors | AnyEDvisits\_prior6 | bit | Y |
| dbo.PatientRiskFactors | anyipsub\_prior12 | bit | Y |
| dbo.PatientRiskFactors | anyipsub\_prior24 | bit | Y |
| dbo.PatientRiskFactors | anymhdisprior12mos | bit | Y |
| dbo.PatientRiskFactors | anymhdisprior1mos | bit | Y |
| dbo.PatientRiskFactors | anymhdisprior24mos | bit | Y |
| dbo.PatientRiskFactors | anymhdisprior3mos | bit | Y |
| dbo.PatientRiskFactors | anymhdisprior6mos | bit | Y |
| dbo.PatientRiskFactors | anymhdx12 | bit | Y |
| dbo.PatientRiskFactors | anymhdx24 | bit | Y |
| dbo.PatientRiskFactors | anymhtx12 | bit | Y |
| dbo.PatientRiskFactors | anymhtx24 | bit | Y |
| dbo.PatientRiskFactors | anypain12 | bit | Y |
| dbo.PatientRiskFactors | anypain24 | bit | Y |
| dbo.PatientRiskFactors | anyresbed\_prior12 | bit | Y |
| dbo.PatientRiskFactors | anyresbed\_prior24 | bit | Y |
| dbo.PatientRiskFactors | anyressub\_prior12 | bit | Y |
| dbo.PatientRiskFactors | anyressub\_prior24 | bit | Y |
| dbo.PatientRiskFactors | AnyUCvisits\_prior1 | bit | Y |
| dbo.PatientRiskFactors | AnyUCvisits\_prior12 | bit | Y |
| dbo.PatientRiskFactors | AnyUCvisits\_prior18 | bit | Y |
| dbo.PatientRiskFactors | AnyUCvisits\_prior2 | bit | Y |
| dbo.PatientRiskFactors | AnyUCvisits\_prior24 | bit | Y |
| dbo.PatientRiskFactors | AnyUCvisits\_prior3 | bit | Y |
| dbo.PatientRiskFactors | AnyUCvisits\_prior6 | bit | Y |
| dbo.PatientRiskFactors | anyusein12moprior | bit | Y |
| dbo.PatientRiskFactors | anyusein18moprior | bit | Y |
| dbo.PatientRiskFactors | anyusein2moprior | bit | Y |
| dbo.PatientRiskFactors | anyusein3moprior | bit | Y |
| dbo.PatientRiskFactors | anyusein6moprior | bit | Y |
| dbo.PatientRiskFactors | apnea12 | bit | Y |
| dbo.PatientRiskFactors | apnea24 | bit | Y |
| dbo.PatientRiskFactors | arb12 | bit | Y |
| dbo.PatientRiskFactors | arb24 | bit | Y |
| dbo.PatientRiskFactors | arth12 | bit | Y |
| dbo.PatientRiskFactors | arth24 | bit | Y |
| dbo.PatientRiskFactors | attempt1 | smallint | Y |
| dbo.PatientRiskFactors | attempt12 | smallint | Y |
| dbo.PatientRiskFactors | attempt18 | smallint | Y |
| dbo.PatientRiskFactors | attempt2 | smallint | Y |
| dbo.PatientRiskFactors | attempt24 | smallint | Y |
| dbo.PatientRiskFactors | attempt3 | smallint | Y |
| dbo.PatientRiskFactors | attempt6 | smallint | Y |
| dbo.PatientRiskFactors | auto12 | bit | Y |
| dbo.PatientRiskFactors | auto24 | bit | Y |
| dbo.PatientRiskFactors | backpain12 | bit | Y |
| dbo.PatientRiskFactors | backpain24 | bit | Y |
| dbo.PatientRiskFactors | bipoli12 | bit | Y |
| dbo.PatientRiskFactors | bipoli24 | bit | Y |
| dbo.PatientRiskFactors | bipolii12 | bit | Y |
| dbo.PatientRiskFactors | bipolii24 | bit | Y |
| dbo.PatientRiskFactors | ca\_head12 | bit | Y |
| dbo.PatientRiskFactors | ca\_head24 | bit | Y |
| dbo.PatientRiskFactors | ca\_prost12 | bit | Y |
| dbo.PatientRiskFactors | ca\_prost24 | bit | Y |
| dbo.PatientRiskFactors | ca12 | bit | Y |
| dbo.PatientRiskFactors | ca24 | bit | Y |
| dbo.PatientRiskFactors | cad12 | bit | Y |
| dbo.PatientRiskFactors | cad24 | bit | Y |
| dbo.PatientRiskFactors | cess12 | bit | Y |
| dbo.PatientRiskFactors | cess24 | bit | Y |
| dbo.PatientRiskFactors | change\_Sq | int | Y |
| dbo.PatientRiskFactors | change\_subtract | smallint | Y |
| dbo.PatientRiskFactors | chronic12 | bit | Y |
| dbo.PatientRiskFactors | chronic24 | bit | Y |
| dbo.PatientRiskFactors | clonazepam12 | bit | Y |
| dbo.PatientRiskFactors | clonazepam24 | bit | Y |
| dbo.PatientRiskFactors | conc12 | bit | Y |
| dbo.PatientRiskFactors | conc24 | bit | Y |
| dbo.PatientRiskFactors | copd12 | bit | Y |
| dbo.PatientRiskFactors | copd24 | bit | Y |
| dbo.PatientRiskFactors | CumDaysUsein12MoPrior | smallint | Y |
| dbo.PatientRiskFactors | CumDaysUsein18MoPrior | smallint | Y |
| dbo.PatientRiskFactors | CumDaysUsein1MoPrior | smallint | Y |
| dbo.PatientRiskFactors | CumDaysUsein24MoPrior | smallint | Y |
| dbo.PatientRiskFactors | CumDaysUsein2MoPrior | smallint | Y |
| dbo.PatientRiskFactors | CumDaysUsein3MoPrior | smallint | Y |
| dbo.PatientRiskFactors | CumDaysUsein6MoPrior | smallint | Y |
| dbo.PatientRiskFactors | cva12 | bit | Y |
| dbo.PatientRiskFactors | cva24 | bit | Y |
| dbo.PatientRiskFactors | cvd12 | bit | Y |
| dbo.PatientRiskFactors | cvd24 | bit | Y |
| dbo.PatientRiskFactors | DaysUsein10MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein11MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein12MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein13MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein14MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein15MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein16MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein17MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein18MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein19MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein1MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein20MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein21MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein22MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein23MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein24MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein2MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein3MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein4MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein5MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein6MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein7MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein8MoPrior | smallint | Y |
| dbo.PatientRiskFactors | DaysUsein9MoPrior | smallint | Y |
| dbo.PatientRiskFactors | dementia12 | bit | Y |
| dbo.PatientRiskFactors | dementia24 | bit | Y |
| dbo.PatientRiskFactors | depr12 | bit | Y |
| dbo.PatientRiskFactors | depr24 | bit | Y |
| dbo.PatientRiskFactors | dt12 | bit | Y |
| dbo.PatientRiskFactors | dt24 | bit | Y |
| dbo.PatientRiskFactors | dysthymia12 | bit | Y |
| dbo.PatientRiskFactors | dysthymia24 | bit | Y |
| dbo.PatientRiskFactors | EDvisits\_prior1 | smallint | Y |
| dbo.PatientRiskFactors | EDvisits\_prior12 | smallint | Y |
| dbo.PatientRiskFactors | EDvisits\_prior18 | smallint | Y |
| dbo.PatientRiskFactors | EDvisits\_prior2 | smallint | Y |
| dbo.PatientRiskFactors | EDvisits\_prior24 | smallint | Y |
| dbo.PatientRiskFactors | EDvisits\_prior3 | smallint | Y |
| dbo.PatientRiskFactors | EDvisits\_prior6 | smallint | Y |
| dbo.PatientRiskFactors | ep12 | bit | Y |
| dbo.PatientRiskFactors | ep24 | bit | Y |
| dbo.PatientRiskFactors | fib12 | bit | Y |
| dbo.PatientRiskFactors | fib24 | bit | Y |
| dbo.PatientRiskFactors | FirstUse1Yr | bit | Y |
| dbo.PatientRiskFactors | FirstUse2Yr | bit | Y |
| dbo.PatientRiskFactors | FirstUse3Yr | bit | Y |
| dbo.PatientRiskFactors | FirstUse4Yr | bit | Y |
| dbo.PatientRiskFactors | gu12 | bit | Y |
| dbo.PatientRiskFactors | gu24 | bit | Y |
| dbo.PatientRiskFactors | ha24 | bit | Y |
| dbo.PatientRiskFactors | hc12 | bit | Y |
| dbo.PatientRiskFactors | hear12 | bit | Y |
| dbo.PatientRiskFactors | hear24 | bit | Y |
| dbo.PatientRiskFactors | hemi12 | bit | Y |
| dbo.PatientRiskFactors | hemi24 | bit | Y |
| dbo.PatientRiskFactors | hiv24 | bit | Y |
| dbo.PatientRiskFactors | homeless12 | bit | Y |
| dbo.PatientRiskFactors | homeless24 | bit | Y |
| dbo.PatientRiskFactors | hyp12 | bit | Y |
| dbo.PatientRiskFactors | hyp24 | bit | Y |
| dbo.PatientRiskFactors | IPDaysUsein10MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein11MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein12MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein13MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein14MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein15MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein16MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein17MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein18MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein19MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein1MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein20MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein21MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein22MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein23MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein24MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein2MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein3MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein4MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein5MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein6MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein7MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein8MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPDaysUsein9MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein10MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein11MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein12MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein13MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein14MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein15MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein16MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein17MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein18MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein19MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein1MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein20MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein21MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein22MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein23MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein24MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein2MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein3MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein4MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein5MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein6MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein7MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein8MoPrior | smallint | Y |
| dbo.PatientRiskFactors | IPMHDaysUsein9MoPrior | smallint | Y |
| dbo.PatientRiskFactors | lagca12 | bit | Y |
| dbo.PatientRiskFactors | lagca24 | bit | Y |
| dbo.PatientRiskFactors | lorazepam12 | bit | Y |
| dbo.PatientRiskFactors | lorazepam24 | bit | Y |
| dbo.PatientRiskFactors | mig12 | bit | Y |
| dbo.PatientRiskFactors | mig24 | bit | Y |
| dbo.PatientRiskFactors | mirtazepam12 | bit | Y |
| dbo.PatientRiskFactors | mirtazepam24 | bit | Y |
| dbo.PatientRiskFactors | moodst12 | bit | Y |
| dbo.PatientRiskFactors | moodst24 | bit | Y |
| dbo.PatientRiskFactors | ms12 | bit | Y |
| dbo.PatientRiskFactors | ms24 | bit | Y |
| dbo.PatientRiskFactors | mst | char(1) | Y |
| dbo.PatientRiskFactors | neuro12 | bit | Y |
| dbo.PatientRiskFactors | neuro24 | bit | Y |
| dbo.PatientRiskFactors | OPDaysUsein10MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein11MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein12MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein13MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein14MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein15MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein16MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein17MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein18MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein19MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein1MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein20MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein21MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein22MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein23MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein24MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein2MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein3MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein4MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein5MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein6MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein7MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein8MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPDaysUsein9MoPrior | smallint | Y |
| dbo.PatientRiskFactors | opioid12 | bit | Y |
| dbo.PatientRiskFactors | opioid24 | bit | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein10MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein11MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein12MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein13MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein14MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein15MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein16MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein17MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein18MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein19MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein1MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein20MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein21MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein22MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein23MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein24MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein2MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein3MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein4MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein5MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein6MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein7MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein8MoPrior | smallint | Y |
| dbo.PatientRiskFactors | OPMHDaysUsein9MoPrior | smallint | Y |
| dbo.PatientRiskFactors | orh | char(1) | Y |
| dbo.PatientRiskFactors | OthAnxDis12 | bit | Y |
| dbo.PatientRiskFactors | othanxdis24 | bit | Y |
| dbo.PatientRiskFactors | OthPsych12 | bit | Y |
| dbo.PatientRiskFactors | othpsych24 | bit | Y |
| dbo.PatientRiskFactors | par12 | bit | Y |
| dbo.PatientRiskFactors | par24 | bit | Y |
| dbo.PatientRiskFactors | Persond12 | bit | Y |
| dbo.PatientRiskFactors | persond24 | bit | Y |
| dbo.PatientRiskFactors | psy12 | bit | Y |
| dbo.PatientRiskFactors | psy24 | bit | Y |
| dbo.PatientRiskFactors | ptsd12 | bit | Y |
| dbo.PatientRiskFactors | ptsd24 | bit | Y |
| dbo.PatientRiskFactors | rd12 | bit | Y |
| dbo.PatientRiskFactors | rd24 | bit | Y |
| dbo.PatientRiskFactors | region | smallint | Y |
| dbo.PatientRiskFactors | relax12 | bit | Y |
| dbo.PatientRiskFactors | relax24 | bit | Y |
| dbo.PatientRiskFactors | schizo12 | bit | Y |
| dbo.PatientRiskFactors | schizo24 | bit | Y |
| dbo.PatientRiskFactors | sci12 | bit | Y |
| dbo.PatientRiskFactors | sci24 | bit | Y |
| dbo.PatientRiskFactors | sedative\_anxiolytic12 | bit | Y |
| dbo.PatientRiskFactors | sedative\_anxiolytic24 | bit | Y |
| dbo.PatientRiskFactors | serviceconnectedgroup | int | Y |
| dbo.PatientRiskFactors | sildenafil12 | bit | Y |
| dbo.PatientRiskFactors | sildenafil24 | bit | Y |
| dbo.PatientRiskFactors | sle12 | bit | Y |
| dbo.PatientRiskFactors | sle24 | bit | Y |
| dbo.PatientRiskFactors | sleep12 | bit | Y |
| dbo.PatientRiskFactors | sleep24 | bit | Y |
| dbo.PatientRiskFactors | statin12 | bit | Y |
| dbo.PatientRiskFactors | statin24 | bit | Y |
| dbo.PatientRiskFactors | sud12 | bit | Y |
| dbo.PatientRiskFactors | sud24 | bit | Y |
| dbo.PatientRiskFactors | symptom24 | bit | Y |
| dbo.PatientRiskFactors | tbi12 | bit | Y |
| dbo.PatientRiskFactors | tbi24 | bit | Y |
| dbo.PatientRiskFactors | tca12 | bit | Y |
| dbo.PatientRiskFactors | tca24 | bit | Y |
| dbo.PatientRiskFactors | thy12 | bit | Y |
| dbo.PatientRiskFactors | thy24 | bit | Y |
| dbo.PatientRiskFactors | tobacco12 | bit | Y |
| dbo.PatientRiskFactors | tobacco24 | bit | Y |
| dbo.PatientRiskFactors | topical12 | bit | Y |
| dbo.PatientRiskFactors | topical24 | bit | Y |
| dbo.PatientRiskFactors | trazodone12 | bit | Y |
| dbo.PatientRiskFactors | trazodone24 | bit | Y |
| dbo.PatientRiskFactors | UCvisits\_prior1 | smallint | Y |
| dbo.PatientRiskFactors | UCvisits\_prior12 | smallint | Y |
| dbo.PatientRiskFactors | UCvisits\_prior18 | smallint | Y |
| dbo.PatientRiskFactors | UCvisits\_prior2 | smallint | Y |
| dbo.PatientRiskFactors | UCvisits\_prior24 | smallint | Y |
| dbo.PatientRiskFactors | UCvisits\_prior3 | smallint | Y |
| dbo.PatientRiskFactors | UCvisits\_prior6 | smallint | Y |
| dbo.PatientRiskFactors | vet12 | bit | Y |
| dbo.PatientRiskFactors | vet24 | bit | Y |
| dbo.PatientRiskFactors | vision12 | bit | Y |
| dbo.PatientRiskFactors | vision24 | bit | Y |
| dbo.PatientRiskFactors | zolpidem12 | bit | Y |
| dbo.PatientRiskFactors | zolpidem24 | bit | Y |
| dbo.PatientRiskFactors | dm12 | bit | Y |
| dbo.PatientRiskFactors | dm24 | bit | Y |
| dbo.PatientRiskFactors | ra12 | bit | Y |
| dbo.PatientRiskFactors | ra24 | bit | Y |
| dbo.PatientStation | ReachID | int | N |
| dbo.PatientStation | PatientSID | int | Y |
| dbo.PatientStation | sta3N | smallint | N |
| dbo.PatientStation | PatientIEN | varchar(50) | Y |
| dbo.PatientStation | Active | bit | Y |
| dbo.Ref\_ClinicalDecisionSupportGuideline | CDSG\_ID | int | N |
| dbo.Ref\_ClinicalDecisionSupportGuideline | Features | text | Y |
| dbo.Ref\_ClinicalDecisionSupportGuideline | Action | text | Y |
| dbo.Ref\_ClinicalDecisionSupportGuideline | GuidelineType | varchar(50) | Y |
| dbo.Ref\_ClinicalDecisionSupportGuideline | RiskLevel | smallint | Y |
| dbo.Ref\_GuidelineRiskFactors | RiskFactorCode | varchar(10) | Y |
| dbo.Ref\_GuidelineRiskFactors | RiskFactorName | varchar(50) | Y |
| dbo.Ref\_GuidelineRiskFactors | RiskfactorDescription | varchar(255) | Y |
| dbo.Ref\_GuidelineRiskFactors | GuidelineURL | varchar(255) | Y |
| dbo.Ref\_GuidelineRiskFactors | ToolkitURL | varchar(50) | Y |
| dbo.Ref\_MaritalStatus | MaritalStatusID | smallint | N |
| dbo.Ref\_MaritalStatus | MaritalStatusDesc | varchar(20) | Y |
| dbo.Ref\_MilitaryBranch | BranchID | smallint | N |
| dbo.Ref\_MilitaryBranch | BranchDesc | varchar(50) | Y |
| dbo.Ref\_MilitaryEra | EraID | smallint | N |
| dbo.Ref\_MilitaryEra | EraDesc | varchar(50) | N |
| dbo.Ref\_OutreachStatus | OutReachStatusID | smallint | N |
| dbo.Ref\_OutreachStatus | StatusDesc | varchar(50) | Y |
| dbo.Ref\_Race | RaceID | smallint | N |
| dbo.Ref\_Race | RaceDesc | varchar(50) | Y |
| dbo.Ref\_Region | RegionID | int | N |
| dbo.Ref\_Region | RegionName | varchar(50) | Y |
| dbo.Ref\_RiskLevel | RiskLevelID | smallint | N |
| dbo.Ref\_RiskLevel | RiskLevelDesc | varchar(20) | Y |
| dbo.Ref\_RiskLevel | RiskLevelFullDesc | varchar(255) | Y |
| dbo.Ref\_RiskLevel | Threshold | decimal(11,10) | Y |
| dbo.Ref\_State | StateAbbr | char(2) | N |
| dbo.Ref\_State | StateName | varchar(50) | Y |
| dbo.Ref\_State | RegionID | int | Y |
| dbo.Ref\_VAMC | STA3N | smallint | N |
| dbo.Ref\_VAMC | VISN | smallint | Y |
| dbo.Ref\_VAMC | VAMC\_Name | varchar(100) | Y |
| dbo.Ref\_VAMC | StateAbbr | char(2) | Y |
| dbo.Ref\_VISN | VISN | int | N |
| dbo.Ref\_VISN | NetworkName | varchar(100) | Y |
| dbo.Ref\_VISN | RegionServed | varchar(500) | Y |
| dbo.Ref\_VISN | RegionID | int | Y |
| dbo.Ref\_VistACancelNoShowCode | CancelNoShowCodeID | varchar(5) | N |
| dbo.Ref\_VistACancelNoShowCode | CancelNoShowCodeDesc | varchar(100) | Y |
| dbo.RiskFactorCoefficient | RiskFactor | varchar(100) | N |
| dbo.RiskFactorCoefficient | Coefficient | decimal(20,19) | Y |
| dbo.RiskScoreHistory | ReachID | int | N |
| dbo.RiskScoreHistory | ScoreDate | date | N |
| dbo.RiskScoreHistory | RiskScore | decimal(11,10) | Y |
| dbo.RiskThresholdHistory | LevelID | smallint | Y |
| dbo.RiskThresholdHistory | BeginDate | date | Y |
| dbo.RiskThresholdHistory | EndDate | date | Y |
| dbo.RiskThresholdHistory | Threshold | decimal(11,10) | Y |
| prsystem.Direct\_Message | MsgID | int | N |
| prsystem.Direct\_Message | MsgType | smallint | N |
| prsystem.Direct\_Message | DateSent | date | N |
| prsystem.Direct\_Message | MsgStatus | smallint | N |
| prsystem.Direct\_Message | Resend | smallint | Y |
| prsystem.Direct\_Message | STA3N | smallint | N |
| prsystem.Direct\_NotifyQueue | NotifyID | int | N |
| prsystem.Direct\_NotifyQueue | ReachID | int | N |
| prsystem.Direct\_NotifyQueue | NotifyDate | date | N |
| prsystem.Direct\_NotifyQueue | MsgType | smallint | N |
| prsystem.Direct\_NotifyQueue | MsgStatus | smallint | N |
| prsystem.Direct\_NotifyQueue | MsgID | int | Y |
| prsystem.Direct\_NotifyQueue | RiskLevel | smallint | N |
| prsystem.Direct\_NotifyQueue | sta3n | smallint | Y |
| prsystem.Direct\_Properties | PropType | smallint | N |
| prsystem.Direct\_Properties | PropName | varchar(100) | N |
| prsystem.Direct\_Properties | PropValue | varchar(200) | N |
| prsystem.Direct\_Station | STA3N | smallint | N |
| prsystem.Direct\_Station | RecipientType | smallint | N |
| prsystem.Direct\_Station | RecipientDirectAddress | varchar(75) | N |
| prsystem.Direct\_Station | UserID | int | Y |
| prsystem.Direct\_Station | ExternalAddress | smallint | Y |
| prsystem.Preference | PrefID | int | N |
| prsystem.Preference | PrefCode | varchar(10) | N |
| prsystem.Preference | PrefDesc | varchar(255) | Y |
| prsystem.Ref\_MsgStatus | MsgStatus | smallint | N |
| prsystem.Ref\_MsgStatus | MsgStatusName | varchar(20) | N |
| prsystem.Ref\_MsgStatus | MsgStatusDesc | varchar(100) | N |
| prsystem.Ref\_MsgType | MsgType | smallint | N |
| prsystem.Ref\_MsgType | MsgTypeName | varchar(20) | N |
| prsystem.Ref\_MsgType | MsgTypeDesc | varchar(100) | N |
| prsystem.Ref\_MsgType | MsgTemplate | varchar(1000) | N |
| prsystem.Ref\_PropType | PropType | smallint | N |
| prsystem.Ref\_PropType | PropTypeName | varchar(20) | N |
| prsystem.Ref\_PropType | PropTypeDesc | varchar(100) | N |
| prsystem.Ref\_RecipientType | RecipientType | smallint | N |
| prsystem.Ref\_RecipientType | RecipientTypeName | varchar(20) | N |
| prsystem.UserAccessLog | ID | int | N |
| prsystem.UserAccessLog | UserID | int | N |
| prsystem.UserAccessLog | Action | nvarchar | N |
| prsystem.UserAccessLog | CreatedDateTime | datetime | N |
| prsystem.UserAccessLog | MachineIP | nvarchar | Y |
| prsystem.UserDashboard | DashboardID | varchar(50) | N |
| prsystem.UserDashboard | DashboardData | varchar(max) | Y |
| prsystem.UserPreference | UserID | int | N |
| prsystem.UserPreference | PrefID | int | N |
| prsystem.UserRole | RoleID | int | N |
| prsystem.UserRole | RoleCode | varchar(10) | N |
| prsystem.UserRole | RoleDesc | varchar(255) | Y |
| prsystem.UserRole | Individual\_View\_Access | bit | Y |
| prsystem.UserRole | Facility\_View\_Access | bit | Y |
| prsystem.UserRole | VISN\_State\_Reg\_View\_Access | bit | Y |
| prsystem.UserRole | Analytics\_Reporting\_Access | bit | Y |
| prsystem.UserRole | System\_Admin\_Access | bit | Y |
| prsystem.Users | UserID | int | N |
| prsystem.Users | UserName | varchar(50) | N |
| prsystem.Users | UserRole | int | Y |
| prsystem.Users | UserStateLocation | char(2) | Y |
| prsystem.Users | FirstName | varchar(50) | Y |
| prsystem.Users | LastName | varchar(50) | Y |
| prsystem.Users | UserHomeFacility | smallint | Y |
| prsystem.Users | UserDomain | varchar(50) | Y |
| prsystem.Users | isActive | bit | N |
| prsystem.Users | UserDashboardID | varchar(50) | Y |
| prsystem.Users | ActiveSessionCounter | int | N |
| prsystem.Users | LastLoggedinDateTime | datetime | Y |
| prsystem.Users | CreatedDateTime | datetime | N |
| prsystem.Users | LoginAttemptCount | int | N |
| prsystem.Users | IsAccountLocked | int | N |
| prsystem.Users | LastLoginAttemptDateTime | datetime | Y |

### Veteran De-Duping Process

The Reach data model will contain one master list of all Veterans imported into the IRDS system. As each Veteran is imported into the database through one of the data sources, a record will be created for that Veteran in the Veteran table and a unique ID will be assigned. It is possible that an Individual might be imported into the system through multiple sources. When this occurs, the multiple Veteran records created for the individual will be merged and all case level data will be linked to that one merged Veteran record. To achieve this, a de-duping process will be run after each data import is run.

This de-duping process will be contained in an SSIS package, which does the following:

1. Reviews the Veteran table for possible duplicates
2. Duplicate groups are determined (2 or more records that could be duplicates)
3. For each duplicate group a text file is created containing the pertinent demographic information (Name, SSN, DOB, Gender) for all records in that group
4. A python program processes these files and evaluates all the records in a group via a record de-duping algorithm
5. The python program returns the results to the SSIS process via another set of text files
6. The results from these files are placed in a temporary SQL table and the Veteran table is reconciled accordingly, by merging any records that were determined to be duplicates



**Figure 4: Veteran De-duping Overview**

### Stored Procedures

dbo.sp\_AssessPatients (no inputs): The procedure checks which patients Assessments scores are not up to date based on the ScoringStatus fields, are re-scores any Patient that is not current by applying coefficients from the RiskFactorCoefficients table against the associated indicator values from the PatientRiskFactor table.

Dbo.sp\_SaveOutReachStatus(inputs: ReachID, Status, User): Stored procedure to be called by the dashboard. When a user selects a value from the Outreach Status dropdown in the dashboard, the JavaScript code will pass the procedure the User ID, the Reach ID of the Patient being reviewed and the value selected by the user. The procedure will update the OutreachStatus field for that Patient in the Patient table and add an entry into the OutreachStatusHistory table. The whole process is managed within a SQL transaction.

### Performance Views

To improve dashboard performance, SQL views will be created. The Dashboard APIs will call these views directly from the Node JS code. The views will contain the select statement that would otherwise be hardcoded/parameterized in the Node JS Code. The Current inventory of performance view is:

vw\_PatientRoster – Queries the patient demographic data. The dashboard can filter this view by VAMC.

## Denormalization

To be determined as technical requirements are gathered.

## Performance Improvement

To be determined as technical requirements are gathered.

## Storage

To be determined as technical requirements are gathered.

## Recovery

To be determined when location where system is housed is identified.

# Database Interfaces

## Suicide Data Repository (SDR)

**Table 6: SDR Interface Details**

|  |  |
| --- | --- |
| Interface | Details |
| Purpose | Data Repository for VA Suicide and Mortality data which will be imported into the Reach database |
| Characteristics | The SDR system runs on a Windows 2008 R2 machine with a SQL Server data store |
| Interface Architecture | Import solution will developed using SSIS |
| API and Error Conditions | Data will be imported using T-SQL to pull data directly from the SDR tables to the tables in the IRDS staging area |
| Security | A SQL connection will be made to the SDR server using Windows Authentication |

## Perceptive Reach Dashboard

**Table 7: Perceptive Reach Interface Details**

|  |  |
| --- | --- |
| Interface | Details |
| Purpose | IRDS users will access data via the dashboard which manages their access and presentation of that data |
| Characteristics | The Dashboard can be run on a compliant web browser |
| Interface Architecture | Client Side is developed in Angular JS, the Server side is developed with Node JS which leverages Express JS |
| API and Error Conditions | The Browser will send requests to Server, which will query the database and return the query results to the browser in JSON format |
| Security | Users login to the dashboard. The dashboard queries settings for that user stored in the Reach database to manage data access |

NOTE: Details for CDW, EDW, VistA and the Analytics sandbox are still being gathered.

# Data Access

1. System users: System users will be given a Login (Server level) or User (Database level) account in SQL Server and granted the appropriate set of rights. System users will connect to the IRDS databases via a SQL Server connection with Windows Authentication, using a tool such as SQL Server Management Studio (SSMS), R or Knime.
2. Dashboard users: Dashboard users will connect to the Reach database indirectly by logging in via the Perceptive Reach dashboard. The dashboard will manage their data access via a set of SQL tables in the ‘System’ schema.

* User Roles – List of roles in the system
* Preferences – List of dashboard preferences
* User – Contains general information about dashboard users, as well as their user role and location
* User Preferences – Contains preferences for each user

When a user logs into the dashboard their presentation of/ and access to data is dictated by the user role and preferences.

## Role Definitions (System)

**Table 8: System Role Definitions**

|  |  |  |
| --- | --- | --- |
| Role-name | Account Type | Rights |
| Database Administrator | Login | sys\_admin |
| Tester | Login | db\_creator |
| Analytics | User | db\_datareader, db\_datawriter |
| Ad-Hoc Reporting | User | db\_datareader |

## Role Definitions (Dashboard)

**Table 9: Dashboard Role Definitions**

|  |  |
| --- | --- |
| User name | Description |
| VAMC level | TBD |
| VISN Level | TBD |
| Region Level | TBD |
| National Level | TBD |

## Node JS Data APIs

Data access for dashboard users will be managed by the IRDS system. When the user has a request for data, the browser will make a request to the IRDS server for that data by calling a specific Node JS data API.

Upon startup, the server will have a Node JS program running that constantly listens for these browser requests. Upon receiving a request, the program will run the API, which queries data from the Reach database and returns the query output to the browser to be displayed appropriately in a browser page.



**Figure 5:** **Dashboard Data Request Process**

### API listing

**Table 10: Data APIs**

|  |  |  |  |
| --- | --- | --- | --- |
| **API** | **Description** | **Input Parameters** | **API Data Output** |
| facilitiesByState | Returns data for each VAMC in a given state | State Abbr | VAMC data for facilities within the given state (STA3N, # of Veterans in IRDS tied to each VAMC) |
| facilitiesByStateCount | Returns number of VAMCs for each state | none | A list of states and facility count for that state |
| scoreSummaryByVAMC | Returns a breakdown of risk scores for Veterans in the IRDS system tied to a VAMC | VAMC ID | A count of Veterans for that VAMC who are Extreme/High/Medium/Low risk of suicide based on the score calculated by the IRDS risk model |
| scoreSummaryByVISN | Returns a breakdown of risk scores for Veterans in the IRDS system tied to a VISN | VISN ID | A count of Veterans for that VISN who are Extreme/High/Medium/Low risk of suicide based on the score calculated by the IRDS risk model |
| totalRiskByState | Returns a breakdown of risk factors for Veterans in the IRDS system tied to a State | State Abbr (optional) | A count of Veterans who are positive for risk factors (such as Diagnosed for PTSD, History of drug use) per the state. If the state parameter is not passed, counts are given for all Veterans in the IRDS system |
| totalRiskByVISN | Returns a breakdown of risk factors for Veterans in the IRDS system tied to a VISN | VISN ID | A count of Veterans who are positive for risk factors (such as Diagnosed for PTSD, History of drug use) per the VISN |
| userLogin | Grants user access to the IRDS system | Username, Password | Users Name, Role, VA location |
| VeteranDetails | Returns data for a specific Veteran | Veteran IRDS ID | All demographics and positive risk factors for a given Veteran |
| Veterans ByVAMC | Returns demographic information for Veterans in the IRDS system tied to a VAMC | VAMC ID | Veterans Name, SSN, Phone#, Date Identified as Risk |

# Implementation Considerations

## Large Objects

Free form text data will be stored in fields with a data type of varchar(max). In SQL Server 2012, a field of type varchar(max) will allow up to 8000 characters. Any data elements that require larger capacity will be stored in fields of type Text.

## Partitioning

At this time there is no partitioning planned.

## Error Processing

After each data import is run, a completion report will be created, any errors incurred will be listed in the report. A database administrator will review the report and take the appropriate action which could include troubleshooting and/or data restoration.