Perceptive Reach:

Integrated Reach Database System

(IRDS)

Database Design Specification



Department of Veterans Affairs

November 2014

Version 0.1

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 11/25/2012 | 0.1 | Initial Database Content | Bill Balshem |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 1

1.1. Purpose 1

1.2. Scope, Approach and Methods 1

1.3. System Overview 1

1.4. Acronyms and Abbreviations 2

1.5. Points of Contact 2

1.5.1. Information 2

1.5.2. Coordination 2

1.5.3. Data Owners 3

2. System Overview 3

2.1. System Information 3

2.1.1. Database Management System Configuration 3

2.1.2. Database Software Utilities 4

2.1.3. Support Software 4

2.1.4. Security 4

2.2. Architecture 5

2.2.1. Hardware Architecture 5

2.2.2. Software Architecture 5

2.2.3. Interfaces 5

2.2.4. Data Stores 5

3. Database Design Decisions 5

3.1. Assumptions 6

3.2. Issues 6

3.3. Constraints 6

4. Database Administrative Functions 6

4.1. Responsibility 6

4.2. Naming Conventions 7

4.3. Database Identification 7

4.4. Systems Using the Database 8

4.5. Relationship to Other Databases 8

4.6. Schema Information 8

4.6.1. Description 8

4.6.2. Physical Design 9

4.6.3. Physical Structure 9

4.7. Special Instructions 9

4.8. Standards Deviations 9

4.9. Entity Mapping 9

4.9.1. Mapping rules 10

4.9.2. Entities and Attributes Not Implemented 10

4.9.3. Non-trivial Mapping 10

4.9.4. Additional Objects 10

4.9.5. Key Mappings 10

4.9.6. Other Deviations 11

4.10. Denormalisation 11

4.11. Performance Improvement 11

4.12. Functional Support 11

4.13. Historical Data 11

4.14. Business Rules 12

4.15. Storage 12

4.16. Recovery 12

5. Database Interfaces 12

5.1. Database Interfaces 12

5.2. Operational Implications 12

5.2.1. Data Transfer Requirements 12

5.2.2. Data Formats 13

5.3. Interface [Name] 13

5.4. Dependencies 13

6. Reporting 13

6.1. Reporting Requirements 13

6.2. Design issues 13

7. Data Access 14

7.1. Role Definitions 14

7.2. Users 14

7.3. Table Access Patterns 14

8. Implementation Considerations 15

8.1. Large Objects 15

8.2. Queues 15

8.3. Partitioning 15

9. Non-Functional Design 15

9.1. Security Design 15

9.2. Availability 15

9.3. Scalability 16

9.4. Performance 16

9.5. Error Processing 16

9.6. Backups and Recovery 16

9.7. Archiving 16

# 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.

The Database Design maps the logical data model to the target database management system with consideration to the system’s performance requirements. The Database Design converts logical or conceptual data constructs to physical storage constructs (e.g., tables, files) of the target DBMS.

Use this Database Design Template to define the basis for the [Application] database design. Describe how the database that will support the [Application] Data Model, supported with details of the logical and physical definitions, non-functional issues, database interfaces, and storage requirements. Where possible, provide expected data volumes, functional and non-functional usage of the tables, and performance considerations and requirements.

## Purpose

The purpose of the Database Design is to ensure that every database transaction meets or exceeds its performance requirements. This document takes into account data and transaction volume to produce a schema and environment that will meet necessary performance.

## Scope, Approach and Methods

Describe the scope of this document as it relates to the project. For example:

The Database Design for the [Application] is composed of definitions for database objects derived by mapping entities to tables, attributes to columns, unique identifiers to unique keys and relationships to foreign keys. During design, these definitions may be enhanced to support the functionality described in the functional specifications and defined in the primary and supporting modules of the application’s High-Level Design.

## Acronyms and Abbreviations

Provide a list of the acronyms and abbreviations used in this document and the meaning of each.

## Points of Contact

Identify the points of contact that may be needed for informational purposes.

### Information

Provide a list of the points of organizational contact (POCs) that may be needed by the document user for informational and troubleshooting purposes. Include type of contact, contact name, department, telephone number, and e-mail address (if applicable). Points of contact may include, but are not limited to, helpdesk POC, development/maintenance POC, and operations POC.

| Role | Name | Email | Telephone |
| --- | --- | --- | --- |
| System Architect | Paul Bradley | paul.bradley@us.pwc.com | 401-209-9006 |
| Database Developer | Bill Balshem | william.balshem@us.pwc.com | 703-918-1310 |
| Technical Lead | Robert Snelling | robert.snelling@us.pwc.com | 205-496-5187 |

Table 3: Organizational POC Contact Information

### Coordination

Provide a list of organizations that require coordination between the project and its specific support function (e.g., installation coordination, security, etc.). Include a schedule for coordination activities.

| Organization | POC Name | Email | Telephone |
| --- | --- | --- | --- |
| [Installation] | [Name] | [Email] | [123-345-456] |
| [Development] | [Name] | [Email] | [123-345-456] |
| [Security] | [Name] | [Email] | [123-345-456] |

Table 4: Coordination POC Contact Information

| Phase | Activity | POC | Start Date |
| --- | --- | --- | --- |
| Design | Sign-off document | [Name] | DD/MM/YYYY |
| Development | Develop Database | [Name] | DD/MM/YYYY |
| Testing | Test cycle | [Name] | DD/MM/YYYY |

Table 5: Activity Start Information

### Data Owners

Identify points of contact for those who own or are responsible for data quality, currency, accuracy, etc.

| Type of Data | POC Name | Email | Telephone |
| --- | --- | --- | --- |
|  | Clint Latimer | clint.latimer@va.gov |  |
|  | Rob Bossarte | Rob.bossarte@va.gov |  |
|  |  |  |  |

Table 6: Data Owner POC Information

# System Overview

Provide a brief overview of the system. Ensure that this section is consistent with the high-level design (if it exists).

1. Highlight errors in the High-Level Design document to the Database Designer.

Label each component, so that they may reference consistently across technical documents, diagrams, and spreadsheets when referencing subsystems and components.

The main components of the IRDS system are:

* Reach Database - A SQL Server database on the IRDS server that stores data from the various VHA source imported into the system.
* Analytics Risk Model/Surveillance Model - A program written in R will run periodically to update a list of high factors associated with Veteran suicide. These factors will be persisted in one or more tables in the Reach database. A SQL process run on a regular basis (nightly?, weekly?) to monitor veterans as being high for suicide based on high risk factors as determined by the Analytics Risk model.
* Direct Messaging - As veterans as identified by the surveillance model as being high risk. The VA staff will be notified via direct messaging component of the IRDS system, which leverages the VLER solution adopted by the VA.
* IRDS 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 are imported into the IRDS system via SQL Server Integration Services (SSIS) import solutions. Each data source will have its own SSIS package (.dtsx). In the case of importing data from a VistA installation, one or more RPC calls will be executed from within an SSIS import. The import solution will transform the data and load into the appropriate tables in the Reach database.
2. An R program is run periodically (every year or so) on the production server to update the Risk model. The results of the run are stored in a table in the Reach database.
3. On a regular basis (daily, weekly) a SQL Server process runs that does surveillance against a list of veterans tracked in the Reach database tables against the risk model. The process identifies who are at high risk of attempting suicide.
4. The results of the surveillance run are emailed to the appropriate contacts for those veterans via a secure message that leverages the VA VLER architecture.
5. A system user opens up their dashboard via a compatible web browser and a client side java component connects to a server side java component, which queries the Reach database for both specific and aggregate data regarding high risk veterans at their management level (region, state, VISN, VAMC). The query results are passed to the client browser and populated in the web page.

**Diagram 1 - Business process Diagram**

## System Information

Specify the Database Management System configuration, hardware configuration, database software utilities, and any support software used.

### Database Management System Configuration

Identify the vendor, version and targeted hardware for the database management system. Highlight any restrictions on the initialization and use of the DBMS.

| Vendor | Hardware | Version | Comments |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table : Database Management System Configuration

### Database Software Utilities

Identify any utility software that will be used to support the use or maintenance of the database.

| Vendor | Product | Version | Comments |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table : Database Software Utilities

### Support Software

Identify the support software directly related to the database, including name, version, function, and major operating characteristics.

Examples include software for query language, report writers, storage, database loading, file processing, and data cleansing.

| Product | Version | Purpose |
| --- | --- | --- |
| Microsoft (MS) SQL Server | 2012 | Database platform |
| MS SQL Server Integration Services |  | Data import solution development tool |
|  |  |  |

Table : Support Software

### Security

Discuss any integrity and access controls that apply to database components such as schema, sub-schema, partitions or physical files, records or tables, sets or relations, and data elements.

TBD.

## Architecture

### Hardware Architecture

Provide a brief an overview of the hardware architecture with supporting [flowchart / state / sequence etc] diagrams to illustrate how components are connected. Identify the hardware configurations on which the database will reside.

### Software Architecture

List the components within the subsystem/application. Provide component diagrams to illustrate connections within the application and external systems. Include components, datastores and interfaces within the application as well as interfaces between internal components and external systems.

* Label internal interfaces for reference. Label external interfaces consistently with those used in the high-level design document.
* Indication direction on an interface, i.e. the direction of initiation or the main direction of dataflow.

### Interfaces

Identify interfaces to external systems. Interfaces are described in more detail in the following chapters.

TBD.

### Data Stores

Identify and describe all data stores including databases, file systems and media data stores.

# Database Design Decisions

Discuss the decisions that were made when designing the database for [Application], including problems, alternative solutions, and design assumptions that had to be made. Ensure the analyst team verifies any assumptions made as a result from ambiguities or lack of details.

Divide this section into paragraphs as needed to present database-wide design decisions. Discuss how it will behave, from a user's point of view, in meeting its requirements and other decisions affecting further design of the database.

Examples of design decisions may include:

* Queries or other inputs the database will accept and outputs (displays, reports, messages, responses, etc.) it will produce.
* Database behavior in response to each input or query, including actions, response times and other performance characteristics.
* How databases / data files will appear to the user.
* Type of flexibility to be built into the database for adapting to changing requirements.
* Levels and types of availability, security, privacy, and continuity of operations.
* Database distribution (such as client / server), master database file updates and maintenance, including maintaining consistency, synchronization, enforcing integrity and business rules
* Backup and restoration including distribution strategies, permissible actions, and special considerations for non-standard technologies.
* Decisions on sorting, indexing, synchronization, and consistency including automated disk management, optimizing strategies, storage and size considerations, and population of the database and capture of legacy data

## Assumptions

List any assumptions made due to lack of information, e.g. ambiguous sections in the functional specifications, or made during design, e.g. assumed constraints, assumptions about other systems or where requirements analysis was unclear.

| Ref # | Assumption | Impact |
| --- | --- | --- |
| #1 | Describe the assumption | Discuss its effect on the database design |
| #2 | Describe the assumption | Discuss its effect on the database design |
| #3 | State ‘none’ if appropriate |  |

Table : Assumptions

## Issues

At this point, any outstanding issues should have been converted into design statements or into assumptions as listed above.

## Constraints

Identify any known constraints on the database design. Constraints are factors that may restrict the design/project by scope, resource, platform, language, schedule etc.

| Ref # | Constraint | Impact |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

Table : Constraints

# Database Administrative Functions

## Responsibility

Identify the organizations and personnel responsible for the following database administrative functions: database administrator, system administrator, and security administrator. Describe specific administration skill requirements.

| Role | Name | Responsibility | Email |
| --- | --- | --- | --- |
| Identify role | Identify the person responsible | Identify area of responsibility | Email address |
| Identify role | Identify the person responsible | Identify area of responsibility | Email address |
| Identify role | Identify the person responsible | Identify area of responsibility | Email address |

Table : Database Responsibilities

## Naming Conventions

Identify logical and physical naming standards and conventions.

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

Table : Database Naming Conventions

## Database Identification

Identify the names or labels by which the database will identified. Specify the code name, tag, or label by which each database table or file will be identified.

For example, the following elements provide identification and status information about the database.

| Element | Element Name | Meaning |
| --- | --- | --- |
| db\_name | Database Name | The name the database was given when created. The real name of the database for which information is collected or to which the application is connected. |
| db\_path | Database Path | The full path to the location where the database is stored on the system. |
| db\_location | Database Location | The location of the database in relation to the application. |
| db\_storage\_path | Storage path | Full path of a location that is used by the database for placing automatic storage table spaces |

Table : Database Identification

## Systems Using the Database

Identify the systems that will use the database. Include the full system identification and model, modification, version number, and system code.

| System ID | Model | Version # | System Code |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table : Systems Using the Database

## Relationship to Other Databases

Indicate whether the database will supersede or interface with other databases, and specify which one(s).

| This Database | Supersedes this Database | Interfaces with the Database |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

Table : Relationship to Other Databases

## Schema Information

Describe the overall structure in the schema or other global definition of the database.

The Reach database will contain the following schemas:

* .dbo - The MS SQL Server default schema, will contain data imported from VHA 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 list of ICD codes.
* 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.
* System - Tables used by the IRDS dashboard
* Analytics - Tables that persist the outcome of the latest Risk Model run that determines high risk factors for veterans.

### Description

Describe the schema and each sub-schema of the system including name, file type and name, data description language, access control keys, concurrence locking, data name mapping, overall partition/file limitations and controls, redefinition and access path restrictions and any other limitations or restrictions.

### Data Model

The Reach Database contains the following tables:

Data Tables

The Reach database will store source imported data on 2 levels:

* Veteran Demographics
* Veteran Case Data

For each veteran, a master record will be created for that individual. A record for that be created in the 'Veterans' table and a unique ID will be created for that individual within the reach system 'ReachID'. The table will also contain basic demographic for that individual, such as Name, SSN, DOB, Gender, etc... The table also contains a 'Score' field, which the surveillance piece of this system will populate to for that vet to determine if he or she is at high risk. If another record is imported into the system for that individual from another source, the currently existing Veteran record that exists for the vet will be updated with any additional demographic data the additional source contains.

All other data elements imported for veterans will be stored in case level tables. There will be a one to many relationship from the Veteran table to any case level table. For instance, any data that may be available from the SDR on previous suicide attempts for that individual will stored in a suicide attempt table. For each suicide attempt that is documented for a veteran, a 'SuicideAttempt' record will be created, including a ReachID column that will link the suicide Attempt record back to the veteran.

Reference Tables

List 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.

High Risk Veterans View

The High Risk Veterans view will roll up data from the Veterans List and case data to show all vets that are identified as high risk with their risk score and the factors that contributed to that score.

Risk Factors

The Risk Factors determined by the IRDS Risk Model will be captured in the RiskFactors table(s). This table will be updated any time the Risk model is updated, by running the methodology encapsulated in an R program that exists on the IRDS server.

System Tables

This includes tables created by SSIS data import packages to perform transformations. These tables will be kept after an import run is complete in the case it is necessary to troubleshoot an issue with an import run.

One or more configuration tables will be kept to be used by the dashboard code to track settings and other configuration items.



**Figure 2 - IRDS Reach Database Data Model**

For each veteran, a master record will be created for that individual. A record for that be created in the 'Veterans' table and a unique ID will be created for that individual within the reach system 'ReachID'. The table will also contain basic demographic for that individual, such as Name, SSN, DOB, Gender, etc... The table also contains a 'Score' field, which the surveillance piece of this system will populate to for that vet to determine if he or she is at high risk. If another record is imported into the system for that individual from another source, the currently existing Veteran record that exists for the vet will be updated with any additional demographic data the additional source contains.

All other data elements imported for veterans will be stored in case level tables. There will be a one to many relationship from the Veteran table to any case level table. For instance, any data that may be available from the SDR on previous suicide attempts for that individual will stored in a suicide attempt table. For each suicide attempt that is documented for a veteran, a 'SuicideAttempt' record will be created, including a ReachID column that will link the suicide Attempt record back to the veteran.

### Physical Structure

Provide a diagram illustrating the physical structure (i.e. partitions, files, indexes, pointers) and the logical components of the database.

## Special Instructions

Provide instructions for DBAs, operators and testers who my use the database for testing and operational purposes. For example:

* Specialized criteria for entering data into the database.
* Procedures for submitting data for entry into the database.
* Instructions for generating, modifying, updating, or otherwise using the database.

## Standards Deviations

List any known deviations from corporate standards and recommended guidelines.

## Entity Mapping

Identify the mapping rules and lists tables and columns that either:

* Do not originate from a single entity
* Are not implemented

### Mapping rules

Identify rules for mapping entities to tables, for example:

* Entities are mapped onto tables in a one to one manner
* Attributes are mapped to columns in a one to one manner
* One-to-many relationships are mapped to foreign keys
* Many-to-many relationships are implemented using a keys-only table

### Entities and Attributes Not Implemented

Identify entities and attributes that are not implemented as tables and columns.

| Entity/Attribute | Description | Reason for not implementing |
| --- | --- | --- |
|  |  |  |

Table : Entities and Attributes Not Implemented

### Non-trivial Mapping

List all tables that are not derived from an entity in a one-to-one fashion.

| Table/Column | Mapped from | Purpose | Reason for deviation |
| --- | --- | --- | --- |
|  |  |  |  |

Table : Non-trivial Mapping

### Additional Objects

Lists database objects (tables or columns) that were not derived from an entity but added to the database design for the purpose listed below.

| Table/column | Description | Purpose |
| --- | --- | --- |
|  |  |  |

Table : Additional Objects

### Key Mappings

Identify the tables that have primary keys created from sequences:

| Table | Primary key column | Sequence |
| --- | --- | --- |
|  |  |  |

Table : Key Mappings

### Other Deviations

Identify deviations from a one-to-one mapping of entity and attribute names to table and column names and any foreign key naming deviations.

| Entity/Attribute/Relation | Table/Column/Foreign Key Column | Reason for deviating |
| --- | --- | --- |
|  |  |  |

Table : Other Deviations

## Demoralization

Where appropriate, describe how redundancy is added to the design to improve performance or support specific functionality.

## Performance Improvement

Identify objects that were modified in order to improve performance:

|  |  |  |
| --- | --- | --- |
| Denormalized Table/Column | Source table or entity | Rules and methods for maintaining integrity |
|  |  |  |

Table : Denormalization-related Performance Improvements

## Functional Support

Identify objects that were modified in order to support the proposed functionality of [Application]:

|  |  |  |
| --- | --- | --- |
| Denormalized Table/Column | Source table or entity | Rules and methods for maintaining integrity |
|  |  |  |

Table : Denormalization-related Function Support

## Historical Data

Identify additions made to accommodate data manipulation and to keep/archive/delete historic data.

|  |  |  |
| --- | --- | --- |
| Object | Description | Issues |
|  |  |  |

Table : Historical Data

## Business Rules

Describe business rules modeled in the data model, specified for entities in the data model or in the functional specification that have **NOT** been implemented as table/column constraints/column-defaults.

|  |  |
| --- | --- |
| Business Rule | Implemented |
| Identify business rule | Implemented by using …. |
| Identify business rule | Implemented by using …. |
| Identify business rule | Implemented by using …. |

Table : Business Rules

## Storage

Provide sizing formulas for determining the storage required to support the database. Estimate the internal and peripheral storage requirements.

## Recovery

Describe how data, schemas and support files will be recreated or recovered in the event of a system disaster.

# Database Interfaces

## Database Interfaces

Describe interfaces with other applications including those of other operational capabilities.

## Operational Implications

Describe operational implications of data transfer, including security considerations.

### Data Transfer Requirements

Describe data transfer requirements including content, format, sequence, and conversion issues.

### Data Formats

Describe data formats for the sending and receiving systems, including the data item names, codes, abbreviations, as well as any units of measure/conversion issues.

## Interface [Name]

|  |  |
| --- | --- |
| Interface | Details |
| Purpose | Describe the purpose of the interface |
| Characteristics | Summarize the interface characteristics |
| Interface Architecture | Describe the interface architecture. |
| API and Error Conditions | Describe the API and error conditions. Reference separate module interface specifications for more detailed information. |
| Security | Describe protocols, user authentication, encryption, access control (at the interface entry point). |

Table : Interface Details

## Dependencies

List any dependencies for the [Application] schema, for example, foreign keys across schemas.

|  |  |  |
| --- | --- | --- |
| Table and column in [application] schema | Schema the table/ column refers to | Table |
|  |  |  |

Table : Dependencies

# Reporting

## Reporting Requirements

Describe any reporting requirements.

## Design issues

Describe the design to support reporting and other information requirements.

# Data Access

Discuss which roles are needed to use the database and highlight any significant information related to the physical database implementation, for example, tables subject to high insert or delete activity or with specific archiving rules.

## Role Definitions

Identify the roles defined in the database.

|  |  |
| --- | --- |
| Role-name | Purpose |
|  |  |

Table : Database Role Definitions

## Users

Identify users that will be recognized by the database, including estimates of user volumetrics.

|  |  |
| --- | --- |
| User name | Purpose |
|  |  |

Table : Database Users

## Table Access Patterns

Identify performance-critical functions and their table usage. Where possible, provide volumetric information needed for the physical database design.

|  |  |  |
| --- | --- | --- |
| Function | Peak Frequency | Tables Used |
|  |  |  |

Table : Table Access Patterns

Identify tables that will be classified as one of the following:

|  |  |
| --- | --- |
| Table | Type |
| High-volume read only |  |
| High-volume insert |  |
| High-volume updates |  |

Table : Table Types

# Implementation Considerations

## Large Objects

Describe how large objects will be stored, for example, objects with a maximum size of 50MB will be stored as BLOBS.

## Queues

Describe how queues (i.e. asynchronous messaging techniques) will be used. Specify which functionality the queue implements and the implementing queuing technology (e.g. JMS).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Queue  Name | Table  Name | Queue  Type | Max  Retries | Retry  Delay | Retention  Time | Dependency Tracking | Auto Commit |
|  |  |  |  |  |  |  |  |

Table : Queues Descriptions

## Partitioning

Describe the design and format of each partition/file, including name, type, code, mapping, limitations and controls, access procedures, and mechanisms. Identify the interdependencies of each partition/file in the database.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Partition Table | Index (Y/N) | Partition column | Partition value | Partition Name | Partition size | Comments |
|  |  |  |  |  |  |  |

Table : Partition Descriptions

# Non-Functional Design

Describe the non-functional design elements for the database.

## Security Design

Describe authentication, integrity, and confidentiality requirements that will be supported within the database.

## Availability

Describe the database design subsystem/component availability and resilience requirements.

## Scalability

Describe how the database design supports scalability requirements.

## Performance

Describe how the database has been designed for performance.

## Error Processing

Describe the error processing strategy adopted and how it is supported within the database design.

## Backups and Recovery

Describe the backup and recovery policy to be used.

## Archiving

Describe the archiving policy to be used