PAWSWARE Data Warehouse

# Database Design Document

Version 1.0

03/24/2019

|  |  |
| --- | --- |
| Prepared by: | Luciana Palucci |
| Student number: | 2597128 |
| Course name: | CEBD1250 (Big Data Storage) |

Table of Contents

[1. Overview iii](#_Toc4229595)

[2. Assumptions/Constraints/Risks iv](#_Toc4229596)

[2.1 Assumptions iv](#_Toc4229597)

[2.2 Constraints v](#_Toc4229598)

[2.3 Risks v](#_Toc4229599)

[3. Design Decisions vi](#_Toc4229600)

[3.1 Key Factors Influencing Design vi](#_Toc4229601)

[3.2 Functional Design Decisions ix](#_Toc4229602)

[3.3 Database Management System Decisions x](#_Toc4229603)

[3.4 Performance and Maintenance Design Decisions xi](#_Toc4229604)

[4. Detailed Database Design xii](#_Toc4229605)

[4.1 Performance Monitoring and Database Efficiency xii](#_Toc4229606)

[4.1.1 Operational Implications xii](#_Toc4229607)

[4.1.2 Data Transfer Requirements xii](#_Toc4229608)

[4.1.3 Data Formats xii](#_Toc4229609)

[4.2 Detailed Data Modeling Database Design xiv](#_Toc4229610)

[4.2.1 Conceptual Data Model xv](#_Toc4229611)

[4.2.2 Logical Data Model xvi](#_Toc4229612)

[4.2.3 Physical Data Model xvii](#_Toc4229613)

[Appendix A: Acronyms xix](#_Toc4229614)

[Appendix B: Architecture of Data Warehouse System xx](#_Toc4229615)

[Appendix C: Detailed Database Design: DDL xxi](#_Toc4229616)

List of Tables

Table 1 - Acronyms xix

## Overview

This document describes the database design for a rescue pet adoption data warehouse named PAWSWARE.

Background

This data warehouse (DWH) will contain a registry of pets that are up for adoption from various rescue organizations in the country of Malaysia. The background of the project stems from a Kaggle machine learning competition to determine the adoptability of a pet. The pet source dataset comes from the Kaggle web site and rescue dataset comes from a U.S. web site with rescue names dataset. The rescue dataset was manipulated to be merged with the pet datasets.

The purpose of the data warehouse is to store in a central repository data of pets that are up for adoption, and to share this data through a web application. The goal of a central repository for pet adoption data is to gain knowledge from this central data warehouse and have reporting measurements that reflects the real-life situation.

The data warehouse shall be the source to acquire pet adoption information for analysis, such as:

- Analysis how adoptable is a pet

- How pets are adopted geographically by breed and location

- Locate possible place of high likelihood for a specific pet breed to be adopted

- Analysis of pet adoption trends, in which pets are more likely to be adopted in which neighborhood.

Designs approach

The data architecture for the PAWSARE DWH shall have a data lake built on top of it, where all the raw data sources will be stored into a Hadoop HDFS storage file system. The extraction, transformation, and load (ETL) transactions shall be processed at this level.

Dependancies

The data that is loaded into the DWH is pet characteristics and image files. The pet’s description text attribute is run through Google's Natural Language API to get sentiment attributes in a JSON format. The pet’s image file is run through Google's Vision API to get image properties stored in a JSON format. These collections data will be loaded into database in same JSON format as the source.

Interfaces with other systems

The data warehouse shall interface with a web application at the presentation layer of DWH architecture. It shall also interface at the data staging phase with a Python program which shall run to read in the source data, do some quality control on the data (i.e. the data conforms to the variable constraints in the database), transform, and load to the MySQL database.

## Assumptions/Constraints/Risks

### Assumptions

The MySQL database shall be version 8. The PAWSWARE DWH stores some data in JSON format data type, hence the MySQL version should be above version 5.7.8, where JSON format type is supported.

The PAWSWARE Data Warehouse shall be part of the architecture described in section 3, with the source data being stored in a Hadoop HDFS environment. For the purpose of the project, yet the data is read in from hard disk.

Platform

The platform in which the database resides has been configured and is ready for a database to be deployed on top. The operating system shall be on a Linux Ubuntu. The O.S. is simulated through a Docker container.

Input data

* The raw source data shall be transmitted in a structured format in which may be of any file type (e.g. delimited files, tabular, or Excel).
* The image files shall be transmitted in a picture format file type in which stores computer graphics (e.g. jpeg, gif, bmp, tiff, or png)
* Image and text data put through google API, and then transmitted as JSON files only.
* Source data is stored and available for uploaded in the DWH from the data lake.
* The data being imported into the DWH has been quality controlled or verified at the rescue organization.

Software

The web application software development shall be developed once this data design is approved.

### Constraints

Some constraints on the data warehouse design and its platform are:

The architecture for the data flow from the source to the data warehouse is such that the ETL processing occurs in the Hadoop apache environment.

For purpose of demonstration, running the Python data load program(s) is performed in a PC desktop from Python the

Database Tables

There are database table constraints specified on the physical model, and they specify which attributes are not null, unique, primary key, or foreign key constraints.

Some limitations of the database design are that a source data variable may have invalid values compared to the database columns where it is inserted. This data shall be pre-treated to maintain data integrity with the defined table constraints on foreign key.

Design

* The financial budget for the creation of the data warehouse may be limited.
* Time to develop
* May contain legal limitations on storing personal data.

### Risks

Some risks associated with the design and the building of the PAWSWARE Data Warehouse are:

- The data conversion should not lose any meaning when mapping to the pre-defined values.

- The integration of the source data may take time to fit into the PAWSWARE DWH data variable for loading.

- Much of the computer h/w, s/w, and RDBMS were done on open-source systems, yet there may be costs in building the data warehouse.

- The data is centralized into one depot so there may be some transparency in the way the rescue organization functions (i.e. their data may expose the way they operate, their intent towards the pets that are placed in their care).

## Design Decisions

The following sections describe decisions made that impact the proposed database design at the platform and database management system level for the project.

The data warehouse exists as a layer on top of source data (usually from OLTP databases where the adoption data is entered). The reason for the data warehouse is the fact that operational database (i.e. the source data from rescues) doesn’t lend itself to analytics; hence the data warehouse takes the data from sources and creates a layer that is purposed for analytics. The DWH contains data that can be reconstructed from the source data.

### Key Factors Influencing Design

Some key functional and non-functional requirements that influenced the design of:

The database in which the DWH is built on:

The rescue pet adoption data warehouse was initially designed to store a national registry of data sourced from rescue organizations, and JSON files into a hybrid data warehouse of MySQL and NoSQL Key-value store. The structured data would be stored into MySQL RDBMS and the JSON into a key-value NoSQL database. A DWH system should not exist as a hybrid system, so the content of the JSON files shall be stored into a MySQL database holding JSON format type data (hence, the constraint on the MySQL version).

The data warehouse environment:

The architecture design of Hadoop HDFS as a file system for holding the source data stems from the idea that data warehouses do not store real-time data, nor are they base for a recommendation system. So came about the idea of having a data warehouse repository for pet adoption data and real-time live data for analysis (e.g. machine learning) in HDFS.

With the data warehouse build in this type of architecture, PAWSWARE DWH shall have its batch processing run at the Hadoop level. Here the batch ETL programs will run faster than at the DWH platform with Hadoop utilities handling the processing. This method may also handle the case of having batch processes seem more like a streaming process, which will make the DWH a real-time data warehouse. Refer to appendix B Architecture of Data Warehouse System for diagram.

The functional and non-functional requirements of the data warehouse system shall be:

**Functional Requirements**

|  |  |  |
| --- | --- | --- |
| Functional Requirements |  |  |
| **ID** | **Requirement** |  |
| ***System Requirements*** |  |  |
| REQ01 | The data warehouse shall store structured data |  |
| REQ02 | The data warehouse shall contain pet adoption data from rescue organizations. |  |
| REQ03 | The data warehouse shall have the source data that is specified in the logical and physical data model loaded into it. |  |
| REQ04 | Build a data warehouse (DW) containing all rescue organizations data |  |
| ***User Requirements*** |  |  |
| REQ05 | The DW must interact with web application built to view data and get reports out of the warehouse. |  |
| REQ06 | The file formats transferred to the DW must be in a readable format for loading data into DW. |  |
| ***Business Requirements*** |  |  |
| REQ07 | The DW shall contain historical data. |  |
| REQ08 | The DWH shall contain data that maps where and when pets are often abandoned (identifies areas as abandonment hotspots). |  |
| REQ09 | The DWH shall contain pet demographics, and image data. |  |
| REQ10 | The DWH shall contain data to verify if pet is right fit (that way once adopted does not return to shelter). |  |
| REQ11 | The DWH shall contain API output data in JSON format. |  |
| ***Standards Requirements*** |  |  |
| REQ12 | The data transferred to the DW shall contain the fields identified into the Logical Model of the warehouse in the format specified within. |  |

**Non-Functional Requirements**

|  |  |  |
| --- | --- | --- |
| Non-Functional Requirements |  |  |
| **ID** | **Requirement** |  |
| ***Performance Requirements*** |  |  |
| REQ01 | The DWH shall be available for user access. |  |
| REQ02 | The DWH shall have a good response time |  |
| ***Interoperability Requirements*** |  |  |
| REQ03 | The DWH shall interact with web application built to view data and get reports out of the warehouse. |  |
| REQ04 | The file formats transferred to the DW must be in a readable format for loading data into DW. |  |
| ***Space Requirements*** |  |  |
| REQ05 | DW must interface with importing other databases to import the data |  |
| ***Platform Requirements*** |  |  |
| REQ06 | The DW will be hosted on a LINUX platform. |  |

### Functional Design Decisions

The main repository for the pet adoption data is a relational database on a MySQL version8 as the database management system and shall be hosted on a Linux virtual machine. The database is designed as a normalized Snowflake Schema dimensional model. The conceptual, logical, and physical data model is described in the detail section of the report.

How the database will behave in meeting its requirements from a user's point of view:

* The database shall support the storage of the data variables identified in the Entity-Relationship Diagram that are pertinent for report summaries, and predictive analysis of pet adoption data.
* The database shall interface with a Python conversion and load program.

The database design architecture was chosen from analysis of what is required from source data, what the data is used for, and in what formats the source data would be imported as.

Since the data warehouse shall contain the pet adoption data from many different rescue organizations in the country:

1. The input source data structure shall look like:

* The source data shall be structured data from all rescue agencies, with data entities transferred to the file repository

-variables: pet id with pet demographics description, its rescue ID, the rescue location, an image file, sentiment and image analysis in JSON.

1. The data shall support at the presentation layer:

* Review and analysis of the data, for data or image sharing
* User recommendation based on web activity.
* A reporting and predictive analysis.

List of interfaces with other systems:

|  |  |
| --- | --- |
| **System** |  |
| Docker container | Host the database. |
| Database Schema | SQL DDL programs to create database schema |
| Data warehouse | Python program to upload source data |

### Database Management System Decisions

The database is built on MySQL and the schema is named PAWSWARE. It behaves with the schema-on-write approach, where the schema is first defined, then load the data, and the data is read from the DWH in the same schema pattern. This database design property does not lend to easily database schema changes.

Change control: Changes to the database requirements shall go through a process of change control, where the required modifications would be evaluated for the database and the programs that support the ETL.

### Performance and Maintenance Design Decisions

Design and Architecture

The PAWSWARE database shall be hosted on a Linux Ubuntu server with a MySQL relational database management system (RDMS). Some features of MySQL is that it supports SQL, partitions tables for better performance, support for virtual or container environments, and is the best choice for storing structured data. The data sources loaded into the MySQL database have a fixed consistent variable collected for the purpose of rescue and pet adoption data.

Data Conversion

The data conversion plan is to read in the source data with a Python program and convert or extract the data variables of interest for uploading to the PAWSWARE Data Warehouse. The data variable with lookup value codes (e.g. furlength, vaccinated, etc) shall be entered (or transformed) as a valid value that is found in a parent key tables or lookup vocabulary terms (i.e. the lookup, breed, color, etc. tables). When data cannot be directly mapped to a pre-defined lookup code, a business rule shall be defined for harmonizing the data. For example, FURLENGTH with values of i) very short, and short, and 2) Unknown can be mapped to 1) short and 2) Not Specified.

Service Level Agreements

All rescue organizations that agree to have their data part of the PAWSWARE Data Warehouse, shall have their data integrated into the PAWSWARE DW. This may entail to transform data from source data values to the DWH required fields and vocabulary terms described in section 4.2.2. All historical data shall be maintained in the PAWSWARE DW and accessible to all members that are given access to the PAWSWARE Data Warehouse.

Concurrency Control

The database system chosen for the data warehouse has the ACID property in maintaining control over concurrent transactions (the source of database inconsistencies). The MySQL engine handles the ACID properties:

-atomic: a transaction must complete or have recovery procedures, and not be

-consistency: another transaction cannot perform on same record (a lock on row).

-isolation: a transaction occurs in itself without knowing of other in queue.

-durability: a successful transaction is permanent.

System Availability

The PAWSWARE database must be available from Monday to Friday at 17:00 EST so that the applications interfacing with the database is available for use (ETL, reporting, and data analysis).

The PAWSWARE Data Warehouse Maintenance shall be performed during the weekend starting Friday 17:00 to Sunday 22:00 EST, where the database and the applications interfacing with it will not be accessible. During this period, the a data warehouse version for reading may be available.

## Detailed Database Design

### Performance Monitoring and Database Efficiency

#### Operational Implications

The database shall have data transfer to the DWH database on a scheduled basis so that a database fresh may be done the same scheduled basis. The refresh time span shall be in a 1-hour window.

#### Data Transfer Requirements

The data transfer requirements from source data are formatted files containing the variables in the logical model. Any conversion to the source data shall be done at the ETL phase in the Hadoop layer.

The transferred data location shall be stored in:

* For the fixed format data: shall be stored to folder ‘data’
* Google API outputs; shall be stored in subfolder ‘sentiment’,
* Pet image files: shall be stored in subfolder ‘images’.

#### Data Formats

The data formats from the receiving systems of the rescue organization is:

|  |  |
| --- | --- |
| **Format** | **Information** |
| Custom delimited | The source data for pet description variables described in the logical model shall be transferred in an Excel style format file with comma (‘,’) delimiter. |
| JSON | The source data from the Google’s vision API and Google’s natural language API shall be transferred in a JSON format. |
| Images file | The source file for pet images shall be in jpg or other picture formats. |

Conversion Fields

The variables below shall be transferred in the code values instead of text values.

|  |  |
| --- | --- |
| Variable | Mapped Codes |
| ptype (pet type): | 1= dog  2=cat |
| gender | 1=Male  2=Female  3=Mixed |
| maturitysize | Small  Medium  Large  Extra Large  Not Specified |
| furlength | 1=Short  2=Medium  3=Long  4=Not Specified |
| vaccinated  dewormed  sterilized | 1=Yes  2=No  3=Not Sure |
| health | 1=Healthy  2=Minor Injury  3=Serious Injury  4=Not Specified |
|  |  |

### Detailed Data Modeling Database Design

The detailed design for DWH system and the data it shall store is describes in the following section, with the conceptual, logical, and physical data models of the required data fields.

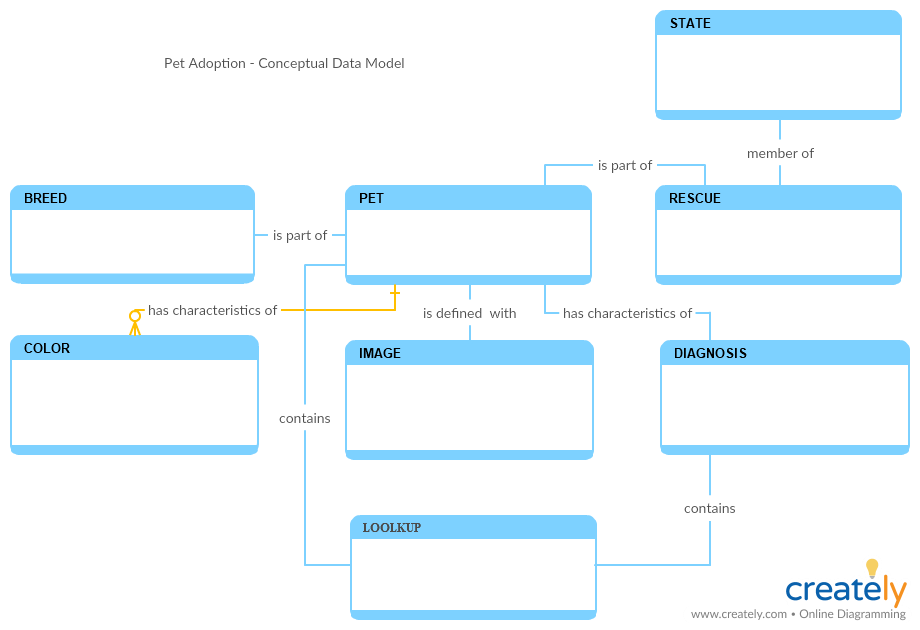
This diagram shows the hierarchy of the tables with their attributes:

**Data Warehouse Hierarchy of Data Tables**

Attribute Value

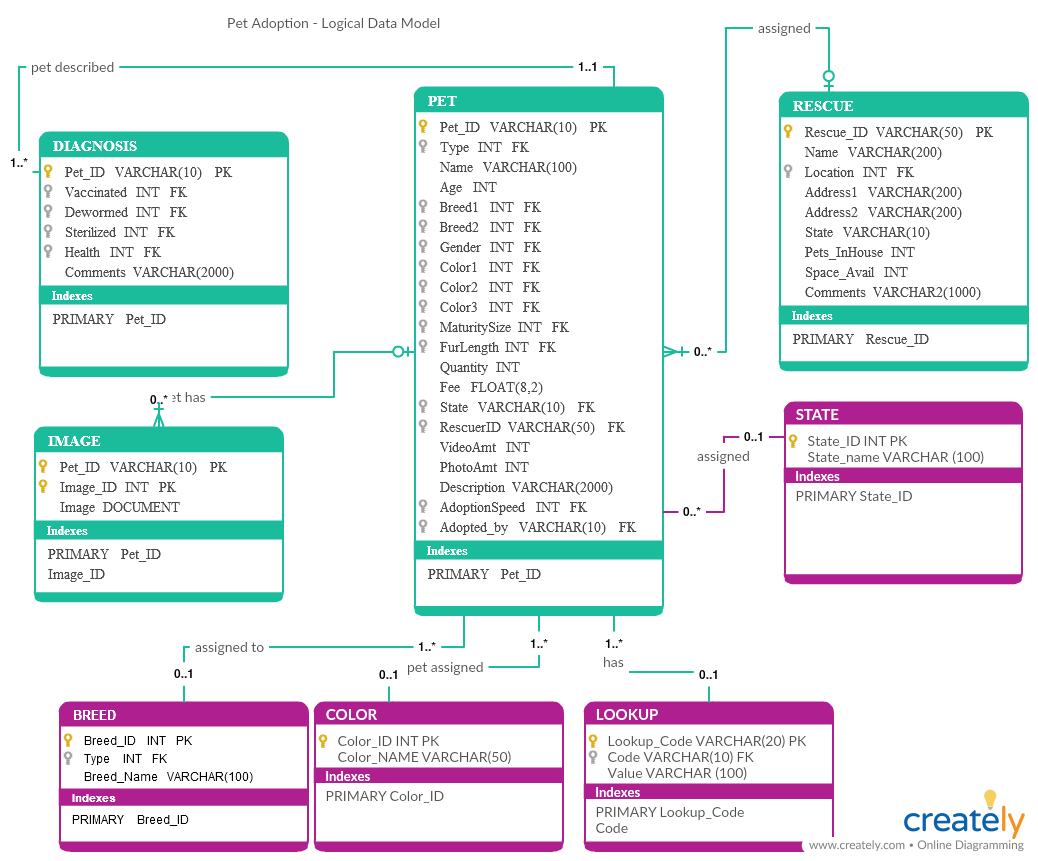
### Conceptual Data Model

The conceptual data model for the data uploaded into the database is defined in this figure.



### Logical Data Model

1. The logical data model describes the data and its entity-relationship between the object with the types of cardinality between the tables. The fact table is created at the physical model, and not shown in logical data model.



### Physical Data Model

The physical data model for the pet rescue data warehouse shall contain these attributes and their properties:

Database schema: PAWSWARE

| **Physical Data Model** | | | |
| --- | --- | --- | --- |
| **Table** | **Column** | **Data Type** | **Notes** |
| State | State\_ID  StateName | VARCHAR(10)  VARCHAR(50) | Primary Key |
| Rescue | Rescue\_ID  Shelter\_name  Address1  City  State  Zip\_code  Hours  Phone  Location  Pets\_inhouse  Space\_avail  Comments | VARCHAR(50)  VARCHAR(100)  VARCHAR(100)  VARCHAR(50)  VARCHAR(10)  VARCHAR(10)  VARCHAR(30)  VARCHAR(15)  VARCHAR(20)  INT  INT  VARCHAR(2000) | Primary Key |
| Breed | Breed\_Id  Btype  BreedName | INT  INT VARCHAR(50) | Primary Key |
| Color | Color\_ID  ColorName | INT  VARCHAR(50) | Primary Key |
|  |  |  |  |
| Pet | Pet\_ID Ptype Name  Age  Breed1  Breed2  Gender  Color1  Color2  Color3  MaturitySize  FullLength  Quantity  Fee  State  Rescue\_ID  VideoAmt  PhotoAmt  Enter\_date  Adopt\_date  Adoptionspeed  Description | VARCHAR(10)  VARCHAR(10)  VARCHAR(100)  INT  INT  INT  VARCHAR(10)  INT  INT  INT  VARCHAR(10)  VARCHAR(10)  INT  FLOAT(8,2)  VARCHAR(10)  VARCHAR(50)  INT  INT  DATE  DATE  VARCHAR(10)  VARCHAR(2000) | Primary Key  Foreign Key  Foreign Key  Foreign Key  Foreign Key  Foreign Key  Foreign Key  Foreign Key |
| Diagnosis | Pet\_ID  Vaccinated  Dewormed  Sterilized  Health  Sentiment  Comments | VARCHAR(10)  VARCHAR(10)  VARCHAR(10)  VARCHAR(10)  VARCHAR(10)  JSON  VARCHAR(2000) | Foreign Key |
| Image | Pet\_ID  Image\_ID  Imagepath  Sentiment | VARCHAR(10)  VARCHAR(10)  VARCHAR(1024)  JSON | Foreign Key  Primary Key |
| Lookup | Lookup\_code  Code  Value | VARCHAR(20)  VARCHAR(10)  VARCHAR(100) |  |

Appendix A: Acronyms

The list acronyms and associated translations used within the document are listed in Table 1.

Table 1 - Acronyms

| Acronym | Literal Translation |
| --- | --- |
| DWH | Data Warehouse |
| DDL | Data definition language |
| ETL | Extraction, transformation, and load |
| HDFS | Hadoop Distributed File System |
| OLTP | Online transaction processing |
| OLAP | Online analytical processing |
| U.S. | United States |
|  |  |

Appendix B: Architecture of Data Warehouse System



Across a cluster of computing nodes on commodity server hardware

In native formats

Upload all data into Data Lake

Structured and unstructured data:

-pet adoption demographics

-image sentiment API in json format



Web Application/User Interface

**Pet Adoption Data Warehouse**

Appendix C: Detailed Database Design: DDL

Data definition language for defining the PAWSWARE database and schema in MySQL:

|  |
| --- |
| # Create MySQL Database Users  CREATE USER 'lpalucci'@'userhostname' IDENTIFIED BY 'project';  # Create MySQL database  CREATE IF NOT EXISTS DATABASE PAWSWARE;  Use PAWSWARE;  # Grant MySQL Database user access using privileges.  GRANT ALL PRIVILEGES ON pawsware.\* TO lpalucci@localhost;  GRANT SELECT, INSERT, UPDATE, DELETE ON pawsware.\* TO lpalucci@localhost;  GRANT ALL ON \*.\* TO 'lpalucci'@'localhost';  # ----------------------------------------------------------------------------  -- -----------------------------------------------------  -- Schema PAWSWARE  -- -----------------------------------------------------  DROP SCHEMA IF EXISTS PAWSWARE;  CREATE SCHEMA IF NOT EXISTS PAWSWARE DEFAULT CHARACTER SET utf8 ;  USE pawsware  # -----------------------------------------------------  # Table pawsware.breed  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.breed;  CREATE TABLE breed (  breed\_ID INT NOT NULL primary key,  btype INT NOT NULL,  breedname VARCHAR(50) NOT NULL  );  # -----------------------------------------------------  # Table pawsware.color  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.color;  CREATE TABLE color (  color\_ID INT NOT NULL primary key,  colorname VARCHAR(50) NOT NULL  );  # -----------------------------------------------------  # Table pawsware.lookup  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.lookup;  CREATE TABLE lookup (  lookup\_code VARCHAR(20) NOT NULL,  code VARCHAR(10) NOT NULL,  value VARCHAR(100),  UNIQUE (lookup\_code, code)  );  # -----------------------------------------------------  # Table pawsware.state  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.state;  CREATE TABLE state (  state\_ID VARCHAR(10) NOT NULL primary key,  statename VARCHAR(50) NOT NULL  );  # -----------------------------------------------------  # Table pawsware.rescue  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.rescue;  CREATE TABLE rescue (  rescue\_ID VARCHAR(50) NOT NULL primary key,  shelter\_name VARCHAR(100) NOT NULL,  address1 VARCHAR(100) NULL,  city VARCHAR(50) NULL,  state VARCHAR(10) NULL,  zip\_code VARCHAR(10) NULL,  hours VARCHAR(30) NULL,  phone VARCHAR(15) NULL,  location VARCHAR(20) NULL,  pets\_inhouse INT NULL,  space\_avail INT NULL,  comments VARCHAR(2000) NULL  );  # -----------------------------------------------------  # Table pawsware.pet  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.pet;  CREATE TABLE pet (  pet\_ID VARCHAR(10) primary key,  ptype VARCHAR(10) NOT NULL,  name VARCHAR(100) NULL,  age INT NULL,  breed1 INT NOT NULL,  breed2 INT NULL,  gender VARCHAR(10) NOT NULL,  color1 INT NOT NULL,  color2 INT NULL,  color3 INT NULL,  maturitysize VARCHAR(10) NULL,  furlength VARCHAR(10) NULL,  quantity INT NULL,  fee FLOAT(8,2) NULL,  state VARCHAR(10) NOT NULL,  rescue\_ID VARCHAR(50) NOT NULL,  videoAmt INT NULL,  photoAmt INT NULL,  enter\_date date NOT NULL,  adopt\_date date NULL,  adoptionspeed VARCHAR(10) NULL,  description VARCHAR(2000) NULL,  CONSTRAINT c1\_pet\_pettype CHECK (ptype IN (select code from lookup where lookup\_code = 'PETTYPE')),  FOREIGN KEY (breed1) REFERENCES breed (breed\_ID),  FOREIGN KEY (breed2) REFERENCES breed (breed\_ID),  CONSTRAINT c1\_pet\_gender CHECK (gender IN (select code from lookup where lookup\_code = 'GENDER')),  FOREIGN KEY (color1) REFERENCES color (color\_ID),  FOREIGN KEY (color2) REFERENCES color (color\_ID),  FOREIGN KEY (color3) REFERENCES color (color\_ID),  CONSTRAINT c1\_pet\_maturitysize CHECK (maturitysize IN (select code from lookup where lookup\_code = 'MATURITY')),  CONSTRAINT c1\_pet\_furlength CHECK (furlength IN (select code from lookup where lookup\_code = 'LENGTH')),  CONSTRAINT c1\_pet\_adoptionspeed CHECK (adoptionspeed IN (select code from lookup where lookup\_code = 'ADOPTIONSPEED')),  FOREIGN KEY (state) REFERENCES state(state\_id),  FOREIGN KEY (Rescue\_ID) REFERENCES rescue(Rescue\_ID)  );  # -----------------------------------------------------  # Table pawsware.diagnosis  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.diagnosis;  CREATE TABLE diagnosis (  pet\_ID VARCHAR(10) NOT NULL,  vaccinated VARCHAR(10) NULL,  dewormed VARCHAR(10) NULL,  sterilized VARCHAR(10) NULL,  health VARCHAR(10) NULL,  sentiment JSON DEFAULT NULL,  comments VARCHAR(2000) NULL,  CONSTRAINT c1\_diagnosis\_vaccinated CHECK (ptype IN (select code from lookup where lookup\_code = 'YESNO')),  CONSTRAINT c1\_diagnosis\_dewormed CHECK (ptype IN (select code from lookup where lookup\_code = 'YESNO')),  CONSTRAINT c1\_diagnosis\_sterilized CHECK (ptype IN (select code from lookup where lookup\_code = 'YESNO')),  CONSTRAINT c1\_diagnosis\_health CHECK (ptype IN (select code from lookup where lookup\_code = 'HEALTH')),  FOREIGN KEY (pet\_ID) REFERENCES pet(pet\_ID)  );  # -----------------------------------------------------  # Table pawsware.image  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.image;  CREATE TABLE image (  pet\_ID VARCHAR(10) NOT NULL ,  image\_ID VARCHAR(10) NOT NULL primary key,  imagepath VARCHAR(1024) NULL,  sentiment JSON NULL,  FOREIGN KEY (pet\_ID) REFERENCES pet(pet\_ID)  );  # -----------------------------------------------------  # Table pawsware.petfact (FACT table)  # -----------------------------------------------------  DROP TABLE IF EXISTS PAWSWARE.petfact;  CREATE TABLE petfact (  pet\_ID VARCHAR(10) NOT NULL ,  breed\_ID INT NULL,  image\_ID VARCHAR(10) NULL,  state VARCHAR(10) NULL,  enter\_date date NOT NULL,  adopt\_date date NULL,  adoptionspeed VARCHAR(10) NULL,  time\_elapsed INT NULL  ); |