# Construcción de un Datawarehouse

## Lab Book Bases de Datos Avanzadas

César Calatrava Ruiz
cesar.calatrava.ruiz@gmail.com

16 Febrero 2015

# Índice

# Lunes, 16 Febrero 2015

## 1 Elección de los datos

Los datos elegidos para la construcción del datawarehouse han sido unos relacionados con el cáncer en Estados Unidos. Dicho *dataset* recoge el número de incidencias y muertes por cáncer durante una década, de 1999 a 2009, en EE.UU. Catorce campos y 671.641 registros definen la base de datos, estos registros están organizados por estado, sexo, tipo de cáncer y raza. También contiene algunos indicadores referentes a la edad. En la figura 2 se muestra una parte del *dataset* con el que se va a trabajar durante todo el proceso de construcción.

```
AREA,AGE_ADJUSTED_CI_LOWER,AGE_ADJUSTED_CI_UPPER,AGE_ADJUSTED_RATE,COUNT,EVENT_TYPE,POPULATION,RACE,SEX,SITE,YEAR,CRUDE_CI_LOWER,CRUDE_CI_UPPER,CRUDE_RATE
Alabama,359.1,374.2,366.6,9286,Incidence,2293259,All Races,Female,All Cancer Sites Combined,1999,396.7,413.2,404.9
Alabama,160.6,170.5,165.5,4366,Mortality,2293259,All Races,Female,All Cancer Sites Combined,1999,184.8,196.1,190.4
Alabama,360.4,375.3,367.8,9433,Incidence,2302777,All Races,Female,All Cancer Sites Combined,2000,401.4,418.0,409.6
Alabama,160.7,170.6,165.6,4425,Mortality,2302777,All Races,Female,All Cancer Sites Combined,2000,186.5,197.9,192.2
Alabama,375.4,390.6,383.0,9916,Incidence,2307692,All Races,Female,All Cancer Sites Combined,2001,421.3,438.2,429.7
Alabama,164.4,174.4,169.3,4550,Mortality,2307692,All Races,Female,All Cancer Sites Combined,2001,191.5,203.0,197.2
Alabama,380.7,395.9,388.2,10117,Incidence,2310501,All Races,Female,All Cancer Sites Combined,2002,429.4,446.5,437.9
Alabama,157.6,167.3,162.4,4407,Mortality,2310501,All Races,Female,All Cancer Sites Combined,2002,185.1,196.5,190.7
Alabama,356.8,371.6,364.1,9577,Incidence,2317733,All Races,Female,All Cancer Sites Combined,2003,405.0,421.6,413.2
Alabama,158.0,167.7,162.8,4472,Mortality,2317733,All Races,Female,All Cancer Sites Combined,2003,187.3,198.7,192.9
Alabama,376.0,391.0,383.4,10199,Incidence,2329595,All Races,Female,All Cancer Sites Combined,2004,429.3,446.4,437.8
Alabama,157.7,167.3,162.5,4498,Mortality,2329595,All Races,Female,All Cancer Sites Combined,2004,187.5,198.8,193.1
Alabama,380.8,395.8,388.3,10474,Incidence,2343529,All Races,Female,All Cancer Sites Combined,2005,438.4,455.6,446.9
Alabama,158.9,168.5,163.7,4599,Mortality,2343529,All Races,Female,All Cancer Sites Combined,2005,190.6,202.0,196.2
Alabama,390.6,397.3,393.9,54831,Incidence,11947369,All Races,Female,All Cancer Sites Combined,2005-2009,455.1,462.8,458.9
Alabama,155.3,159.5,157.4,22867,Mortality,11947369,All Races,Female,All Cancer Sites Combined,2005-2009,188.9,193.9,191.4
Alabama,376.6,391.4,383.9,10516,Incidence,2371941,All Races,Female,All Cancer Sites Combined,2006,434.9,451.9,443.3
```

Figura 1: Dataset utilizado

## 2 Diseño del modelo

Para afrontar el diseño del datawarehouse se va a seguir un modelo ROLAP, que nos permite la construcción de una base de datos multidimensional pero siguiendo un paradigma relacional. La estructura a seguir para el diseño va a ser un modelo en estrella, por tanto en primer lugar cabe señalar las dimensiones elegidas

- **Dimensión área:** tal y como se expuso en el anterior apartado, tanto las incidencias como las muertes vienen organizadas por estado, cada uno de los estados de EE.UU.
- **Dimensión raza:** los datos se encuentran categorizados en distintas razas que son: todas las razas, americana india/nativa de Alaska, asiática, negra, hispánica y blanca.
- **Dimensión zona:** el objetivo de la definición de esta dimensión es la de recoger los diferentes tipos de cáncer recogidos en el *dataset*.

- **Dimensión año:** como ya se ha comentado con anterioridad los datos de los que contamos corresponden al período de 1999-2009.
- **Dimensión sexo:** en la base de datos disponemos de tres agrupaciones referentes al sexo: hombre, mujer y hombre y mujer.

Como medidas se han tomado las incidencias y muertes, y la población total de ese estado en el momento que se registraron los datos. Por tanto nuestra tabla de hechos contendría las claves externas referentes a las tablas de dimensiones definidas anteriormente y las tres medidas escogidas.

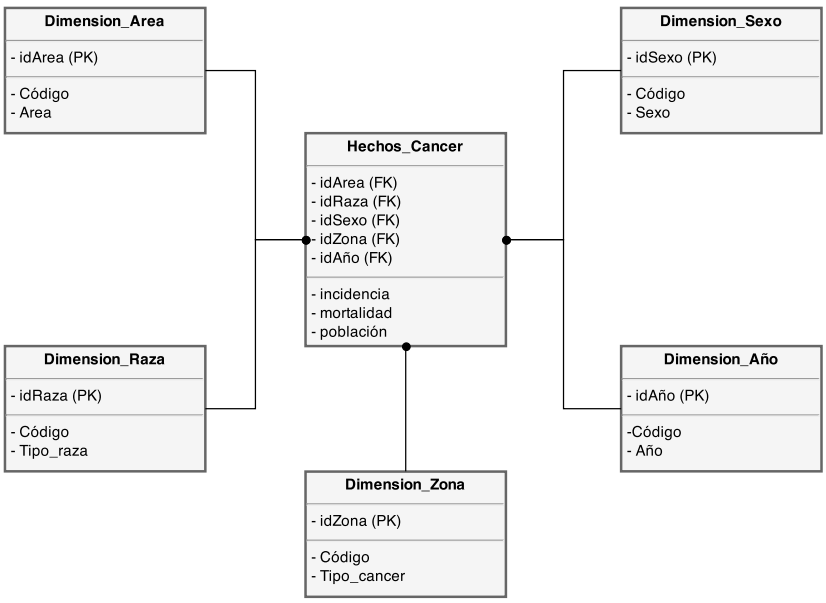


Figura 2: Diseño estrella siguiendo modelo ROLAP

Para la creación del modelo explicado con anterioridad se ha desarrollado un script, el 1 que tiene como entrada la base de datos donde queremos alojar nuestro datawarehouse, encargado de la construcción de todas las tablas y la relación entre ellas.

# Miércoles, 18 Febrero 2015

## 1 Comienzo del proceso ETL

Una vez definido el modelo a seguir, vamos a realizar un primer preproceso de los datos para eliminar los indicadores de edad que aparecen ya que estos no forman parte de nuestro diseño. Para esta limpieza se ha usado el script 2, el resultado ha sido el siguiente, figura 1:

```
Alabama,61,Incidence,539198,Black,Male,Pancreas,1999
Alabama,59,Mortality,539198,Black,Male,Pancreas,1999
Alabama,55,Incidence,543195,Black,Male,Pancreas,2000
Alabama,54,Mortality,543195,Black,Male,Pancreas,2000
Alabama,49,Incidence,546671,Black,Male,Pancreas,2001
Alabama,66,Mortality,546671,Black,Male,Pancreas,2001
Alabama,52,Incidence,548932,Black,Male,Pancreas,2002
Alabama,55,Mortality,548932,Black,Male,Pancreas,2002
Alabama,58,Incidence,552396,Black,Male,Pancreas,2003
Alabama,57,Mortality,552396,Black,Male,Pancreas,2003
Alabama,48,Incidence,555151,Black,Male,Pancreas,2004
Alabama,38,Mortality,555151,Black,Male,Pancreas,2004
Alabama,65,Incidence,559813,Black,Male,Pancreas,2005
Alabama,67,Mortality,559813,Black,Male,Pancreas,2005
```

Figura 1: Salida correspondiente al preproceso

## 2 Extracción de las dimensiones

A continuación se procederá a la extracción de las diferentes dimensiones que conforman nuestro datawarehouse:

- Dimensión area: se extraerá la posición 0 del csv que corresponde a cada uno de los estados recogidos en el conjunto de datos con el script 3.
- Dimensión raza: para la extracción de la dimensión raza el script utilizado ha sido 4.
- Dimensión zona: para la dimensión zona, correspondiente a los distintos tipos de cánceres recogidos en el *dataset* se ha utilizado el script 5.
- Dimensión año: además del registro de los años comprendidos entre 1999 y 2009, también encontramos un registro correspondiente a 2005-2009 el cual no será

añadido al data warehouse. Para la extracción del año se ha usado el script 6 encargado de sacar cada uno de los años.

- Dimensión sexo: el script utilizado en este caso ha sido 7.

## 3 Extracción de las medidas

Para la extracción de las tres medidas seleccionadas con anterioridad se decidido extraer por un lado las incidencias además de la población para su posterior transformación y por el otro las muertes. Una vez he hecho esto, se añadirán las muertes al archivo que contiene las incidencias y los demás datos para la posterior transformación. Los scripts usados para esto han sido 8 y 9.

# Domingo, 1 Marzo 2015

## 1 Automatización del proceso de extracción

Para evitar la ejecución una a una de los scripts encargados de la extracción de los datos se ha realizado un script para automatizar dicho proceso, el script 10.

## 2 Transformación de los datos

Una vez que tenemos todos los datos extraídos, el siguiente paso es la transformación de los mismos centrada en dos aspectos: la limpieza de valores perdidos y/o nulos, y la codificación de todos los datos referentes a las dimensiones.

Para la limpieza de las medidas "incidencias" y "mortalidad" se han desarrollado dos scripts encargados de eliminar los registros que contienen como dato '?', '-' o '.' y también los registros correspondientes al período 2005-2009, estos registros también serán eliminados de la dimensión año. Los script utilizados para la limpieza de las medidas han sido 11 y 12.

En cuanto a la codificación de las dimensiones, el proceso ha sido el de codificar cada uno de los registros que conforman las tablas de dimensiones, es decir, en la dimensión año por ejemplo al año 1999 se le ha asignado un 0 y así sucesivamente. Los scripts usados para esta transformación han sido 13, 14, 15, 16 y 17.

Tanto la tarea de limpieza como la de codificación ha sido automatizada con el script 18.

Una vez que los datos ya están limpios y transformados el siguiente paso es la inclusión de los datos de la mortalidad a los demás datos, donde ya tenemos las incidencias, población, sexo, etc. Después de esto, lo siguiente será la codificación de toda las dimensiones en la tabla de hechos, es decir, por ejemplo donde tenemos el año 1999, ahora tendremos un 1. Esto es debido al uso de las claves foránea (PK), y se aplicará a todas las dimensiones.

Este proceso de reemplazo o codificación ha sido realizado con *rStudio* y *Numbers*, herramientas tremendamente útiles para el trabajo con archivos de tipo csv.

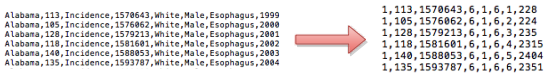


Figura 1: Proceso de transformación

# Lunes, 2 Marzo 2015

## 1 Desarrollo scripts para la carga de los datos

La estructura de todos los script que se van a mostrar a continuación es la misma: realizan un **INSERT TO** de los diversos datos extraídos (archivos csv) en el datawarehouse.

- Carga area: el script usado para ello ha sido 19.
- Carga raza: el script usado para realizar la carga ha sido 20.
- Carga zona: para esta carga el script utilizado ha sido 21.
- Carga año: el script utilizado para cargar los datos ha sido 22.
- Carga sexo: el script usado para ello ha sido 23.
- Carga hechos: para llevar a cabo esta carga se ha utilizado el script 24.

## 2 Automatización del proceso

Como ya ocurría con el proceso de extracción, el proceso de carga también ha sido automatizado, el script utilizado para dicha automatización ha sido 25.

# Martes, 3 Marzo 2015

## 1 Resultados y consultas realizadas al datawarehouse

Señalar que no han sido cargados todos los datos, en concreto se ha realizado la carga de los primeros catorce estados. En la figura 1 podemos ver una parte de la tabla de hechos tras realizar la carga.

| # | Id_Area | Id_Raza | Id_Sexo | Id_Zona | Id_Anyo | Incidencia | Mortalidad | Poblacion |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 2 | 1 | 1 | 8733 | 4425 | 2302734 |
| 2 | 1 | 1 | 2 | 1 | 2 | 9433 | 4550 | 2302777 |
| 3 | 1 | 1 | 2 | 1 | 3 | 9916 | 4407 | 2307692 |
| 4 | 1 | 1 | 2 | 1 | 4 | 10117 | 4472 | 2310501 |
| 5 | 1 | 1 | 2 | 1 | 5 | 9577 | 4498 | 2317733 |
| 6 | 1 | 1 | 2 | 1 | 6 | 10199 | 4599 | 2329595 |
| 7 | 1 | 1 | 2 | 1 | 7 | 10474 | 4490 | 2343529 |
| 8 | 1 | 1 | 2 | 1 | 8 | 10516 | 4492 | 2371941 |
| 9 | 1 | 1 | 2 | 1 | 9 | 10924 | 4618 | 2392116 |
| 10 | 1 | 1 | 2 | 1 | 10 | 11519 | 4668 | 2412687 |
| 11 | 1 | 1 | 2 | 1 | 11 | 11398 | 95 | 2427096 |
| 12 | 1 | 1 | 2 | 2 | 1 | 131 | 91 | 2293259 |
| 13 | 1 | 1 | 2 | 2 | 2 | 147 | 99 | 2302777 |
| 14 | 1 | 1 | 2 | 2 | 3 | 144 | 100 | 2307692 |
| 15 | 1 | 1 | 2 | 2 | 4 | 138 | 91 | 2310501 |

Figura 1: Tabla Hechos

Se han realizado algunas consultas sobre el datawarehouse para comprobar sus posibles aplicaciones:

1. ¿Cuál es el número total de incidencias en Alabama en el año 2009 por tipo de sexo?, la consulta utilizada para dar respuesta a esta pregunta ha sido 26.

2. ¿Cuál es el número total de incidencias ordenadas de forma ascendente en todos los años registrados en función del tipo de cáncer?, consulta 27.

El resultado de ambas consultas puede verse en la figura 2

| # | SUM(Hechos.Incidencia) | Dimension_Sexo.Sexo |
|---|---|---|
| 1 | 44875 | Female |
| 2 | 51186 | Male |
| 3 | 95960 | Male and Female |

(a) Query 1

| # | SUM(Hechos.Incidencia) | Dimension_Zona.Tipo_cancer |
|---|---|---|
| 1 | 19931 | Kaposi Sarcoma |
| 2 | 49272 | Mesothelioma |
| 3 | 176657 | Hodgkin Lymphoma |
| 4 | 293342 | Cervix |
| 5 | 307136 | Esophagus |
| 6 | 256295 | Larynx |
| 7 | 360171 | Myeloma |
| 8 | 458302 | Stomach |
| 9 | 394014 | Liver and Intrahepatic Bile Duct |

(b) Query 2

Figura 2: Consultas realizadas

# Scripts

Código 1: Construcción de la base de datos

```
import sqlite3
import sys
import csv

connects = sqlite3.connect(sys.argv[1])
query = connects.cursor()

query.executescript("""
DROP TABLE IF EXISTS Dimension_Area;
CREATE TABLE Dimension_Area(
  IdArea INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  Codigo INTEGER NOT NULL,
  Area TEXT NOT NULL);
""")

query.executescript("""
DROP TABLE IF EXISTS Dimension_Raza;
CREATE TABLE Dimension_Raza(
  IdRaza INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  Codigo INTEGER NOT NULL,
  Tipo_raza TEXT NOT NULL);
""")

query.executescript("""
DROP TABLE IF EXISTS Dimension_Sexo;
CREATE TABLE Dimension_Sexo(
  IdSexo INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  Codigo INTEGER NOT NULL,
  Sexo TEXT NOT NULL);
""")

query.executescript("""
DROP TABLE IF EXISTS Dimension_Zona;
CREATE TABLE Dimension_Zona(
  IdZona INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  Codigo INTEGER NOT NULL,
  Tipo_cancer TEXT NOT NULL);
""")

query.executescript("""
DROP TABLE IF EXISTS Dimension_Anyo;
CREATE TABLE Dimension_Anyo(
```

```
43      IdAnyo INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
44      Codigo INTEGER NOT NULL,
45      Anyo INTEGER NOT NULL);
46  """)
47
48  query.executescript("""
49  DROP TABLE IF EXISTS Hechos;
50  CREATE TABLE Hechos(
51      Id_Area INTEGER NOT NULL,
52      Id_Raza INTEGER NOT NULL,
53      Id_Sexo INTEGER NOT NULL,
54      Id_Zona INTEGER NOT NULL,
55      Id_Anyo INTEGER NOT NULL,
56      Incidencia INTEGER NOT NULL,
57      Mortalidad INTEGER NOT NULL,
58      Poblacion INTEGER NOT NULL,
59      FOREIGN KEY (Id_Area) REFERENCES Dimension_Area(idArea),
60      FOREIGN KEY (Id_Raza) REFERENCES Dimension_Raza(idRaza),
61      FOREIGN KEY (Id_Sexo) REFERENCES Dimension_Sexo(idSexo),
62      FOREIGN KEY (Id_Zona) REFERENCES Dimension_Zona(idZona),
63      FOREIGN KEY (Id_Anyo) REFERENCES Dimension_Anyo(idAnyo));
64  """)
65
66  connects.commit()
67  connects.close()
```

Código 2: Preproceso de los datos

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
try:
    reader = csv.reader(f)
    for row in reader:
        if row[4]!='?' and row[4]!='-' and row[4]!='.':
                    lista = [row[0],row[4], row[5], row[6], row[7], row
                        [8], row[9], row[10]]
                    data.append(lista)

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 3: Extracción dimensión area

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
est=""
try:
    reader = csv.reader(f)
    for row in reader:
        nest=row[0]
        if est!=nest:
            lista=[row[0]]
            data.append(lista)
            lista=[]
            est=nest

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 4: Extracción dimensión raza

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
raza=""
lit = []
try:
    reader = csv.reader(f)
    for row in reader:
        nraza=row[7]
        if raza!=nraza:
            if nraza in lit:
                pass
            else:
                lista=[row[7]]
                lit.append(row[7])
                data.append(lista)
                lista=[]
                raza=nraza

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 5: Extracción dimensión zona

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
can=""
lit = []
try:
    reader = csv.reader(f)
    for row in reader:
        ncan=row[9]
        if can!=ncan:
            if ncan in lit:
                pass
            else:
                lista=[row[9]]
                lit.append(row[9])
                data.append(lista)
                lista=[]
                can=ncan

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 6: Extracción dimensión año

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
ano=""
lit = []
try:
    reader = csv.reader(f)
    for row in reader:
        nano=row[10]
        if ano!=nano:
            if nano in lit:
                pass
            else:
                lista=[row[10]]
                lit.append(row[10])
                data.append(lista)
                lista=[]
                ano=nano

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 7: Extracción dimensión sexo

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
sexo=""
lit = []
try:
    reader = csv.reader(f)
    for row in reader:
        nsexo=row[8]
        if sexo!=nsexo:
            if nsexo in lit:
                pass
            else:
                lista=[row[8]]
                lit.append(row[8])
                data.append(lista)
                lista=[]
                sexo=nsexo

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 8: Extracción incidencias

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
try:
    reader = csv.reader(f)
    for row in reader:
        if row[5]=='Incidence':
            lista = [row[0],row[4],row[5],row[6],row[7],row[8],row[9],row
                [10]]
            data.append(lista)

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 9: Extracción mortalidad

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
try:
    reader = csv.reader(f)
    for row in reader:
        if row[5]=='Mortality':
            lista = [row[0],row[4],row[5],row[6],row[7],row[8],row[9],row
                [10]]
            data.append(lista)

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 10: Automatización proceso extracción

```
#!/bin/bash

echo 'Proceso de extraccion en marcha...'

python extraccion_area.py csv/CANCER_BY_AREA.csv csv/area.csv

python extraccion_raza.py csv/CANCER_BY_AREA.csv csv/raza.csv

python extraccion_zona.py csv/CANCER_BY_AREA.csv csv/zona.csv

python extraccion_anyo.py csv/CANCER_BY_AREA.csv csv/anyo.csv

python extraccion_sexo.py csv/CANCER_BY_AREA.csv csv/sexo.csv

python extraccion_incidencia.py csv/CANCER_BY_AREA.csv csv/incidencias.
    csv

python extraccion_mortalidad.py csv/CANCER_BY_AREA.csv csv/mortalidad.csv

echo 'Proceso de extraccion finalizado'
```

Código 11: Transformación incidencias

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
try:
    reader = csv.reader(f)
    for row in reader:
        if row[4]!='?' and row[4]!='-' and row[4]!='.':
            if row[10]!='2005-2009':
                lista = [row[0],row[4],row[5],row[6],row[7],row[8],row
                    [9],row[10]]
                data.append(lista)

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 12: Transformación mortalidad

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
try:
    reader = csv.reader(f)
    for row in reader:
        if row[4]!='?' and row[4]!='-' and row[4]!='.':
            if row[10]!='2005-2009':
                lista = [row[4]]
                data.append(lista)

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 13: Transformación dimensión area

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
cod=-1
est=""
try:
    reader = csv.reader(f)
    for row in reader:
        lista =[cod,row[0]]
        data.append(lista)
        cod=cod+1

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 14: Transformación dimensión raza

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
cod=-1
est=""
try:
    reader = csv.reader(f)
    for row in reader:
        lista =[cod,row[0]]
        data.append(lista)
        cod=cod+1

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 15: Transformación dimensión zona

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
cod=-1
est=""
try:
    reader = csv.reader(f)
    for row in reader:
        lista =[cod,row[0]]
        data.append(lista)
        cod=cod+1

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 16: Transformación dimensión año

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
cod=-1
est=""
try:
    reader = csv.reader(f)
    for row in reader:
        if row[0]!="2005-2009":
            lista =[cod,row[0]]
            data.append(lista)
            cod=cod+1

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 17: Transformación dimensión sexo

```
import csv
import sys

f = open(sys.argv[1], 'rt')
data = []
cod=-1
est=""
try:
    reader = csv.reader(f)
    for row in reader:
        lista =[cod,row[0]]
        data.append(lista)
        cod=cod+1

finally:
    f.close()

f = open(sys.argv[2],'wt')
try:
  writer = csv.writer(f)
  for i in range(1,len(data)):
    writer.writerow((data[i]))

finally:
    f.close()
```

Código 18: Automatización proceso transformación

```
#!/bin/bash

echo 'Proceso de transformacion en marcha...'

python trans_area.py csv/area.csv csv/area.csv

python trans_raza.py csv/raza.csv csv/raza.csv

python trans_zona.py csv/zona.csv csv/zona.csv

python trans_anyo.py csv/anyo.csv csv/anyo.csv

python trans_sexo.py csv/sexo.csv csv/sexo.csv

python trans_incidencia.py csv/incidencias.csv csv/incidencias.csv

python trans_mortalidad.py csv/mortalidad.csv csv/mortalidad.csv

echo 'Proceso de transformacion finalizado'
```

Código 19: Carga area

```
import sqlite3
import sys
import csv

connects = sqlite3.connect(sys.argv[1])
query = connects.cursor()

f = open(sys.argv[2], 'rb')
try:
  reader = csv.reader(f,delimiter=',')
  for row in reader:
    to_db = [unicode(row[0], "utf8"), unicode(row[1], "utf8")]
    try:
      query.execute("INSERT INTO Dimension_Area(Codigo,Area) VALUES(?, ?)
          ;", to_db)
    except sqlite3.IntegrityError as err:
        print(err)
    connects.commit()
  connects.close()
finally:
    f.close()
```

Código 20: Carga raza

```
import sqlite3
import sys
import csv

connects = sqlite3.connect(sys.argv[1])
query = connects.cursor()

f = open(sys.argv[2], 'rb')
try:
  reader = csv.reader(f,delimiter=',')
  for row in reader:
    to_db = [unicode(row[0], "utf8"), unicode(row[1], "utf8")]
    try:
      query.execute("INSERT INTO Dimension_Raza(Codigo,Tipo_raza) VALUES
          (?, ?);", to_db)
    except sqlite3.IntegrityError as err:
        print(err)
    connects.commit()
  connects.close()
finally:
    f.close()
```

Código 21: Carga zona

```
import sqlite3
import sys
import csv

connects = sqlite3.connect(sys.argv[1])
query = connects.cursor()

f = open(sys.argv[2], 'rb')
try:
  reader = csv.reader(f,delimiter=',')
  for row in reader:
    to_db = [unicode(row[0], "utf8"), unicode(row[1], "utf8")]
    try:
      query.execute("INSERT INTO Dimension_Zona(Codigo,Tipo_cancer)
          VALUES(?, ?);", to_db)
    except sqlite3.IntegrityError as err:
        print(err)
    connects.commit()
  connects.close()
finally:
    f.close()
```

Código 22: Carga año

```
import sqlite3
import sys
import csv

connects = sqlite3.connect(sys.argv[1])
query = connects.cursor()

f = open(sys.argv[2], 'rb')
try:
  reader = csv.reader(f,delimiter=',')
  for row in reader:
    to_db = [unicode(row[0],"utf8"),unicode(row[1],"utf8")]
    try:
      query.execute("INSERT INTO Dimension_Anyo(Codigo, Anyo) VALUES(?,
          ?);", to_db)
    except sqlite3.IntegrityError as err:
        print(err)
    connects.commit()
  connects.close()
finally:
    f.close()
```

Código 23: Carga sexo

```
import sqlite3
import sys
import csv

connects = sqlite3.connect(sys.argv[1])
query = connects.cursor()

f = open(sys.argv[2], 'rb')
try:
  reader = csv.reader(f,delimiter=',')
  for row in reader:
    to_db = [unicode(row[0], "utf8"), unicode(row[1], "utf8")]
    try:
      query.execute("INSERT INTO Dimension_Sexo(Codigo,Sexo) VALUES(?, ?)
          ;", to_db)
    except sqlite3.IntegrityError as err:
        print(err)
    connects.commit()
  connects.close()
finally:
    f.close()
```

Código 24: Carga hechos

```
import sqlite3
import sys
import csv

connects = sqlite3.connect(sys.argv[1])
query = connects.cursor()

f = open(sys.argv[2], 'rb')
try:
  reader = csv.reader(f,delimiter=',')
  for row in reader:
    to_db = [unicode(row[0],"utf8"),unicode(row[3],"utf8"),unicode(row
        [4],"utf8"),unicode(row[5],"utf8"),
    unicode(row[6],"utf8"),unicode(row[1],"utf8"),unicode(row[7],"utf8"),
        unicode(row[2],"utf8")]
    try:
      query.execute("INSERT INTO Hechos(Id_Area, Id_Raza, Id_Sexo,
          Id_Zona, Id_Anyo, Incidencia, Mortalidad, Poblacion) VALUES(?,
          ?, ?, ?, ?, ?, ?, ?);", to_db)
    except sqlite3.IntegrityError as err:
        print(err)
    connects.commit()
  connects.close()
finally:
    f.close()
```

Código 25: Automatización proceso carga

```
#!/bin/bash

echo 'Proceso de carga en marcha...'
python carga_sexo.py datawarehouse.db csv/sexo.csv

python carga_raza.py datawarehouse.db csv/razas.csv

python carga_cancer.py datawarehouse.db csv/cancer.csv

python carga_area.py datawarehouse.db csv/area.csv

python carga_anyo.py datawarehouse.db csv/anyo.csv

python carga_hechos.py datawarehouse.db csv/hechos.csv

echo 'Proceso de carga finalizado'
```

Código 26: Query 1

```
SELECT SUM(Hechos.Incidencia), Dimension_Sexo.Sexo
FROM Hechos
INNER JOIN Dimension_Sexo, Dimension_Anyo, Dimension_Area
ON Hechos.Id_Sexo = Dimension_Sexo.IdSexo
AND Hechos.Id_Anyo = Dimension_Anyo.IdAnyo
AND Hechos.Id_Area = Dimension_Area.IdArea
WHERE Dimension_Anyo.Anyo = "2009"
AND Dimension_Area.Area = "Alabama"
GROUP BY Dimension_Sexo.Sexo;
```

Código 27: Query 2

```
SELECT SUM(Hechos.Incidencia), Dimension_Zona.Tipo_cancer
FROM Hechos
INNER JOIN Dimension_Zona
ON Hechos.Id_Zona = Dimension_Zona.IdZona
GROUP BY Dimension_Zona.Tipo_cancer
ORDER BY Hechos.Incidencia ASC;
```