Proyecto final

HERRAMIENTAS PARA DATOS ESTRUCTURADOS

- RAMÍREZ SÁNCHEZ JUAN CARLOS
- ESCAMILLA ELIAS MIGUEL SALOMÓN

TECNOLÓGICO NACIONAL DE MÉXICO EN CELAYA

INGENIERÍA EN SISTEMAS COMPUTACIONALES

Introducción

En la actualidad, la era de la información y tecnología se encuentra en pleno auge lo que ha permitido el descubrimiento de un elemento indispensable hoy en día: los datos. Actualmente, se producen grandes cantidades de datos, los cuales mediante su análisis es posible obtener información para la toma de decisiones estrategias y el desarrollo de acciones, así como la predicción de futuros escenarios. Dada la magnitud de dicha herramienta esta se puede emplear en diferentes áreas y sectores, como el económico, empresarial, institucional, etc.

Por lo cual, para el desarrollo de este proyecto se realizó un análisis de datos empleando información de reservaciones de hoteles de una asociación a nivel global, para la extracción de información clave que permita la toma de decisiones y acciones. Mediante el desarrollo de un Data WareHouse para el almacenamiento de grandes cantidades de datos empleando el proceso ETL, para la obtención de datos importantes para análisis de datos. Además del desarrollo de un cubo OLAP que permita potenciar el análisis de datos mediante el manejo de dimensiones para la obtención de información descriptiva, esto representado por un Dashboard, que, mediante gráficas y mecanismos de segmentación, permiten la eficiente y óptima visualización de estos datos. Finalmente, se implementó un algoritmo de minería de datos para la obtención de patrones y tendencias en la información. Esto con la finalidad de desarrollar herramientas que permitan el análisis y exploración de grandes cantidades de datos de reservaciones que permitan a los hoteles la realización de una toma de decisiones y acciones estratégicas para la mejora de sus servicios mediante la explotación de áreas de oportunidad.

Objetivo General

Desarrollar un análisis de datos basado en la información de reservaciones de hoteles que permita a la administración de los hoteles, su centralización y obtención de información clave y valiosa para la toma de decisiones estratégicas para la mejora de sus servicios. Mediante el desarrollo de un Data WareHouse y un cubo OLAP además de la implementación de algoritmos de minería de datos para la identificación de patrones y tendencias.

Objetivos Específicos

- Implementar una base de datos con la información obtenida
- Diseñar e implementar un Data WareHouse para la centralización de datos
- Diseñar e implementar una base de datos intermedia para el proceso ETL
- Desarrollar un proceso ETL para la alimentación de datos del Data Warehouse
- Implementar un cubo OLAP para el análisis de datos del Data WareHouse

- Desarrollar un Dashboard empleando el cubo OLAP para la presentación de información visual
- Emplear el algoritmo de minería de datos de árboles de decisión para la obtención de información clave

Metodología

Para este proyecto se ha definido una metodología de Desarrollo de Sistemas de Información en Cascada o Waterfall, que cuenta con características específicas que se adaptan a proyectos de análisis de datos. Se ha elegido esta metodología de acuerdo con la naturaleza de este proyecto que no necesita de constantes pruebas y despliegues del producto final, teniendo entonces un desarrollo más estructurado y ordenado de forma simple y efectiva.

Las fases para utilizar en esta metodología son:

Requisitos:

- Identificar las necesidades y problemas del ámbito del proyecto.
- Definir las métricas e indicadores de análisis que se plantea analizar.

BD original:

- Definir las fuentes de datos y sus transformaciones necesarias.
- Preparar los datos para la extracción.

BD Intermedia:

- Diseñar modelo relacional.
- Construir estructura de la BD en SQL.
- Extraer datos de la BD original y depositarlos conforme corresponda en la BD Intermedia.
- Transformar y corregir irregularidades en los datos.

Data Warehouse:

- Diseñar modelo multidimensional.
- Construir estructura del Data Warehouse en SQL.
- Cargar datos de la BD Intermedia y depositarlos conforme corresponda en el Data Warehouse.

Cubo OLAP:

- Establecer el Data Warehouse como origen de datos
- Generar vista del origen de datos, conforme a los procedimientos de Visual Studio 2019.

- Generar y procesar el cubo OLAP conforme a la vista establecida.

Dashboard:

- Diseñar la interfaz y la disposición de los elementos visuales.
- Establecer conexión entre el cubo OLAP y Excel.
- Generar tablas dinámicas.
- Establecer segmentadores de datos.
- Generar gráficas y elementos visuales dinámicos.

Minería de datos:

- Analizar los requerimientos y usos de la minería de datos con el modelo obtenido.
- Definir algoritmo, dimensiones y medidas a utilizar.
- Diseñar uso e implementación del algoritmo.
- Diseñar estructura de minería de datos en el cubo OLAP.
- Implementar algoritmo conforme a los procedimientos de Visual Studio 2019 y establecer dimensión y medidas.
- Procesar estructura de minería de datos.
- Analizar modelo de predicción resultante.

Mediante esta metodología se plantea un desarrollo ordenado en cadena que permita la generación lineal del análisis de datos

Indicadores de Análisis

Por medio de este Dashboard, se plantea su uso con el fin de conocer de forma gráfica y eficiente la información respecto a los siguientes indicadores, relacionados con el tema del proyecto y el contenido de la fuente de datos:

- Ingresos totales por reservación
- Tarifa promedio por reservación
- Recuento de reservaciones
- Numero de cambios por reservación
- Número de solicitudes especiales
- Número de días que realizan reservaciones con antelación
- Numero de noches por reservación
- Número de días en lista de espera para la realización de reservaciones
- Cantidad de espacios de estacionamiento solicitados por reservación
- Número total de personas por reservas
- Cantidad adultos, niños y bebes por reservas

Bases de Datos

Fuente (breve explicación y diagrama relacional)

La base de datos fuente corresponde al dataset de Hotel Booking Dataset, el cual contiene aproximadamente 119,390 registros de datos de reservas realizadas entre el 1 de julio del 2015 y el 31 de agosto del 2017 en dos tipos de hoteles además de incluir información tanto de reservas realizadas como canceladas. Dichos registros corresponden a datos reales de hoteles, centrándose en información específica acerca de la reserva sin elementos relacionales como: país, estado de la reserva, fecha de llegada, numero de noches, etc. El dataset almacena la información en un archivo del tipo CSV.

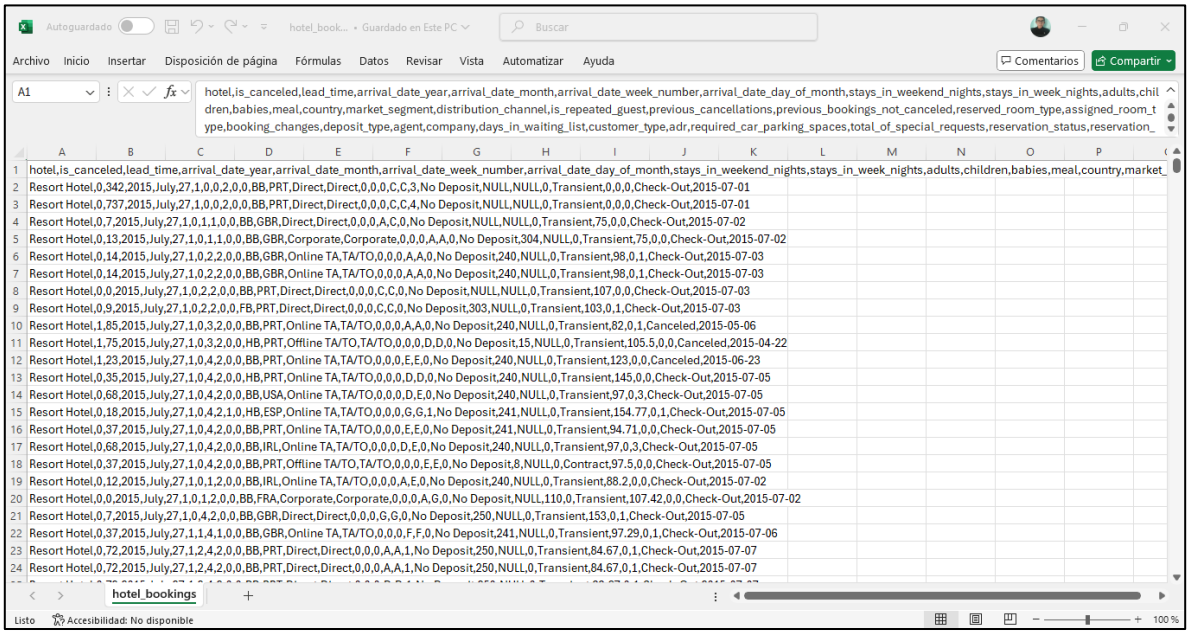

Figura 5.1.1. Captura del dataset en formato CSV

Por lo cual, para el desarrollo del proyecto se cargó el archivo CSV en una base de datos llamada hotel_bookings conformada por una única tabla.

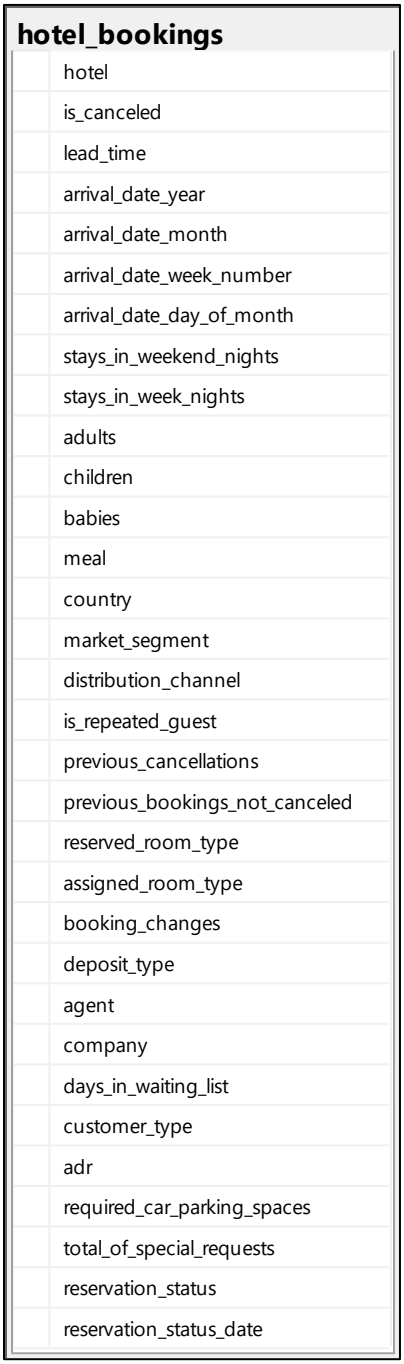

Figura 5.1.2. Captura de la tabla resultante de la transformación de datos

Data WareHouse (diagrama multidimensional y script de creación)

Diagrama multidimensional

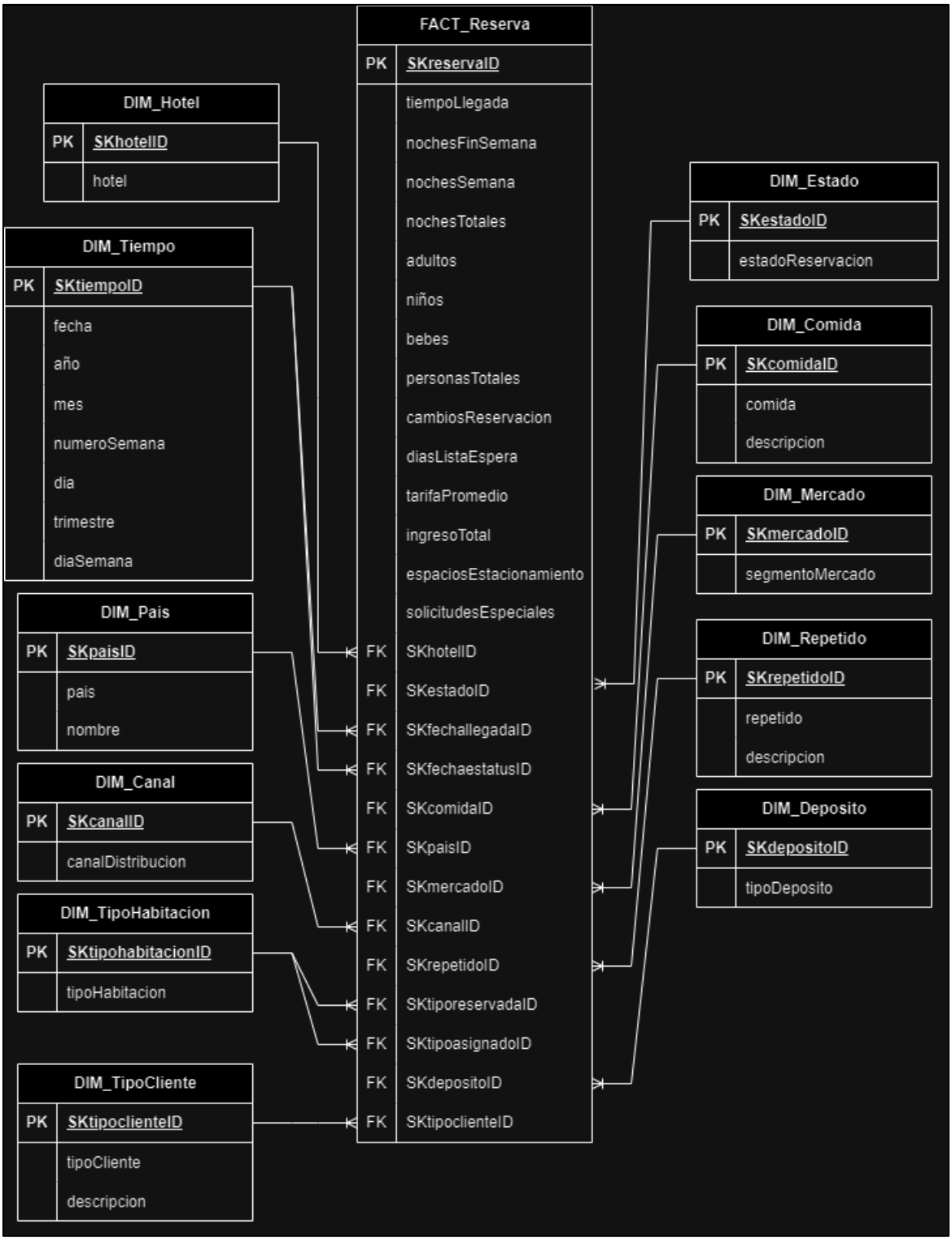

Figura 5.2.1. Diagrama multidimensional del Data warehouse

Script

```
-- Crear base de datos
create database reservasDWH;

-- Usar base de datos
use reservasDWH;

--  Crear tablas
create table DIM_Hotel(
	SKhotelID int not null,
	hotel nvarchar(50),
	constraint hotelPK primary key(SKhotelID)
);

create table DIM_Estado(
	SKestadoID int not null,
	estadoReservacion nvarchar(50),
	constraint estadoPK primary key(SKestadoID)
);

create table DIM_Tiempo(
	SKtiempoID int not null,
	fecha date,
	año smallint,
	mes nvarchar(50),
	numeroSemana tinyint,
	dia tinyint,
	trimestre tinyint,
	diaSemana varchar(20),
	constraint tiempoPK primary key(SKtiempoID)
);

create table DIM_Comida(
	SKcomidaID int not null,
	comida nvarchar(50),
	descripcion varchar(100),
	constraint comidaPK primary key(SKcomidaID)
);

create table DIM_Pais(
	SKpaisID int not null,
	pais nvarchar(50),
	nombre varchar(50),
	constraint paisPK primary key(SKpaisID)
);

create table DIM_Mercado(
	SKmercadoID int not null,
	segmentoMercado nvarchar(50),
	constraint mercadoPK primary key(SKmercadoID)
);

create table DIM_Canal(
	SKcanalID int not null,
	canalDistribucion nvarchar(50),
	constraint canalPK primary key(SKcanalID)
);
```

```sql
create table DIM_Repetido(
	SKrepetidoID int not null,
	repetido tinyint,
	descripcion varchar(30),
	constraint repetidoPK primary key(SKrepetidoID)
);

create table DIM_TipoHabitacion(
	SKtipohabitacionID int not null,
	tipoHabitacion nvarchar(50),
	constraint tipohabitacionPK primary key(SKtipohabitacionID)
);

create table DIM_Deposito(
	SKdepositoID int not null,
	tipoDeposito nvarchar(50),
	constraint depositoPK primary key(SKdepositoID)
);

create table DIM_TipoCliente(
	SKtipoclienteID int not null,
	tipoCliente nvarchar(50),
	descripcion nvarchar(50),
	constraint tipoClientePK primary key(SKtipoclienteID)
);

create table FACT_Reserva(
	SKreservaID bigint not null,
	tiempoLlegada smallint,
	nochesFinSemana tinyint,
	nochesSemana tinyint,
	nochesTotales tinyint,
	adultos tinyint,
	niños tinyint,
	bebes tinyint,
	personasTotales tinyint,
	cambiosReservacion tinyint,
	diasListaEspera smallint,
	tarifaPromedio float,
	ingresoTotal float,
	espaciosEstacionamiento tinyint,
	solicitudesEspeciales tinyint,
	SKhotelID int,
	SKestadoID int,
	SKfechallegadaID int,
	SKfechaestatusID int,
	SKcomidaID int,
	SKpaisID int,
	SKmercadoID int,
	SKcanalID int,
	SKrepetidoID int,
	SKtiporeservadoID int,
	SKtipoasignadoID int,
	SKdepositoID int,
	SKtipoclienteID int,
	constraint reservaPK primary key(SKreservaID),
```

```
	constraint reservaFK foreign key(SKhotelID) references
DIM_Hotel(SKhotelID),
	constraint reservaFK2 foreign key(SKestadoID) references
DIM_Estado(SKestadoID),
	constraint reservaFK3 foreign key(SKfechallegadaID) references
DIM_Tiempo(SKtiempoID),
	constraint reservaFK4 foreign key(SKfechaestatusID) references
DIM_Tiempo(SKtiempoID),
	constraint reservaFK5 foreign key(SKcomidaID) references
DIM_Comida(SKcomidaID),
	constraint reservaFK6 foreign key(SKpaisID) references DIM_Pais(SKpaisID),
	constraint reservaFK7 foreign key(SKmercadoID) references
DIM_Mercado(SKmercadoID),
	constraint reservaFK8 foreign key(SKcanalID) references
DIM_Canal(SKcanalID),
	constraint reservaFK9 foreign key(SKrepetidoID) references
DIM_Repetido(SKrepetidoID),
	constraint reservaFK10 foreign key(SKtiporeservadoID) references
DIM_TipoHabitacion(SKtipohabitacionID),
	constraint reservaFK11 foreign key(SKtipoasignadoID) references
DIM_TipoHabitacion(SKtipohabitacionID),
	constraint reservaFK12 foreign key(SKdepositoID) references
DIM_Deposito(SKdepositoID),
	constraint reservaFK13 foreign key(SKtipoclienteID) references
DIM_TipoCliente(SKtipoclienteID)
);
```

Base de Datos Intermedia (diagrama relacional y script de creación

Diagrama relacional

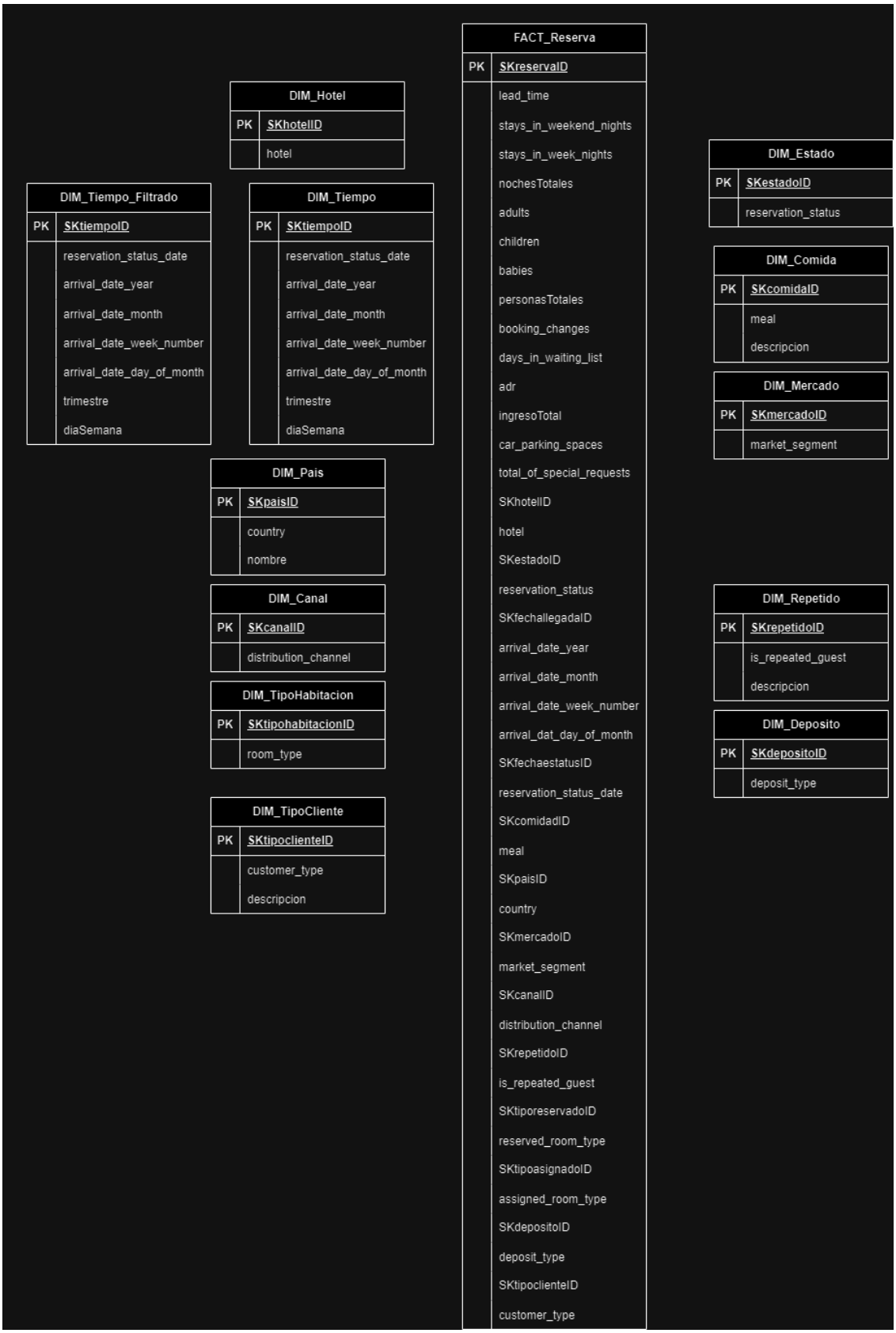

Figura 5.3.1. Diagrama relacional de la base de datos intermedia

Script

```
-- Crear base de datos
create database reservasStage;

-- Usar base de datos
use reservasStage;

--  Crear tablas
create table DIM_Hotel(
	SKhotelID int identity not null,
	hotel nvarchar(50),
	constraint hotelPK primary key(SKhotelID)
);

create table DIM_Estado(
	SKestadoID int identity not null,
	reservation_status nvarchar(50),
	constraint estadoPK primary key(SKestadoID)
);

create table DIM_Tiempo(
	SKtiempoID int identity not null,
	reservation_status_date date,
	arrival_date_year smallint,
	arrival_date_month nvarchar(50),
	arrival_date_week_number tinyint,
	arrival_date_day_of_month tinyint,
	trimestre tinyint,
	diaSemana varchar(20),
	constraint tiempoPK primary key(SKtiempoID)
);

create table DIM_Tiempo_Filtrado(
	SKtiempoID int identity not null,
	reservation_status_date date,
	arrival_date_year smallint,
	arrival_date_month nvarchar(50),
	arrival_date_week_number tinyint,
	arrival_date_day_of_month tinyint,
	trimestre tinyint,
	diaSemana varchar(20),
	constraint tiempoFPK primary key(SKtiempoID)
);

create table DIM_Comida(
	SKcomidaID int identity not null,
	meal nvarchar(50),
	descripcion varchar(100),
	constraint comidaPK primary key(SKcomidaID)
);

create table DIM_Pais(
	SKpaisID int identity not null,
	country nvarchar(50),
	nombre varchar(50),
	constraint paisPK primary key(SKpaisID)
);
```

```sql
create table DIM_Mercado(
	SKmercadoID int identity not null,
	market_segment nvarchar(50),
	constraint mercadoPK primary key(SKmercadoID)
);

create table DIM_Canal(
	SKcanalID int identity not null,
	distribution_channel nvarchar(50),
	constraint canalPK primary key(SKcanalID)
);

create table DIM_Repetido(
	SKrepetidoID int identity not null,
	is_repeated_guest tinyint,
	descripcion varchar(30),
	constraint repetidoPK primary key(SKrepetidoID)
);

create table DIM_TipoHabitacion(
	SKtipohabitacionID int identity not null,
	room_type nvarchar(50),
	constraint tipohabitacionPK primary key(SKtipohabitacionID)
);

create table DIM_Deposito(
	SKdepositoID int identity not null,
	deposit_type nvarchar(50),
	constraint depositoPK primary key(SKdepositoID)
);

create table DIM_TipoCliente(
	SKtipoclienteID int identity not null,
	customer_type nvarchar(50),
	descripcion nvarchar(50)
	constraint tipoClientePK primary key(SKtipoclienteID)
);

create table FACT_Reserva(
	SKreservaID bigint identity not null,
	lead_time smallint,
	stays_in_weekend_nights tinyint,
	stays_in_week_nights tinyint,
	nochesTotales tinyint,
	adults tinyint,
	children tinyint,
	babies tinyint,
	personasTotales tinyint,
	booking_changes tinyint,
	days_in_waiting_list smallint,
	adr float,
	ingresoTotal float,
	car_parking_spaces tinyint,
	total_of_special_requests tinyint,
	SKhotelID int,
	hotel nvarchar(50),
	SKestadoID int,
```

```
        reservation_status nvarchar(50),
        SKfechallegadaID int,
        arrival_date_year smallint,
        arrival_date_month nvarchar(50),
        arrival_date_week_number tinyint,
        arrival_date_day_of_month tinyint,
        SKfechaestatusID int,
        reservation_status_date date,
        SKcomidaID int,
        meal nvarchar(50),
        SKpaisID int,
        country nvarchar(50),
        SKmercadoID int,
        market_segment nvarchar(50),
        SKcanalID int,
        distribution_channel nvarchar(50),
        SKrepetidoID int,
        is_repeated_guest tinyint,
        SKtiporeservadoID int,
        reserved_room_type nvarchar(50),
        SKtipoasignadoID int,
        assigned_room_type nvarchar(50),
        SKdepositoID int,
        deposit_type nvarchar(50),
        SKtipoclienteID int,
        customer_type nvarchar(50),
        constraint reservaPK primary key(SKreservaID)
);
```

Proceso ETL

Extracción

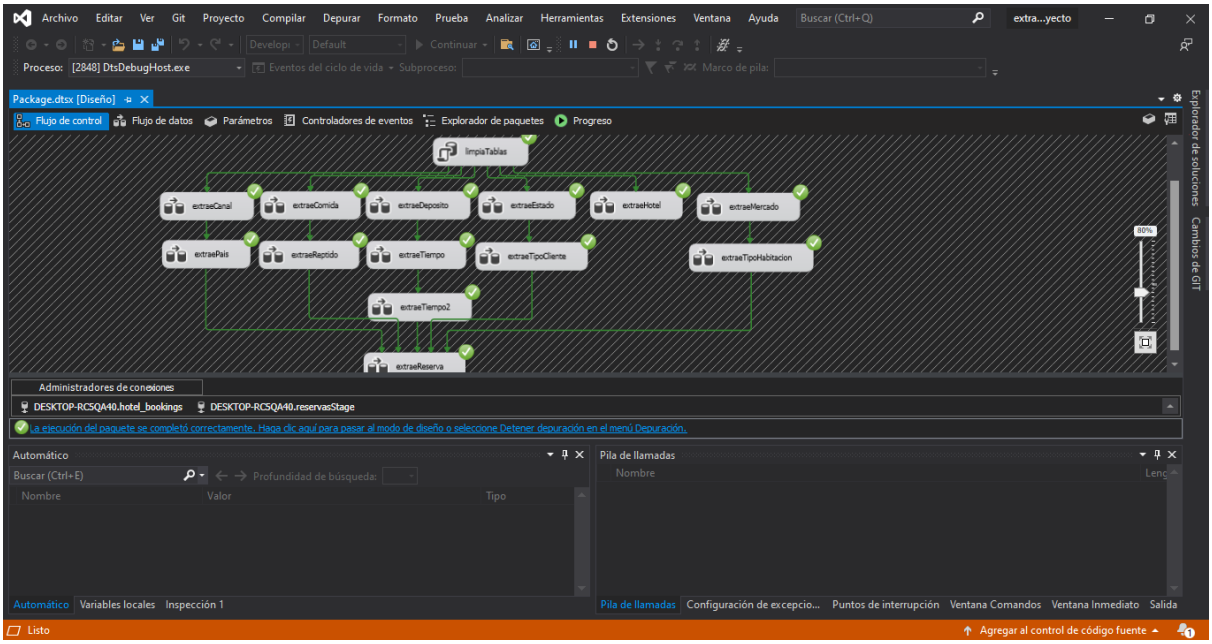

Figura 6.1.1. Captura del proceso exitoso de extracción de datos

Extracción – Script de procedimientos almacenados y vistas

```
-- Usar base de datos reservasStage
use reservasStage;

-- Crear procedimiento limpiaTablas
create procedure limpiaTablas
as
begin
		delete from FACT_Reserva;
		delete from DIM_Canal;
		delete from DIM_Comida;
		delete from DIM_Deposito;
		delete from DIM_Estado;
		delete from DIM_Hotel;
		delete from DIM_Mercado;
		delete from DIM_Pais;
		delete from DIM_Repetido;
		delete from DIM_Tiempo;
		delete from DIM_Tiempo_Filtrado;
		delete from DIM_TipoCliente;
		delete from DIM_TipoHabitacion;
		DBCC checkident('FACT_Reserva',reseed,0);
		DBCC checkident('DIM_Canal',reseed,0);
		DBCC checkident('DIM_Comida',reseed,0);
		DBCC checkident('DIM_Deposito',reseed,0);
		DBCC checkident('DIM_Estado',reseed,0);
		DBCC checkident('DIM_Hotel',reseed,0);
		DBCC checkident('DIM_Mercado',reseed,0);
		DBCC checkident('DIM_Pais',reseed,0);
		DBCC checkident('DIM_Repetido',reseed,0);
		DBCC checkident('DIM_Tiempo',reseed,0);
		DBCC checkident('DIM_Tiempo_Filtrado',reseed,0);
		DBCC checkident('DIM_TipoCliente',reseed,0);
		DBCC checkident('DIM_TipoHabitacion',reseed,0);
	end;

-- Ejecutar procedimiento limpiaTablas
exec limpiaTablas;

-- Crear vista para recuperar datos unicos de canal
create view rCanal as
	select distinct distribution_channel
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de Canal
select * from rCanal;

-- Crear vista para recuperar datos unicos de comida
create view rComida as
	select distinct meal
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de Comida
select * from rComida;

-- Crear vista para recuperar datos unicos de deposito
create view rDeposito as
	select distinct deposit_type
	from hotel_bookings.dbo.hotel_bookings;
```

```sql
-- Probar la vista de Deposito
select * from rDeposito;

-- Crear vista para recuperar datos unicos de estado
create view rEstado as
	select distinct reservation_status
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de  Estado
select * from rEstado;

-- Crear vista para recuperar datos unicos de hotel
create view rHotel as
	select distinct hotel
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de Hotel
select * from rHotel;

-- Crear vista para recuperar datos unicos de mercado
create view rMercado as
	select distinct market_segment
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de Mercado
select * from rMercado;

-- Crear vista para recuperar datos unicos de pais
create view rPais as
	select distinct country
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista dePais
select * from rPais;

-- Crear vista para recuperar datos unicos de repetido
create view rRepetido as
	select distinct is_repeated_guest
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de Tiempo
select * from rRepetido;

-- Crear vistas para recuperar datos unicos de tiempo
create view rTiempo as
	select distinct
arrival_date_year,arrival_date_month,arrival_date_week_number,arrival_date_day_of_
month
	from hotel_bookings.dbo.hotel_bookings;

create view rTiempo2 as
	select distinct reservation_status_date
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vistas de Tiempo
select * from rTiempo;
select * from rTiempo2;

-- Crear vista para recuperar datos unicos de tipo de cliente
create view rTipoCliente as
	select distinct customer_type
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de tipo de cliente
select * from rTipoCliente;
```

```sql
-- Crear vista para recuperar datos unicos de tipo de habitacion
create view rTipoHabitacion as
	select distinct assigned_room_type
	from hotel_bookings.dbo.hotel_bookings;
-- Probar la vista de tipo de habitacion
select * from rTipoHabitacion;
```

Transformación

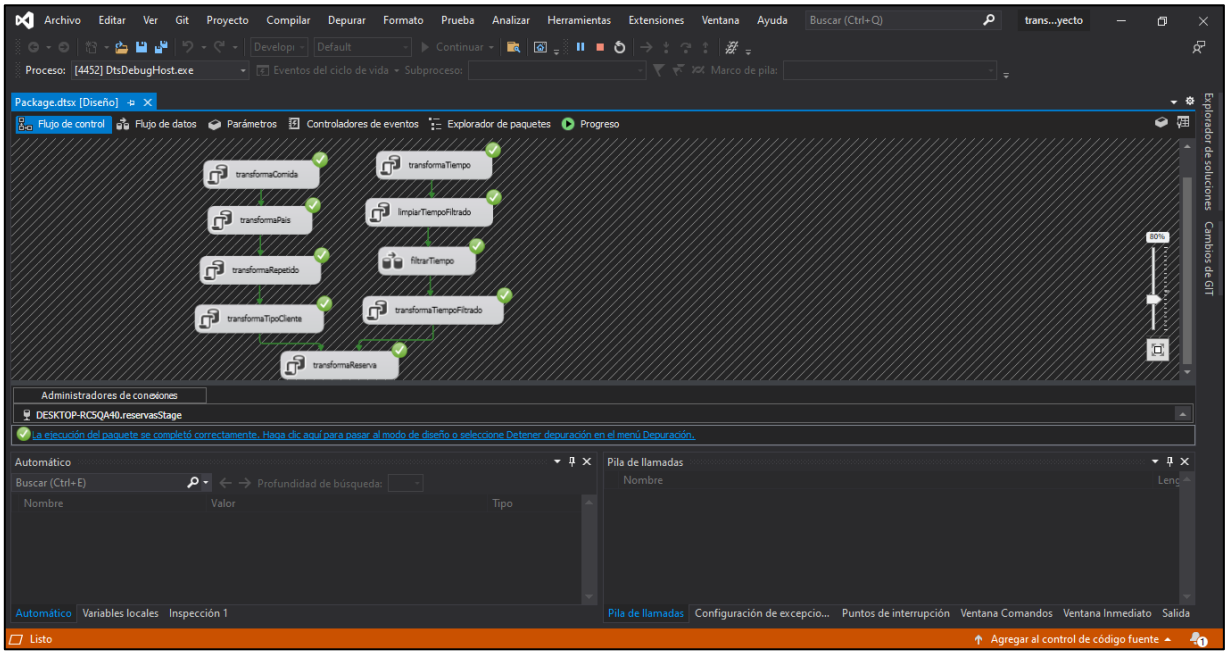

Figura 6.2.1. Captura del proceso exitoso de transformación de datos

Transformación – Script de procedimientos almacenados y vistas

```sql
-- Usar base de datos ventasStage
use reservasStage;

-- Verificar tabla de DIM_Canal
select * from DIM_Canal;

-- Verificar tabla de DIM_Comida
select * from DIM_Comida;
-- Crear procedimiento almacenado para transformar DIM_Comida(Generar columna de
descripcion)
create procedure transformaComida
as begin
	UPDATE DIM_Comida
	SET descripcion = CASE meal
		WHEN 'FB' THEN 'Full Board (desayuno, comida y cena)'
		WHEN 'HB' THEN 'Half Board (desayuno y una comida)'
		WHEN 'SC' THEN 'Self Catering (sin alimentos incluidos)'
		WHEN 'BB' THEN 'Bed and Breakfast (desayuno incluido)'
		WHEN 'Undefined' THEN 'Indefinido'
		ELSE meal
```

```
        end
        where descripcion is null
end;

-- Verificar tabla de DIM_Deposito
select * from DIM_Deposito;
-- Verificar tabla de DIM_Estado
select * from DIM_Estado;
-- Verificar tabla de DIM_Hotel
select * from DIM_Hotel;
-- Verificar tabla de DIM_Mercado
select * from DIM_Mercado;

-- Verificar tabla de DIM_Pais
select * from DIM_Pais;
-- Crear procedimiento almacenado para transformar DIM_Pais(Generar columna de
nombre)
create procedure transformaPais
as begin
        UPDATE DIM_Pais
        SET nombre = CASE country
            WHEN 'ISR' THEN 'Israel'
               WHEN 'BOL' THEN 'Bolivia'
               WHEN 'CHN' THEN 'China'
               WHEN 'TGO' THEN 'Togo'
               WHEN 'SYR' THEN 'Siria'
               WHEN 'CHL' THEN 'Chile'
               WHEN 'SLE' THEN 'Sierra Leona'
               WHEN 'COL' THEN 'Colombia'
               WHEN 'AND' THEN 'Andorra'
               WHEN 'JPN' THEN 'Japón'
               WHEN 'GUY' THEN 'Guyana'
               WHEN 'IRQ' THEN 'Irak'
               WHEN 'SYC' THEN 'Seychelles'
               WHEN 'NZL' THEN 'Nueva Zelanda'
               WHEN 'PLW' THEN 'Palaos'
               WHEN 'UZB' THEN 'Uzbekistán'
               WHEN 'USA' THEN 'Estados Unidos'
               WHEN 'ARG' THEN 'Argentina'
               WHEN 'KNA' THEN 'San Cristóbal y Nieves'
               WHEN 'PRT' THEN 'Portugal'
               WHEN 'AUS' THEN 'Australia'
               WHEN 'MEX' THEN 'México'
               WHEN 'SVN' THEN 'Eslovenia'
               WHEN 'GIB' THEN 'Gibraltar'
               WHEN 'ATA' THEN 'Antártida'
               WHEN 'ARE' THEN 'Emiratos Árabes Unidos'
               WHEN 'HRV' THEN 'Croacia'
               WHEN 'DEU' THEN 'Alemania'
               WHEN 'VEN' THEN 'Venezuela'
               WHEN 'NLD' THEN 'Países Bajos'
               WHEN 'HKG' THEN 'Hong Kong'
               WHEN 'SVK' THEN 'Eslovaquia'
               WHEN 'CHE' THEN 'Suiza'
               WHEN 'PRI' THEN 'Puerto Rico'
               WHEN 'KAZ' THEN 'Kazajistán'
               WHEN 'LBN' THEN 'Líbano'
               WHEN 'MYS' THEN 'Malasia'
```

```
            WHEN 'IDN' THEN 'Indonesia'
            WHEN 'BDI' THEN 'Burundi'
            WHEN 'CZE' THEN 'República Checa'
            WHEN 'NAM' THEN 'Namibia'
            WHEN 'ARM' THEN 'Armenia'
            WHEN 'NIC' THEN 'Nicaragua'
            WHEN 'GAB' THEN 'Gabón'
            WHEN 'ITA' THEN 'Italia'
            WHEN 'URY' THEN 'Uruguay'
            WHEN 'KOR' THEN 'Corea del Sur'
            WHEN 'ZWE' THEN 'Zimbabue'
            WHEN 'DJI' THEN 'Yibuti'
            WHEN 'SAU' THEN 'Arabia Saudita'
         WHEN 'CRI' THEN 'Costa Rica'
         WHEN 'VNM' THEN 'Vietnam'
            WHEN 'STP' THEN 'Santo Tomé y Príncipe'
            WHEN 'PYF' THEN 'Polinesia Francesa'
            WHEN 'TUR' THEN 'Turquía'
            WHEN 'NULL' THEN 'Desconocido'
            WHEN 'MRT' THEN 'Mauritania'
            WHEN 'DMA' THEN 'Dominica'
            WHEN 'ALB' THEN 'Albania'
            WHEN 'LCA' THEN 'Santa Lucía'
            WHEN 'FRO' THEN 'Islas Feroe'
            WHEN 'BEN' THEN 'Benín'
            WHEN 'MLI' THEN 'Malí'
            WHEN 'IRL' THEN 'Irlanda'
            WHEN 'POL' THEN 'Polonia'
            WHEN 'NGA' THEN 'Nigeria'
            WHEN 'MKD' THEN 'Macedonia del Norte'
            WHEN 'AUT' THEN 'Austria'
            WHEN 'JEY' THEN 'Jersey'
            WHEN 'CUB' THEN 'Cuba'
            WHEN 'IMN' THEN 'Isla de Man'
            WHEN 'TZA' THEN 'Tanzania'
            WHEN 'KEN' THEN 'Kenia'
            WHEN 'BEL' THEN 'Bélgica'
            WHEN 'BRA' THEN 'Brasil'
            WHEN 'BFA' THEN 'Burkina Faso'
            WHEN 'MAC' THEN 'Macao'
            WHEN 'FIN' THEN 'Finlandia'
            WHEN 'MMR' THEN 'Birmania (Myanmar)'
            WHEN 'FJI' THEN 'Fiyi'
            WHEN 'GEO' THEN 'Georgia'
            WHEN 'MAR' THEN 'Marruecos'
            WHEN 'GGY' THEN 'Guernsey'
            WHEN 'CYM' THEN 'Islas Caimán'
            WHEN 'GHA' THEN 'Ghana'
            WHEN 'MYT' THEN 'Mayotte'
            WHEN 'LTU' THEN 'Lituania'
            WHEN 'EGY' THEN 'Egipto'
            WHEN 'MDV' THEN 'Maldivas'
            WHEN 'MNE' THEN 'Montenegro'
            WHEN 'PAK' THEN 'Pakistán'
            WHEN 'ATF' THEN 'Territorios Australes Franceses'
            WHEN 'DNK' THEN 'Dinamarca'
            WHEN 'ZAF' THEN 'Sudáfrica'
            WHEN 'CIV' THEN 'Costa de Marfil'
```

```
            WHEN 'DZA' THEN 'Argelia'
            WHEN 'BGR' THEN 'Bulgaria'
            WHEN 'RWA' THEN 'Ruanda'
            WHEN 'MDG' THEN 'Madagascar'
            WHEN 'BHR' THEN 'Baréin'
            WHEN 'SGP' THEN 'Singapur'
            WHEN 'RUS' THEN 'Rusia'
            WHEN 'PRY' THEN 'Paraguay'
            WHEN 'GRC' THEN 'Grecia'
            WHEN 'ROU' THEN 'Rumania'
         WHEN 'OMN' THEN 'Omán'
         WHEN 'HND' THEN 'Honduras'
            WHEN 'AIA' THEN 'Anguila'
            WHEN 'LUX' THEN 'Luxemburgo'
            WHEN 'THA' THEN 'Tailandia'
            WHEN 'JOR' THEN 'Jordania'
            WHEN 'CAF' THEN 'República Centroafricana'
            WHEN 'LIE' THEN 'Liechtenstein'
            WHEN 'BWA' THEN 'Botsuana'
            WHEN 'SRB' THEN 'Serbia'
            WHEN 'TUN' THEN 'Túnez'
            WHEN 'SWE' THEN 'Suecia'
            WHEN 'SEN' THEN 'Senegal'
            WHEN 'VGB' THEN 'Islas Vírgenes Británicas'
            WHEN 'TWN' THEN 'Taiwán'
            WHEN 'MLT' THEN 'Malta'
            WHEN 'SUR' THEN 'Surinam'
            WHEN 'TMP' THEN 'Timor Oriental'
            WHEN 'CMR' THEN 'Camerún'
            WHEN 'SDN' THEN 'Sudán'
            WHEN 'COM' THEN 'Comoras'
            WHEN 'MWI' THEN 'Malaui'
            WHEN 'BLR' THEN 'Bielorrusia'
            WHEN 'EST' THEN 'Estonia'
            WHEN 'UGA' THEN 'Uganda'
            WHEN 'SMR' THEN 'San Marino'
         WHEN 'KWT' THEN 'Kuwait'
         WHEN 'FRA' THEN 'Francia'
            WHEN 'DOM' THEN 'República Dominicana'
            WHEN 'PAN' THEN 'Panamá'
            WHEN 'NCL' THEN 'Nueva Caledonia'
            WHEN 'IRN' THEN 'Irán'
            WHEN 'GLP' THEN 'Guadalupe'
            WHEN 'IND' THEN 'India'
            WHEN 'PHL' THEN 'Filipinas'
            WHEN 'MOZ' THEN 'Mozambique'
            WHEN 'ISL' THEN 'Islandia'
            WHEN 'SLV' THEN 'El Salvador'
            WHEN 'ESP' THEN 'España'
            WHEN 'LVA' THEN 'Letonia'
            WHEN 'GBR' THEN 'Reino Unido'
            WHEN 'AZE' THEN 'Azerbaiyán'
            WHEN 'CYP' THEN 'Chipre'
            WHEN 'GNB' THEN 'Guinea-Bisáu'
            WHEN 'LKA' THEN 'Sri Lanka'
            WHEN 'UMI' THEN 'Islas Ultramarinas Menores de Estados Unidos'
            WHEN 'ABW' THEN 'Aruba'
            WHEN 'ECU' THEN 'Ecuador'
```

```
                WHEN 'LAO' THEN 'Laos'
                WHEN 'PER' THEN 'Perú'
                WHEN 'ETH' THEN 'Etiopía'
                WHEN 'QAT' THEN 'Catar'
                WHEN 'MCO' THEN 'Mónaco'
                WHEN 'KHM' THEN 'Camboya'
                WHEN 'AGO' THEN 'Angola'
                WHEN 'CPV' THEN 'Cabo Verde'
                WHEN 'HUN' THEN 'Hungría'
                WHEN 'GTM' THEN 'Guatemala'
                WHEN 'BIH' THEN 'Bosnia y Herzegovina'
                WHEN 'UKR' THEN 'Ucrania'
                WHEN 'JAM' THEN 'Jamaica'
                WHEN 'KIR' THEN 'Kiribati'
                WHEN 'NPL' THEN 'Nepal'
                WHEN 'LBY' THEN 'Libia'
                WHEN 'BRB' THEN 'Barbados'
                WHEN 'TJK' THEN 'Tayikistán'
                WHEN 'BTN' THEN 'Bután'
                WHEN 'NOR' THEN 'Noruega'
                WHEN 'MUS' THEN 'Mauricio'
                WHEN 'BHS' THEN 'Bahamas'
                WHEN 'BGD' THEN 'Bangladesh'
                WHEN 'ZMB' THEN 'Zambia'
                WHEN 'ASM' THEN 'Samoa Americana'
                WHEN 'CN' THEN 'China'
                ELSE nombre
        end
        where nombre is null
end;

-- Verificar tabla de DIM_Repetido
select * from DIM_Repetido;
-- Crear procedimiento almacenado para transformar DIM_Repetido(Generar columna de
descripcion)
create procedure transformaRepetido
as begin
        UPDATE DIM_Repetido
        SET descripcion = CASE is_repeated_guest
                WHEN 0 THEN 'Cliente nuevo'
                WHEN 1 THEN 'Cliente recurrente'
                ELSE descripcion
        end
        where descripcion is null
end;

-- Verificar tabla de DIM_Tiempo
select * from DIM_Tiempo;
-- Crear procedimiento almacenado para transformar DIM_Tiempo(Generar datos
faltantes en la columna correspondiente a fecha)
create procedure transformaTiempo
as begin
        UPDATE DIM_Tiempo
        SET reservation_status_date=DATEFROMPARTS(
                                                                arrival_date_year,
                                                                CASE
                                                                        WHEN
arrival_date_month='January' THEN 1
```

```
                                                                WHEN
arrival_date_month='February' THEN 2
                                                                WHEN
arrival_date_month='March' THEN 3
                                                                WHEN
arrival_date_month='April' THEN 4
                                                                WHEN
arrival_date_month='May' THEN 5
                                                                WHEN
arrival_date_month='June' THEN 6
                                                                WHEN
arrival_date_month='July' THEN 7
                                                                WHEN
arrival_date_month='August' THEN 8
                                                                WHEN
arrival_date_month='September' THEN 9
                                                                WHEN
arrival_date_month='October' THEN 10
                                                                WHEN
arrival_date_month='November' THEN 11
                                                                WHEN
arrival_date_month='December' THEN 12
                                                                ELSE 0
                                                        END,

        arrival_date_day_of_month
                                                        )
        where reservation_status_date is null
end;
-- Limpiar Tabla de DIM_Tiempo_Filtrado
create procedure limpiarTiempoFiltrado
as begin
        delete from  DIM_Tiempo_Filtrado;
        DBCC checkident('DIM_Tiempo_Filtrado',reseed,0);
end;
-- Crear vista para filtrar fechas unicas y pasarlas de DIM_Tiempo a
DIM_Tiempo_Filtrado
create view rTiempoFiltrado as
        select reservation_status_date,
                        MIN(arrival_date_year) as arrival_date_year,
                        MIN(arrival_date_month) as arrival_date_month,
                        MIN(arrival_date_day_of_month) as arrival_date_day_of_month,
                        MIN(arrival_date_week_number) as arrival_date_week_number
        from DIM_Tiempo
        group BY reservation_status_date;

-- Verificar tabla de DIM_Tiempo
select * from DIM_Tiempo_Filtrado;
-- Crear procedimiento almacenado para transformar DIM_Tiempo_Filtrado(Generar
datos faltantes en todas las columnas)
create procedure transformaTiempoFiltrado
as begin
        update DIM_Tiempo_Filtrado
        SET arrival_date_year=YEAR(reservation_status_date)
        where arrival_date_year is null;

        UPDATE DIM_Tiempo_Filtrado
        SET arrival_date_month = FORMAT(reservation_status_date, 'MMMM', 'en-US')
```

```
        where arrival_date_month is null ;

        update DIM_Tiempo_Filtrado
        SET arrival_date_week_number=DATENAME(WEEK,reservation_status_date)
        where arrival_date_week_number is null;

        update DIM_Tiempo_Filtrado
        SET    arrival_date_day_of_month=DAY(reservation_status_date)
        where arrival_date_day_of_month is null;

        update DIM_Tiempo_Filtrado
        SET    trimestre=(MONTH(reservation_status_date)+2)/3
        where trimestre is null;

        update DIM_Tiempo_Filtrado
        SET    diaSemana=DATENAME(WEEKDAY,reservation_status_date)
        where diaSemana is null;
end;
-- Verificar tabla de DIM_TipoCliente
select * from DIM_TipoCliente;
-- Crear procedimiento almacenado para transformar DIM_TipoCliente(Generar columna
de descripcion)
create procedure transformaTipoCliente
as begin
        UPDATE DIM_TipoCliente
        SET descripcion = CASE customer_type
                WHEN 'Group' THEN 'Reservas Grupales'
                WHEN 'Contract' THEN 'Clientes bajo contrato'
                WHEN 'Transient' THEN 'Reservas independientes individuales'
                WHEN 'Transient-Party' THEN 'Combinación de transitorios y grupos
pequeños'
                ELSE descripcion
        end
        where descripcion is null
end;

-- Verificar tabla de DIM_TipoHabitacion
select * from DIM_TipoHabitacion;

-- Verificar tabla de FACT_Resrva
select * from FACT_Reserva;

-- Crear procedimiento almacenado para transformar FACT_Reserva(Generar datos
faltantes en todas las columnas)
create procedure transformaReserva
as begin
        update FACT_Reserva
        SET nochesTotales = stays_in_week_nights+stays_in_weekend_nights
        where nochesTotales is null;

        update FACT_Reserva
        SET personasTotales=adults+children+babies
        where personasTotales is null;

        update FACT_Reserva
        SET ingresoTotal=nochesTotales*adr
        where ingresoTotal is null;
```

```
	update FACT_Reserva
	SET SKhotelID=(select SKhotelID from DIM_Hotel
                                                    where
hotel=FACT_Reserva.hotel)
	where SKhotelID is null;

	update FACT_Reserva
	SET SKestadoID=(select SKestadoID from DIM_Estado
                                                    where
reservation_status=FACT_Reserva.reservation_status)
	where SKestadoID is null;

	update FACT_Reserva
	SET SKfechallegadaID=(select SKtiempoID from DIM_Tiempo_Filtrado
                                                    where
arrival_date_year=FACT_Reserva.arrival_date_year
                                                    AND
arrival_date_month=FACT_Reserva.arrival_date_month
                                                    AND
arrival_date_day_of_month=FACT_Reserva.arrival_date_day_of_month)
	where SKfechallegadaID is null;

	update FACT_Reserva
	SET SKfechaestatusID=(select SKtiempoID from DIM_Tiempo_Filtrado
                                                    where
reservation_status_date=FACT_Reserva.reservation_status_date)
	where SKfechaestatusID is null;

	update FACT_Reserva
	SET SKcomidaID=(select SKcomidaID from DIM_Comida
                                                    where
meal=FACT_Reserva.meal)
	where SKcomidaID is null;

	update FACT_Reserva
	SET SKpaisID=(select SKpaisID from DIM_Pais
                                                    where
country=FACT_Reserva.country)
	where SKpaisID is null;

	update FACT_Reserva
	SET SKmercadoID=(select SKmercadoID from DIM_Mercado
                                                    where
market_segment=FACT_Reserva.market_segment)
	where SKmercadoID is null;

	update FACT_Reserva
	SET SKcanalID=(select SKcanalID from DIM_Canal
                                                    where
distribution_channel=FACT_Reserva.distribution_channel)
	where SKcanalID is null;

	update FACT_Reserva
	SET SKrepetidoID=(select SKrepetidoID from DIM_Repetido
                                                    where
is_repeated_guest=FACT_Reserva.is_repeated_guest)
	where SKrepetidoID is null;
```

```
        update FACT_Reserva
        SET SKtiporeservadoID=(select SKtipohabitacionID from DIM_TipoHabitacion
                                                        where
room_type=FACT_Reserva.reserved_room_type)
        where SKtiporeservadoID is null;

        update FACT_Reserva
        SET SKtipoasignadoID=(select SKtipohabitacionID from DIM_TipoHabitacion
                                                        where
room_type=FACT_Reserva.assigned_room_type)
        where SKtipoasignadoID is null;

        update FACT_Reserva
        SET SKdepositoID=(select SKdepositoID from DIM_Deposito
                                                        where
deposit_type=FACT_Reserva.deposit_type)
        where SKdepositoID is null;

        update FACT_Reserva
        SET SKtipoclienteID=(select SKtipoclienteID from DIM_TipoCliente
                                                        where
customer_type=FACT_Reserva.customer_type)
        where SKtipoclienteID is null;

end;
```

Carga

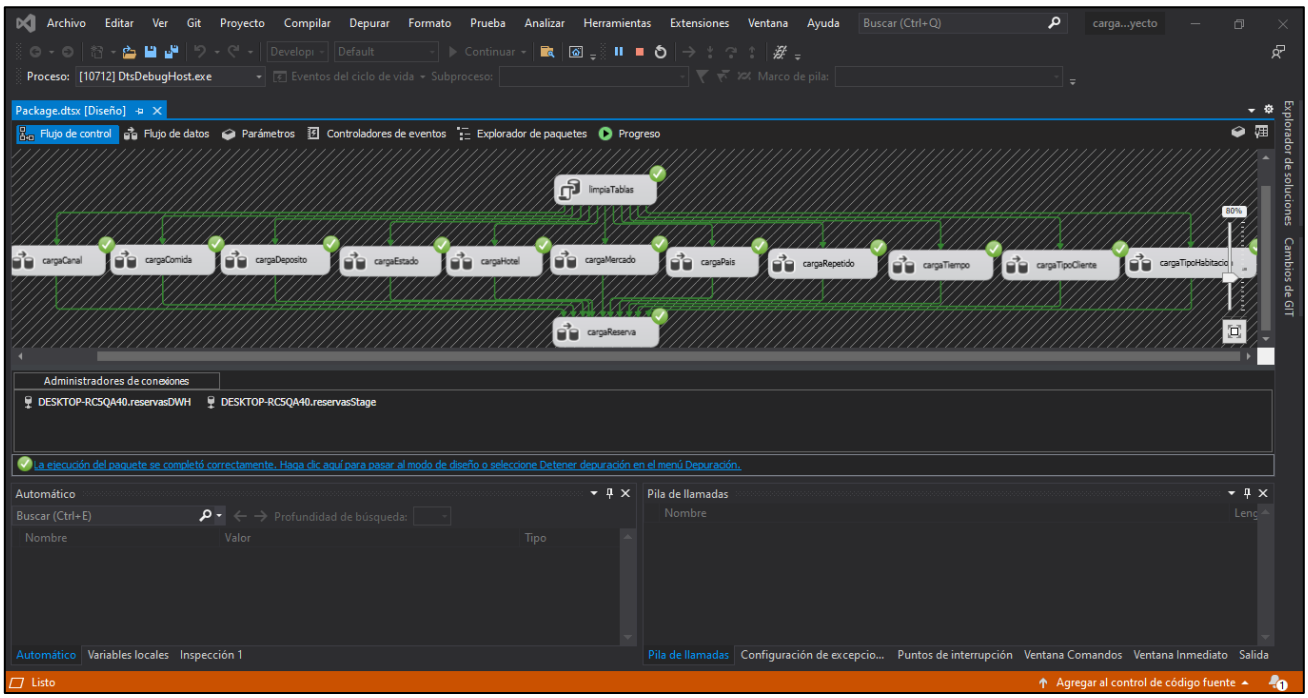

Figura 6.3.1. Captura del proceso exitoso de carga de datos

Carga – Script de procedimientos almacenados y vistas

```
-- Usar base de datos ventasDWH
use reservasDWH;

-- Crear procedimiento limpiaTablas
create procedure limpiaTablas
as
begin
            delete from FACT_Reserva;
            delete from DIM_Canal;
            delete from DIM_Comida;
            delete from DIM_Deposito;
            delete from DIM_Estado;
            delete from DIM_Hotel;
            delete from DIM_Mercado;
            delete from DIM_Pais;
            delete from DIM_Repetido;
            delete from DIM_Tiempo;
            delete from DIM_TipoCliente;
            delete from DIM_TipoHabitacion;
      end;
-- Ejecutar procedimiento limpiaTablas
exec limpiaTablas;
```

OLAP

Cubo

El cubo OLAP fue realizado por medio de los servicios de integración de datos de Microsoft, utilizados por medio de la plataforma Visual Studio 2019.

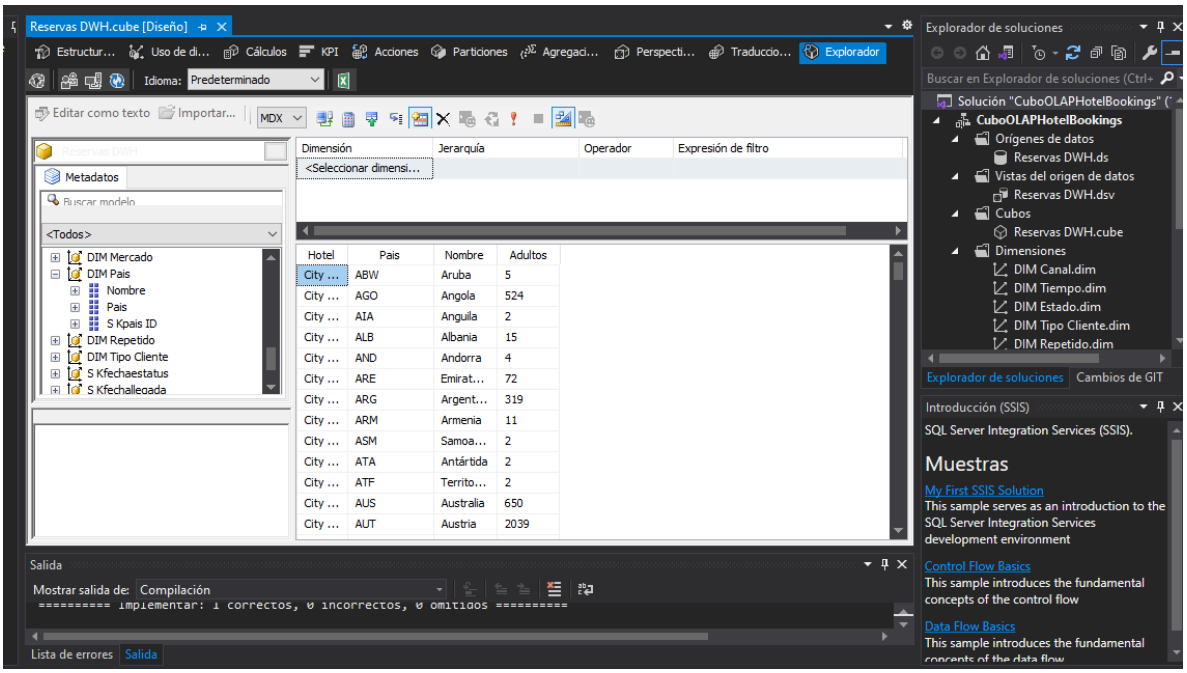

Figura 7.1.1. Captura de exposición de correcta implementación del cubo OLAP, con una consulta de prueba

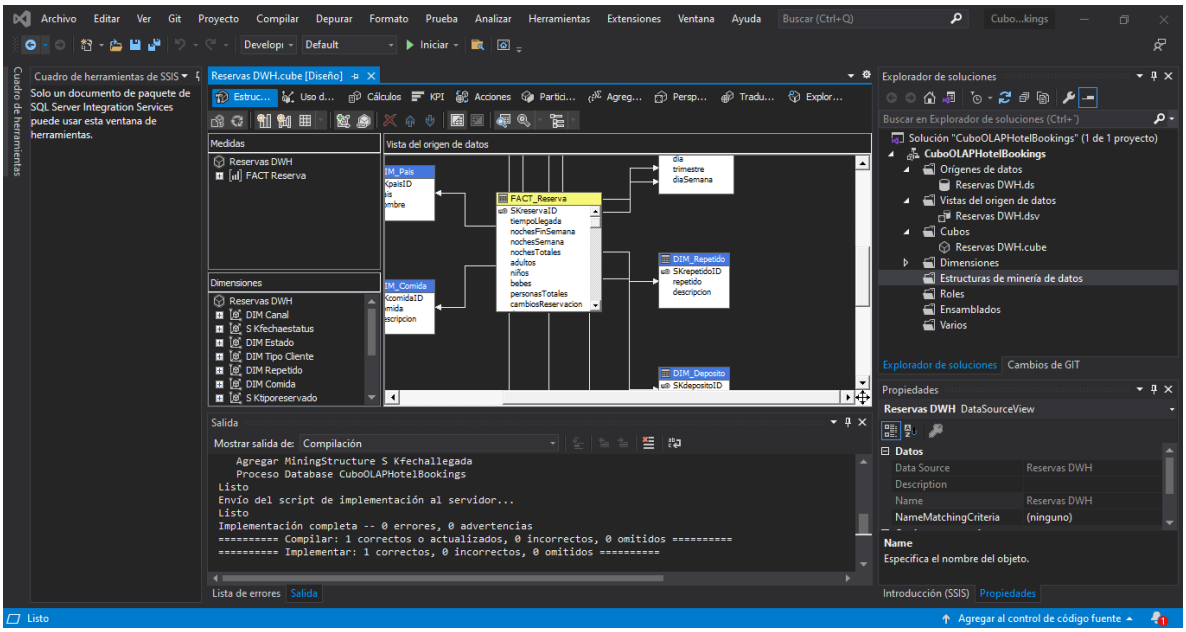

Figura 7.1.2. Captura de exposición de diagrama de la implementación del cubo OLAP

Gráficas

El Dashboard fue realizado por medio del software Excel, debido a su versatilidad y libertad de configuraciones, este está dividido en 4 cuadrantes que clasifican la información en cuatro categorías conforme a la naturaliza de sus dimensiones correspondientes, siendo estas las de ganancias, reservaciones, reservaciones días y reservaciones personas.

Este Dashboard esta segmentado por cada dimensión del Data Warehouse, donde los segmentadores se encuentran en una hoja distinta al Dashboard, con el fin de no aumentar el ruido visual con las gráficas. Debido a las diferencias en las naturalezas de las dimensiones y medidas, no todos los segmentadores afectan a todas las gráficas y leyendas, si no a las correspondientes a sus dimensiones relacionadas.

El Dashboard se presenta en la Figura 8.1.1.

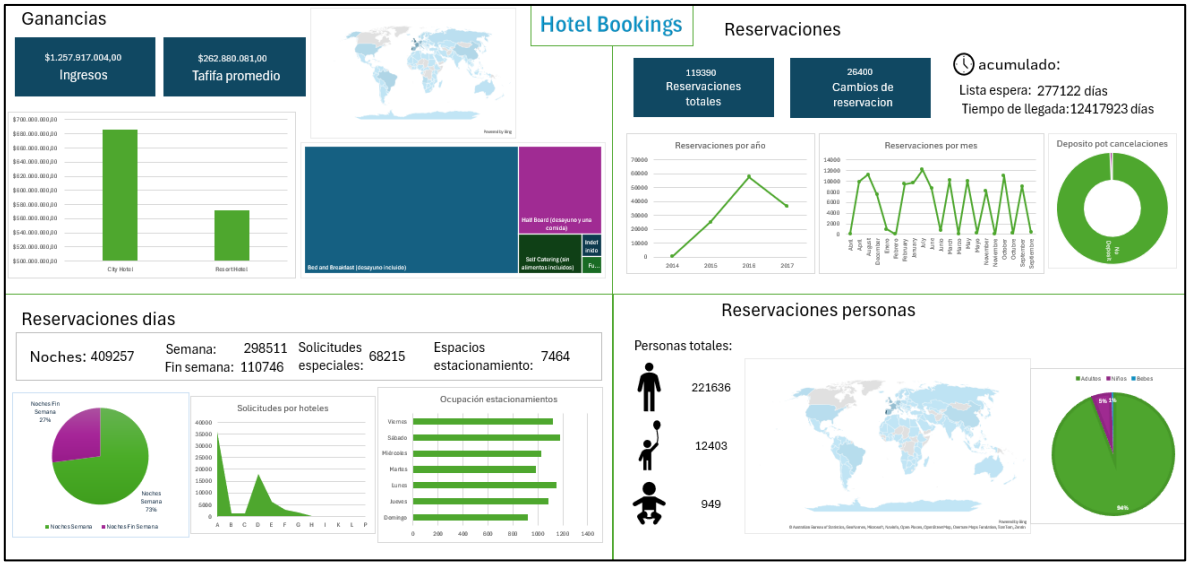

Figura 8.1.1. Dashboard

Caso de uso

Como caso de uso se muestra el insólito caso de un par de reservaciones en la Antártida, donde se muestra que estas fueron realizadas por dos adultos entre el 2016 y 2017.

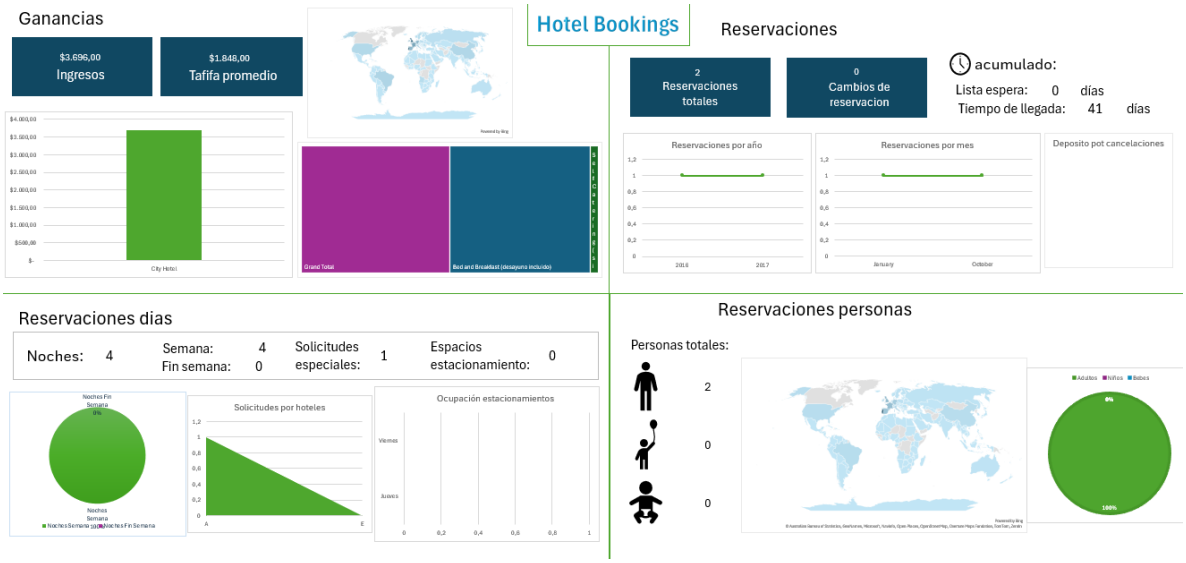

Figura 8.2.1. Caso de uso de país Antártida

De igual manera se muestran los datos relacionados al hotel City Hotel.

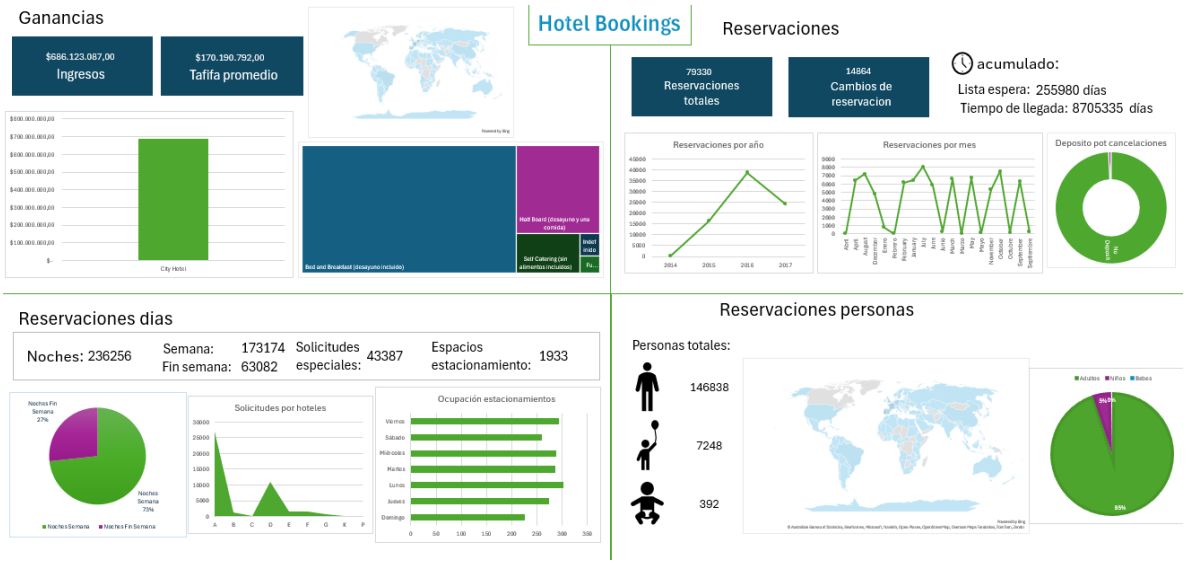

Figura 8.2.2. Caso de uso de hotel City Hotel

Conclusiones

El desarrollo de un cubo OLAP sobre un data WareHouse proporciona una herramienta bastante útil para el análisis de grandes cantidades de datos, lo que permite la exploración de grandes volúmenes de información en cortos periodos de tiempo. De esta manera aplicando el cubo OLAP a nuestro caso de estudio se puede obtener información descriptiva que permita brindarle un contexto mas completo a la administración de los hoteles mediante la presentación de información de manera visual a través de un Dashboard para la toma de decisiones estrategias.

Minería de Datos

Algoritmo Aplicado.

Explicación: Se ha elegido el algoritmo de árboles de decisión debido a su uso eficaz en un contexto de toma de decisiones y predicción de acciones como el de este caso, así como su manejo de datos numéricos para la toma de decisiones en cada ramificación y su fácil interpretación, que permite identificación de interacciones previas al resultado final, que se plantea sea una predicción de un valor cuantitativo con respecto a factores determinados.

Los árboles de decisión son una técnica de minería de datos utilizada para la clasificación y la regresión. Se representan como un modelo en forma de árbol, donde cada nodo interno representa una prueba en un atributo, cada rama representa el resultado de la prueba, y cada hoja representa una clase de resultado o un valor, dependiendo de la finalidad del algoritmo.

En este contexto el algoritmo divide el conjunto de datos en subconjuntos más pequeños basándose en el valor de los atributos. Esta división se realiza de manera recursiva hasta que se cumplen ciertos criterios de parada, como la pureza del nodo

(todas las instancias pertenecen a la misma clase) o un número mínimo de instancias en el nodo.

Objetivo: comprender el comportamiento de los ingresos mediante la identificación de patrones y tendencias en los ingresos de las reservas de los hoteles a través de los meses y trimestres para realizar una toma de acciones y acciones.

Para lo cual se establece el uso de los atributos mes y trimestre de la dimensión Tiempo llegada y el atributo ingresos totales de la tabla de hechos Reservas.

Aplicación: para la implementación del algoritmo de minería de datos de árboles de decisión empleando el cubo OLAP se empleó los servicios de análisis de datos de Microsoft mediante el empleo de Visual Studio. Por lo cual, se creó una estructura de minería de datos empleando el asistente de minería de datos de Visual Studio. Para empezar, se definió que se emplearía un cubo OLAP y el algoritmo de árboles de decisión.

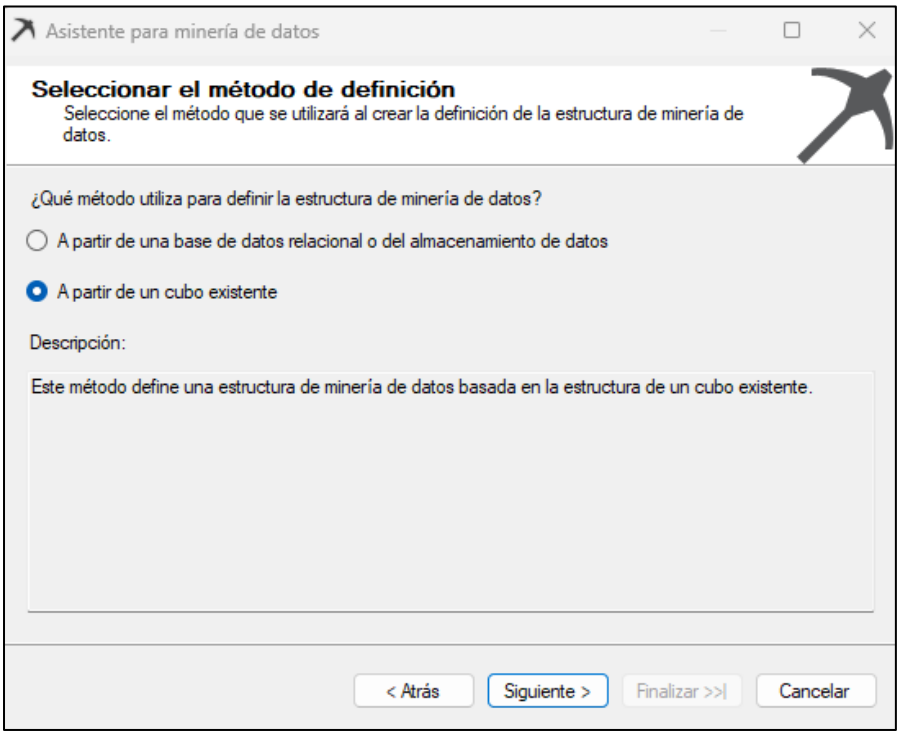

Figura 9.1.1. Selección de método de definición del modelo

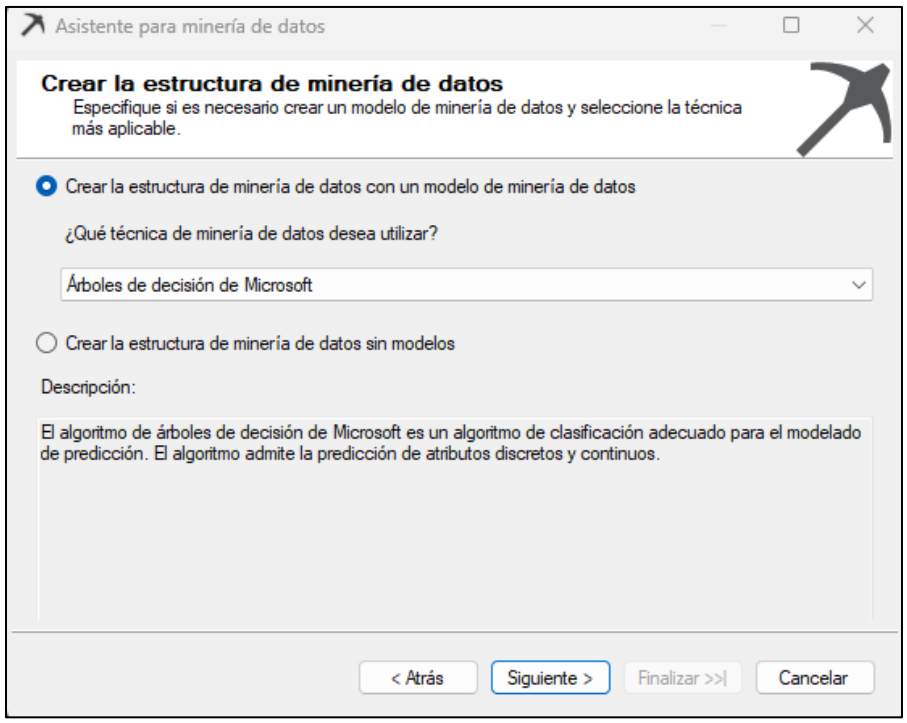

Figura 9.1.2. Creación de estructura de minería de datos y elección de algoritmo

Después se definió una dimensión de cubo de origen para la extracción de información. Para este caso se eligió la dimensión tiempo mediante el atributo correspondiente a fecha de llegada.

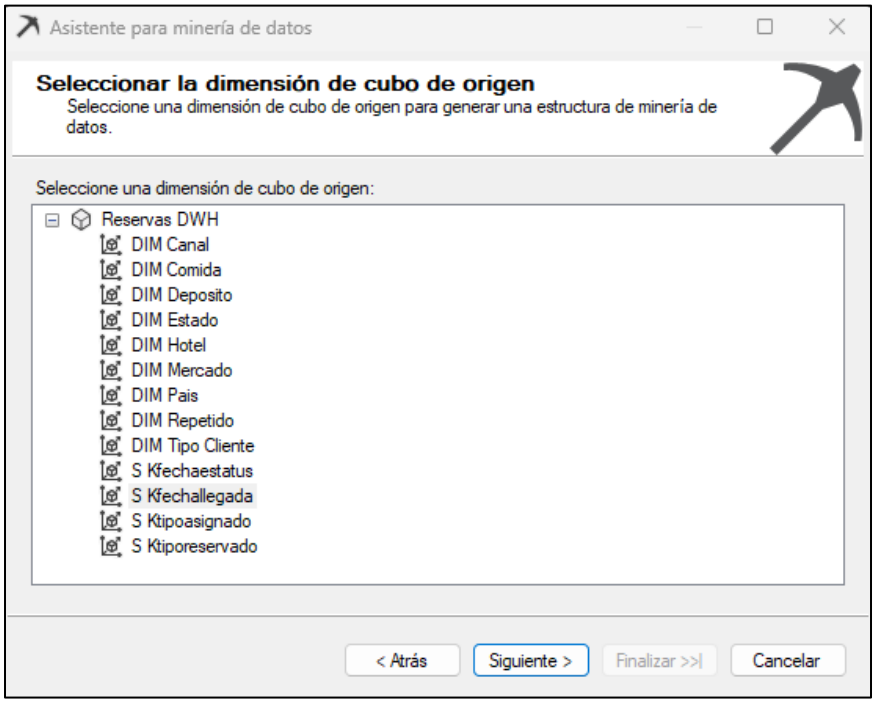

Figura 9.1.3. Selección de vista del cubo OLAP

Posteriormente se definió como clave de caso el atributo correspondiente a la llave primaria de la dimensión tiempo: SKtiempoID.

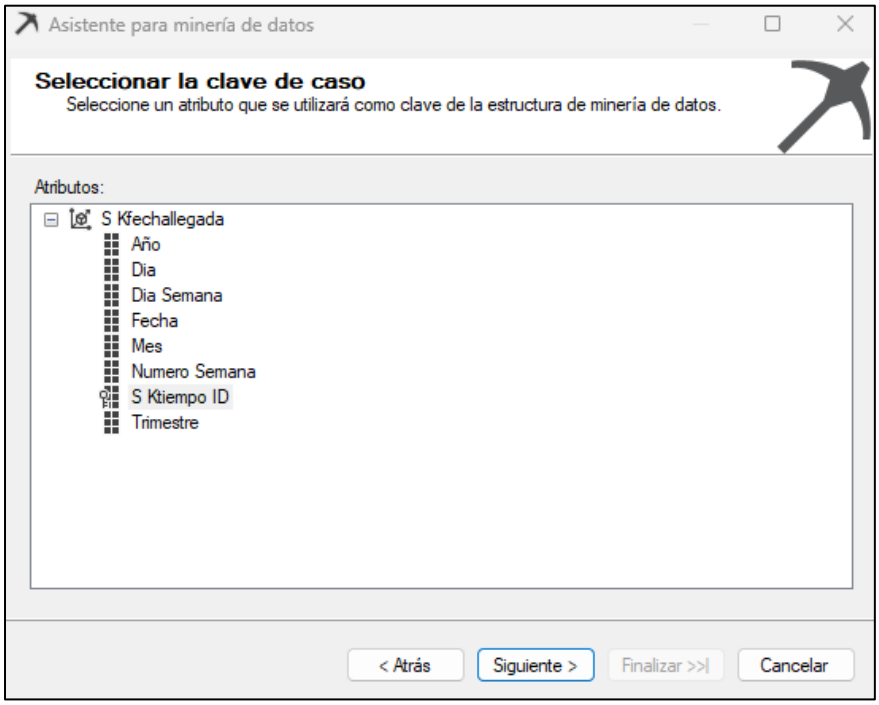

Figura 9.1.4. Selección de dimensión

Para finalizar la implementación del algoritmo de minería de datos se definieron las columnas que se emplearan para aplicar el algoritmo de minería de datos. Para este caso, que se plantea obtener el comportamiento de los ingresos del hotel a través de los meses y trimestres, se seleccionaron las columnas de la dimensión tiempo de mes y trimestre y la columna de la tabla de hechos de reservas de ingreso total. Posteriormente se definió como columnas de entrada los atributos de mes y trimestre y como columna de predicción el atributo de ingreso total.

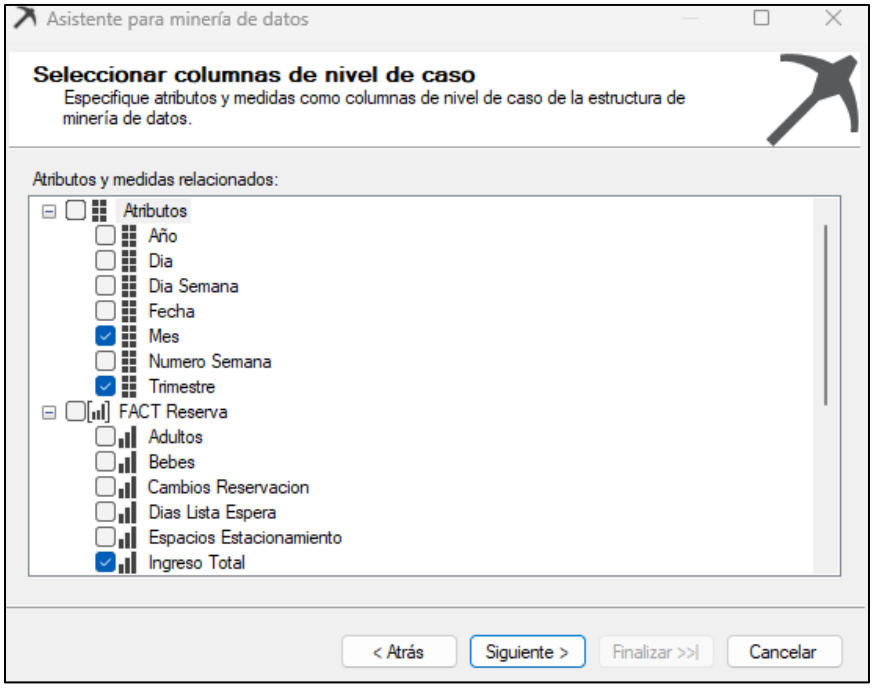

Figura 9.1.5. Selección de atributos y medidas

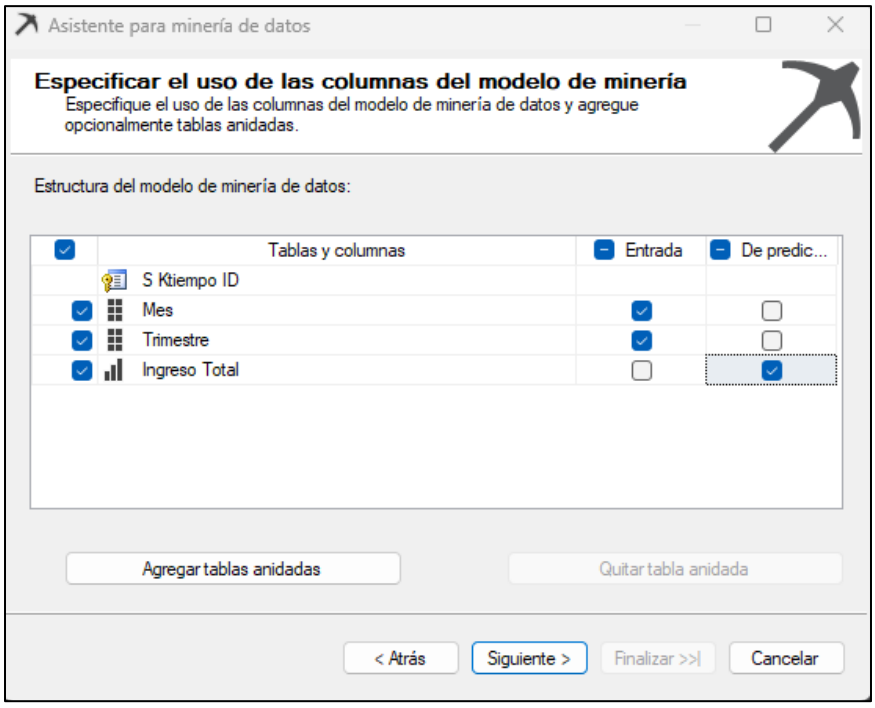

Figura 9.1.6. Definición de columnas de predicción y entrada

Resultados Obtenidos.

A continuación, se presentan capturas de pantalla como evidencia de la implementación del algoritmo de minería de datos de árboles de decisión.

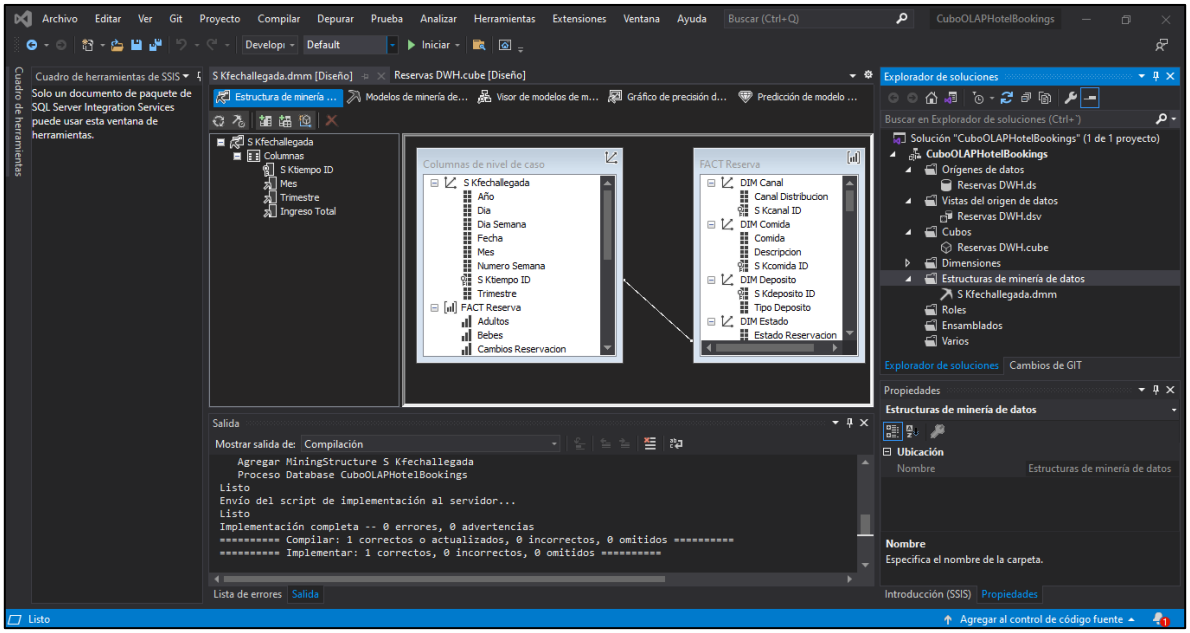

Figura 9.2.1. Captura de creación de estructura de minería de datos

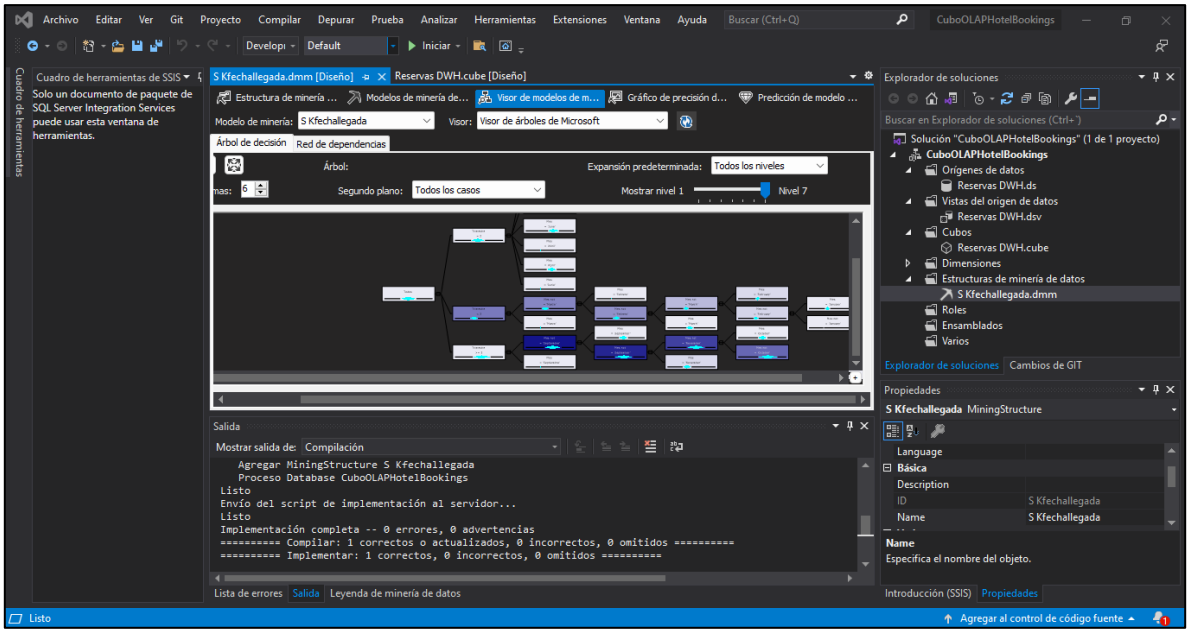

Figura 9.2.2. Captura de creación de árbol de decisión

Tras procesar la estructura de minería de datos definida anteriormente y realizar el proceso de compilación, se generó el siguiente árbol de decisión empleando como datos de entrada los meses y trimestres y como datos de salida los ingresos totales.

Figura 9.2.3. Árbol resultante del modelo de minería de datos

Resultados:

Este árbol muestra ramificaciones en base al ingreso total de las reservaciones en torno a los meses, este hace predicciones iniciales que se definen en base a tres trimestres del año, delimitando rangos. Posteriormente realiza ramificaciones, utilizando los meses contenidos en los trimestres establecidos.

Es de dos de estos trimestres que continúa realizando ramificaciones y decisiones calificando cada mes correspondiente a su respectivo ingreso hasta un quinto nivel, es de la segunda ramificación base que origina una hoja hasta un sexto nivel.

En base a estas hojas finales se obtiene que el mes de mayores ingresos es diciembre, seguido de julio, septiembre, octubre y noviembre; marcando una clara tendencia de aumento de ingresos en el último trimestre del año.

Conclusiones

La aplicación de un algoritmo de minería de datos permite obtener información clave para la toma de decisiones y acciones de un caso de estudio mediante la identificación de patrones, tendencias y correlaciones de grandes cantidades de datos. Por lo cual, la aplicación de algoritmos de minería de datos resulta ser clave en los procesos de análisis de datos pues permite obtener información difícil de encontrar de manera visual o descriptiva.

De esta manera mediante la aplicación del algoritmo de árboles de decisión a nuestro caso de estudio acerca de hoteles se ha identificado el comportamiento de los ingresos del hotel por trimestres y meses, permitiendo reconocer que los meses con mayor ingreso son los correspondientes a fechas vacacionales: julio y diciembre. Mientras que lo de menos ingresos son los correspondientes al primer trimestre del año. Información clave para la administración de un hotel pues a través de una toma de acciones se puede hacer un enfoque esencial en la mejora de los servicios durante los meses con mayor ingreso y trabajar campañas publicitarias,

así como alianzas con agencias y corporativos para mejorar el ingreso en los meses con menores cantidades además de trabajar estrategias que permitan una mejor administración de los recursos conforme al comportamiento de ingreso por mes.

Este proyecto ha demostrado la importancia de un enfoque analítico basado en datos para mejorar la competitividad en la industria hotelera. La combinación de tecnologías de Data Warehousing, análisis multidimensional y técnicas de minería de datos no solo optimiza la operación actual, sino que también establece una base sólida para la toma de decisiones informadas y estratégicas en el futuro para la administración de los hoteles.