```sql
--This statement creates dimension table d_customer with attributes and primary key is assigned to
dimension_customer_id

CREATE TABLE AdventureWorks_DW.star_schema.d_customer
(
    dimension_customer_id INT         NOT NULL IDENTITY,
    customer_id           INT         NOT NULL,
    title                 varchar(10) NOT NULL,
    first_name            varchar(50) NOT NULL,
    middle_name           varchar(50) NOT NULL,
    last_name             varchar(50) NOT NULL,
    valid_from            DATETIME    NOT NULL,
    valid_to              DATETIME    NOT NULL,
    PRIMARY KEY (dimension_customer_id)
);
--This statement creates dimension table d_product with attributes and primary key is assigned to
dimension_product_id
CREATE TABLE AdventureWorks_DW.star_schema.d_product
(
    dimension_product_id INT         NOT NULL IDENTITY,
    product_id           INT         NOT NULL,
    name                 varchar(50) NOT NULL,
    valid_from           DATETIME    NOT NULL,
    valid_to             DATETIME    NOT NULL,
    PRIMARY KEY (dimension_product_id)
);
--This statement creates dimension table d_date with attributes and primary key is assigned to
dimension_date_id
CREATE TABLE AdventureWorks_DW.star_schema.d_date
(
    dimension_date_id INT         NOT NULL IDENTITY,
    month_name        varchar(10) NOT NULL,
    day_name          varchar(10) NOT NULL,
    date              date        NOT NULL,
    PRIMARY KEY (dimension_date_id)
);

--This statement creates fact table f_sales with attributes, primary key is assigned to sales_id and
foreign keys customer_id, product_id
CREATE TABLE AdventureWorks_DW.star_schema.f_sales
(
    sales_id    INT NOT NULL IDENTITY,
    customer_id INT NOT NULL,
    product_id  INT NOT NULL,
    date_id     INT NOT NULL,
    quantity    INT NOT NULL,
    line_total  INT NOT NULL,
    PRIMARY KEY (sales_id),
    FOREIGN KEY (customer_id) REFERENCES AdventureWorks_DW.star_schema.d_customer (dimension_customer_id),
    FOREIGN KEY (product_id) REFERENCES AdventureWorks_DW.star_schema.d_product (dimension_product_id),
    FOREIGN KEY (date_id) REFERENCES AdventureWorks_DW.star_schema.d_date (dimension_date_id)
);
```

```
-- ***********************************************
-- ************ CREATING STAGE TABLES ************
-- ***********************************************
--This statement creates staging dimension table stage_dim_customer with attributes and assigned
-- primary key as customer_id
CREATE TABLE StagingDatabase.staging.stage_dim_customer
(
    customer_id           INT NOT NULL,
    title                 varchar(10),
    first_name            varchar(50),
    middle_name           varchar(50),
    last_name             varchar(50),
    valid_from            DATETIME,
    valid_to              DATETIME,
);
--This statement creates staging dimension table stage_dim_product with attributes and assigned
-- primary key as dimension_product_id
CREATE TABLE StagingDatabase.staging.stage_dim_product
(
    product_id           INT,
    name                 varchar(50),
    valid_from           DATETIME,
    valid_to             DATETIME,
);
--This statement creates staging dimension table stage_dim_date with attributes and assigned primary
-- key as dimension_date_id
CREATE TABLE StagingDatabase.staging.stage_dim_date
(
    month_name        varchar(10),
    day_name          varchar(10),
    date              DATETIME NOT NULL,
);

-- Create a table which holds last update variable
CREATE TABLE StagingDatabase.staging.LastUpdate
(
    lastUpdate DATETIME DEFAULT GETDATE()
);

--This statement creates staging fact table stage_f_sales with attributes and assigned primary key as
sales_id
CREATE TABLE StagingDatabase.staging.stage_f_sales
(
    customer_id          INT      NULL,
    product_id           INT      NULL,
    date_id              INT      NULL,
    business_customer_id INT      NULL,
    business_product_id  INT      NULL,
    business_order_date  DATETIME NULL,
    quantity             INT      NULL,
    line_total           FLOAT    NULL,
);

-- Create temporary table for f_sales.
CREATE TABLE StagingDatabase.staging.temp_f_sales
(
    customer_id          INT      NULL,
    product_id           INT      NULL,
    date_id              INT      NULL,
    business_customer_id INT      NULL,
    business_product_id  INT      NULL,
    business_order_date  DATETIME NULL,
    quantity             INT      NULL,
    line_total           FLOAT    NULL,
);

-- Create temporary table to store added products so we can handle valid_to attribute
```

```sql
CREATE TABLE StagingDatabase.staging.stage_dim_product_added
(
    product_id           INT,
    name                 varchar(50),
    valid_from           DATETIME,
    valid_to             DATETIME,
);

-- Create temporary table to store updated products so we can handle valid_to attribute and
-- deleting old products
CREATE TABLE StagingDatabase.staging.stage_dim_product_changed
(
    product_id           INT,
    name                 varchar(50),
    valid_from           DATETIME,
    valid_to             DATETIME,
);

-- Create temporary table to store updated customers so we can handle valid_to attribute and
-- deleting old customers
CREATE TABLE StagingDatabase.staging.stage_dim_customer_changed
    (
    customer_id         INT,
    title               varchar(10),
    first_name          varchar(50),
    middle_name          varchar(50),
    last_name          varchar(50),
    valid_from          DATETIME,
    valid_to         DATETIME,

    );
-- Create temporary table to store added customers so we can handle valid_to attribute
CREATE TABLE StagingDatabase.staging.stage_dim_customer_added
    (
    customer_id         INT,
    title               varchar(10),
    first_name          varchar(50),
    middle_name          varchar(50),
    last_name          varchar(50),
    valid_from          DATETIME,
    valid_to         DATETIME,

    );
```

```sql
-- ****************************************************************
-- ************ INSERTING AND FIXING DATA IN STAGE TABLES ************
-- ****************************************************************

-- _______________________ CUSTOMER _______________________

--This statement inserts attribute values into staging dimension table stage_dim_customer
INSERT INTO StagingDatabase.staging.stage_dim_customer(customer_id, title, first_name, middle_name,
last_name)
SELECT CustomerID, Title, FirstName, MiddleName, LastName
FROM AdventureWorks2017.Sales.Customer
         JOIN AdventureWorks2017.Person.Person ON Customer.PersonID = Person.BusinessEntityID;

--This statement removes null values in the title attribute in stage_dim_customer table by replacing with
'N/A'
UPDATE StagingDatabase.staging.stage_dim_customer
SET title='N/A'
WHERE title IS NULL;

--This statement removes null values in the middle_name attribute in stage_dim_customer table by
replacing with 'N/A'
UPDATE StagingDatabase.staging.stage_dim_customer
SET middle_name='N/A'
WHERE middle_name IS NULL;

-- Create new date for valid_from attribute. It will be the date when it was added to data warehouse
UPDATE StagingDatabase.staging.stage_dim_customer
SET valid_from=GETDATE()
WHERE valid_from IS NULL;

-- This statement replaces all NULL values with date 31.12.9999
UPDATE StagingDatabase.staging.stage_dim_customer
SET valid_to='9999-12-31'
WHERE valid_to IS NULL;


-- _______________________ PRODUCT _______________________

--This statement inserts attribute values into staging dimension table stage_dim_product
INSERT INTO StagingDatabase.staging.stage_dim_product(product_id, name, valid_from, valid_to)
SELECT ProductID, Name, SellStartDate, SellEndDate
from AdventureWorks2017.Production.Product;

--This statement removes null values in the name attribute in stage_dim_product table by replacing with
'N/A'
UPDATE StagingDatabase.staging.stage_dim_product
SET name='N/A'
WHERE name IS NULL;

-- This statement replaces all NULL values with date 31.12.9999
UPDATE StagingDatabase.staging.stage_dim_product
SET valid_to='9999-12-31'
WHERE valid_to IS NULL;


-- _______________________ DATE _______________________

--This statement inserts attribute values unto staging dimension table stage_dim_date
DECLARE @StartDate DATETIME = '2011-05-31'
DECLARE @EndDate DATETIME = '2014-06-30'

WHILE @StartDate <= @EndDate
    BEGIN
        INSERT INTO StagingDatabase.staging.stage_dim_date (date,
                                                            day_name,
                                                            month_name)
        SELECT @StartDate,
               DATENAME(weekday, @StartDate),
               DATENAME(month, @StartDate)

        SET @StartDate = DATEADD(dd, 1, @StartDate)
    END

-- ***********************************************************************
-- ************ INSERTING FIXED DATA INTO DW DIMENSION TABLES ************
-- ***********************************************************************

--This statement inserts attribute vales into dimension d_product table
INSERT INTO AdventureWorks_DW.star_schema.d_product (product_id, name, valid_from, valid_to)
SELECT *
FROM StagingDatabase.staging.stage_dim_product;

--This statement inserts attribute vales into dimension d_customer table
INSERT INTO AdventureWorks_DW.star_schema.d_customer (customer_id, title, first_name, middle_name,
last_name,
                                                      valid_from, valid_to)
SELECT *
FROM StagingDatabase.staging.stage_dim_customer;

--This statement inserts attribute vales into dimension d_date table
INSERT INTO AdventureWorks_DW.star_schema.d_date (month_name, day_name, date)
SELECT *
FROM StagingDatabase.staging.stage_dim_date;


-- ***************************************************************
-- ************ INSERTING FIXED DATA INTO FACT TABLES ************
-- ***************************************************************

--This statement inserts attribute values into staging fact table stage_f_sales
INSERT INTO StagingDatabase.staging.stage_f_sales(business_customer_id, business_product_id,
business_order_date,
                                                  quantity, line_total)
    (SELECT C.CustomerID, P.ProductID, OrderDate, SOD.OrderQty, SOD.LineTotal
     FROM AdventureWorks2017.Sales.SalesOrderHeader SOH
              JOIN AdventureWorks2017.Sales.SalesOrderDetail SOD on SOH.SalesOrderID = SOD.SalesOrderID
              JOIN AdventureWorks2017.Sales.Customer C on SOH.CustomerID = C.CustomerID
              JOIN AdventureWorks2017.Production.Product P on SOD.ProductID = P.ProductID
     WHERE OnlineOrderFlag = 1);


-- *******************************************************************
-- ********************** LOOKUP SURROGATE KEYS **********************
-- *******************************************************************

--This statement extracts the customer_id from staging dimension table stage_dim_customer and assigns it
to the
-- customer_id attribute in stage_f_sales table when the value is null
UPDATE StagingDatabase.staging.stage_f_sales
SET customer_id = (SELECT dimension_customer_id
                   FROM AdventureWorks_DW.star_schema.d_customer AS dim_C_id
                   WHERE dim_C_id.customer_id = business_customer_id)
WHERE customer_id IS NULL;

--This statement extracts the product_id from staging dimension table stage_dim_product and assigns it to
the
-- product_id attribute in stage_f_sales table when the value is null
UPDATE StagingDatabase.staging.stage_f_sales
SET product_id = (SELECT dimension_product_id
                  FROM AdventureWorks_DW.star_schema.d_product AS dim_P_id
                  WHERE dim_P_id.product_id = business_product_id)
WHERE product_id IS NULL;

--This statement extracts the date_id from staging dimension table stage_dim_date and assigns it to the
```

```sql
-- date_id attribute in stage_f_sales table when the value is null
UPDATE StagingDatabase.staging.stage_f_sales
SET date_id = (SELECT dimension_date_id
               FROM AdventureWorks_DW.star_schema.d_date AS dim_D_id
               WHERE dim_D_id.date = business_order_date)
WHERE date_id IS NULL;


-- ******************************************************************************************
-- ********************** INSERT VALUES INTO DATA WAREHOUSE FACT TABLE **********************
-- ******************************************************************************************

INSERT INTO AdventureWorks_DW.star_schema.f_sales(customer_id, product_id, date_id, quantity, line_total)
SELECT customer_id, product_id, date_id, quantity, line_total
FROM StagingDatabase.staging.stage_f_sales;
```

```sql
-- ******************************************************************************************
-- ************************************* NEW PRODUCTS ***************************************
-- ******************************************************************************************

-- Search and insert newly added product into staging added product table
INSERT INTO StagingDatabase.staging.stage_dim_product_added (product_id, name, valid_from, valid_to)
SELECT ProductID, Name, SellStartDate, SellEndDate
FROM AdventureWorks2017.Production.Product
WHERE productID IN (SELECT productID
                    FROM AdventureWorks2017.Production.Product
                        EXCEPT
                    SELECT product_id
                    FROM AdventureWorks_DW.star_schema.d_product);

-- Replace all NULL values with date 31.12.9999
UPDATE StagingDatabase.staging.stage_dim_product_added
SET valid_to='9999-12-31'
WHERE valid_to IS NULL;

-- Load newly added and modified rows into the Data Warehouse
INSERT INTO AdventureWorks_DW.star_schema.d_product
SELECT *
FROM StagingDatabase.staging.stage_dim_product_added;

-- **********************************************************************************************
-- ************************************* DELETED PRODUCTS ***************************************
-- **********************************************************************************************

-- Retrieve and update data warehouse, set valid_to attribute to yesterdays date for deleted
-- products.
UPDATE AdventureWorks_DW.star_schema.d_product
SET valid_to = DATEADD(dd, -1, GETDATE())
WHERE product_id in (
    SELECT product_id
    FROM AdventureWorks_DW.star_schema.d_product
    WHERE product_id IN (SELECT product_id
                         FROM AdventureWorks_DW.star_schema.d_product
                             EXCEPT
                         SELECT productID
                         FROM AdventureWorks2017.Production.Product)
)

-- **********************************************************************************************
-- ************************************* UPDATED PRODUCTS ***************************************
-- **********************************************************************************************

-- Inserting updated rows into the temporary table to handle changes
INSERT INTO StagingDatabase.staging.stage_dim_product_changed
    (product_id, name, valid_from) (SELECT ProductID, Name, SellStartDate
                                    FROM AdventureWorks2017.Production.Product
                                        EXCEPT
                                    SELECT product_id, name, valid_from
                                    FROM StagingDatabase.staging.stage_dim_product
                                        EXCEPT (
                                             SELECT ProductID, Name, SellStartDate
                                             FROM AdventureWorks2017.Production.Product
                                             WHERE productID IN
                                                   (SELECT productID
                                                    FROM AdventureWorks2017.Production.Product
                                                        EXCEPT
                                                    SELECT product_id
                                                    FROM StagingDatabase.staging.stage_dim_product)
                                         ));

-- Update valid_to attribute to '9999-12-31'
UPDATE StagingDatabase.staging.stage_dim_product_changed
SET valid_to = '9999-12-31'
```

```sql
WHERE valid_to IS NULL;

-- Alter changed rows in Data Warehouse
UPDATE AdventureWorks_DW.star_schema.d_product
SET valid_to = DATEADD(dd, -1, GETDATE())
WHERE product_id in (SELECT product_id FROM StagingDatabase.staging.stage_dim_product_changed);

-- Insert new product to Data Warehouse
INSERT INTO AdventureWorks_DW.star_schema.d_product
SELECT *
FROM StagingDatabase.staging.stage_dim_product_changed;
```

```sql
-- Search and insert newly added customer into staging added Customer table
INSERT INTO StagingDatabase.staging.stage_dim_customer_added(customer_id, title, first_name, middle_name,
last_name)
SELECT CustomerID,Title,FirstName,MiddleName,LastName
FROM AdventureWorks2017.Sales.Customer JOIN AdventureWorks2017.Person.Person ON Customer.PersonID =
Person.BusinessEntityID
WHERE CustomerID IN (SELECT CustomerID
                     FROM AdventureWorks2017.Sales.Customer
                         EXCEPT
                     SELECT customer_id
                     FROM StagingDatabase.staging.stage_dim_customer);

-- Replace all NULL values with current date
UPDATE StagingDatabase.staging.stage_dim_customer_added
SET valid_from = GETDATE()
WHERE valid_to IS NULL;

-- Replace all NULL values with date 31.12.9999
UPDATE StagingDatabase.staging.stage_dim_customer_added
SET valid_to = '9999-12-31'
WHERE valid_to IS NULL;

-- Load newly added and modified rows into the Data Warehouse
INSERT INTO AdventureWorks_DW.star_schema.d_customer
SELECT *
FROM StagingDatabase.staging.stage_dim_customer_added;


-- Retrieve and update data warehouse, set valid_to attribute to yesterdays date for deleted
-- customers.
UPDATE AdventureWorks_DW.star_schema.d_customer
SET valid_to = DATEADD(dd, -1, GETDATE())
WHERE customer_id IN (
    SELECT customer_id
    FROM AdventureWorks_DW.star_schema.d_customer
    WHERE customer_id IN (SELECT customer_id
                         FROM AdventureWorks_DW.star_schema.d_customer
                             EXCEPT
                         SELECT CustomerID
                         FROM AdventureWorks2017.Sales.Customer)
)


-- Inserting updated rows into the temporary table to handle changes
INSERT INTO StagingDatabase.staging.stage_dim_customer_changed
    (customer_id, title, first_name,middle_name,last_name) (SELECT CustomerID,Title, FirstName,
MiddleName, LastName
                                    FROM AdventureWorks2017.Sales.Customer
                                    JOIN AdventureWorks2017.Person.Person ON Customer.PersonID =
Person.BusinessEntityID
                                        EXCEPT
                                    SELECT customer_id, title, first_name,middle_name,last_name
                                    FROM StagingDatabase.staging.stage_dim_customer
                                        EXCEPT (
                                             SELECT CustomerID,Title, FirstName, MiddleName, LastName
                                             FROM AdventureWorks2017.Sales.Customer
                                             JOIN AdventureWorks2017.Person.Person ON Customer.PersonID =
Person.BusinessEntityID
                                             WHERE CustomerID IN
                                                   (SELECT CustomerID
                                                    FROM AdventureWorks2017.Sales.Customer
                                                        EXCEPT
                                                    SELECT customer_id
                                                    FROM StagingDatabase.staging.stage_dim_customer)
                                         ));

-- Update valid_to attribute to '9999-12-31'
```

```sql
UPDATE StagingDatabase.staging.stage_dim_customer_changed
SET valid_from = GETDATE()
WHERE valid_from IS NULL;


-- Update title attribute to 'N/A'
UPDATE StagingDatabase.staging.stage_dim_customer_changed
SET valid_to = '9999-12-31'
WHERE valid_to IS NULL;

-- Update middle_name attribute to 'N/A'
UPDATE StagingDatabase.staging.stage_dim_customer_changed
SET title = 'N/A'
WHERE title IS NULL;

-- Update valid_to attribute to 'N/A'
UPDATE StagingDatabase.staging.stage_dim_customer_changed
SET middle_name = 'N/A'
WHERE middle_name IS NULL;

-- Alter changed rows in Data Warehouse
UPDATE AdventureWorks_DW.star_schema.d_customer
SET valid_to = DATEADD(dd, -1, GETDATE())
WHERE customer_id in (SELECT customer_id FROM StagingDatabase.staging.stage_dim_customer_changed);

-- Insert new customer to Data Warehouse
INSERT INTO AdventureWorks_DW.star_schema.d_customer
SELECT *
FROM StagingDatabase.staging.stage_dim_customer_changed;
```

```sql
-- ***********************************************************************************************
-- ************************************* FACT TABLE UPDATE ***************************************
-- ***********************************************************************************************

DECLARE @LAST_UPDATE as DATETIME = (SELECT lastUpdate
                                    FROM StagingDatabase.staging.LastUpdate);

-- Insert newly updated rows into temp_f_sales table. Select only the ones newer than the last update.
INSERT INTO StagingDatabase.staging.stage_f_sales
(business_customer_id, business_product_id, business_order_date, quantity, line_total)
    (SELECT C.CustomerID, P.ProductID, OrderDate, SOD.OrderQty, SOD.LineTotal
     FROM AdventureWorks2017.Sales.SalesOrderHeader SOH
              JOIN AdventureWorks2017.Sales.SalesOrderDetail SOD on SOH.SalesOrderID = SOD.SalesOrderID
              JOIN AdventureWorks2017.Sales.Customer C on SOH.CustomerID = C.CustomerID
              JOIN AdventureWorks2017.Production.Product P on SOD.ProductID = P.ProductID
     WHERE OnlineOrderFlag = 1
       AND OrderDate > @LAST_UPDATE);

-- Find corresponding surrogate keys.
-- ***************************** Customer *****************************
UPDATE StagingDatabase.staging.stage_f_sales
SET customer_id = (SELECT dimension_customer_id
                   FROM AdventureWorks_DW.star_schema.d_customer AS dim_C_id
                   WHERE dim_C_id.customer_id = business_customer_id
                     AND valid_to = '9999-12-31')
WHERE customer_id IS NULL;

-- ***************************** Product *****************************
UPDATE StagingDatabase.staging.stage_f_sales
SET product_id = (SELECT dimension_product_id
                  FROM AdventureWorks_DW.star_schema.d_product AS dim_P_id
                  WHERE dim_P_id.product_id = business_product_id
                    AND valid_to = '9999-12-31')
WHERE product_id IS NULL;

-- ***************************** Date *****************************
UPDATE StagingDatabase.staging.stage_f_sales
SET date_id = (SELECT dimension_date_id
               FROM AdventureWorks_DW.star_schema.d_date AS dim_D_id
               WHERE dim_D_id.date = business_order_date)
WHERE date_id IS NULL;

-- Insert data into Data Warehouse Fact Sales table
INSERT INTO AdventureWorks_DW.star_schema.f_sales(customer_id, product_id, date_id, quantity, line_total)
SELECT customer_id, product_id, date_id, quantity, line_total
FROM StagingDatabase.staging.temp_f_sales;

-- Update last update table with the newest date
UPDATE StagingDatabase.staging.LastUpdate
SET lastUpdate = GETDATE()
WHERE lastUpdate = @LAST_UPDATE;
```