Business Template

BANDCAMP (ONLINE MUSIC PLATFORM) SALES

CONTENTS

1 BUSINESS DESCRIPTION

1.1 BUSINESS BACKGROUND

Bandcamp is an online music platform that allows independent artists and labels to sell their music directly to fans.

Artists can create their own pages on Bandcamp and upload their music for sale, and fans can purchase digital or physical copies of albums, as well as merchandise like t-shirts and posters. Bandcamp takes a percentage of each sale, but allows artists to set their own prices and keep a larger portion of the revenue than they would with most traditional music distribution models.

1.2 PROBLEMS BECAUSE OF POOR DATA MANAGEMENT

Due to the rapid growth of the business, the current OLTP (Online Transaction Processing) database is unable to provide the necessary speed and performance required for efficient analyses.

1.3 BENEFITS FROM IMPLEMENTING A DATA WAREHOUSE

Using of data warehouse can help you with the problems described above. Business can gain insights into customer behavior, sales trends, and marketing effectiveness, and use these insights to make more informed business decisions.

Implementing a data warehouse can answer you the following questions:

- Which artists have the highest prices?
- Which geographic regions generate the most sales revenue?
- What is the average time between a customer's first and second purchase?
- What is the average price range of merchandise sold on Bandcamp?

1.4 DATASETS DESCRIPTION

The first dataset contains the following information about sales of **digital** products on Bandcamp site:

Product Information:

Prod_id: unique product id

URL: the path to the item on Bandcamp.

item_description: description.

image_url: path to the item's art image.

album_title: Title of the album, if applicable.

item_type: denotes the type of object. a for digital albums, t for digital tracks , b for bundles.

item_price: price of the item in the seller's currency.

currency: the seller's currency.

Sales Information:

_id: unique identifier combining the sale's URL and UTC timestamp.

amount_paid: amount paid in the seller's currency.

amount_paid_usd: amount paid converted to US Dollars.

Artist Information:

artist_name: Name of the artist.

Genre: main artists genre

Time information:

utc_date: the UTC timestamp of the sale datetime.

Geography information:

country_code: country code of the buyer.

country: full country code name of the buyer.

Region: the appropriate region for each country.

Customer Information:

Username: username.

Name: full name.

Sex: M or F

Address: text address.

Posstcode

Mail: email

Birthdate: date YYYU-MM-DD

Promotion information:

Promotion ID

Promotion name

Promotion description

The 2nd dataset contains the information about sales of **physical items** on Bandcamp site with similar data structure with next differences:

1. Item type:

Product Information:

Prod_id: uniqiue identifier

url: the path to the item on Bandcamp.

item_description: description.

art_url: path to the item's art image.

album_title: Title of the album, if applicable.

item_type: denotes the type of object. **p for physical items.**

item_price: price of the item in the seller's currency.

currency: the seller's currency.

2. Additional dimension:

Shipping information:

Shipment track number.

Shipment date.

Shipment received: date, when shipment was received.

The data on sources captured till milliseconds, because it is a part of key for uniqueness. But for grain I decided to choose a day as the level, cause daily data is easy to understand and interpret, it can be used to track trends and patterns over time, and compare performance across different time periods.

A business process: The seller sells a product to a customer the customer pays for it.

The grain is: each individual sale of a digital or merch product.

One row per sale of the certain product, made by the artist, at certain time, with eligible promotion or without it, with shipping if necessary, to the customer, who pays the amount of money for it.

The dimensions are: product, artist, time, promotion, shipping, customer, address, payments currency.

The facts are sale item price, amount in dollars, amount in sellers currency, platform fee percent, fee discount.

Dimension: DIM_PRODUCTS

Describes every item in the music online store.

Column name	Description	Data Type
PRODUCT_URL	The path to the item on site.	VARCHAR(255)
PRODUCT_DESCRIPTION	Description.	VARCHAR(255)
PRODUCT_IMAGE_URL	Path to the item's art image.	VARCHAR(100)
PRODUCT_ALBUM_TITLE	Title of the album, if applicable.	VARCHAR(255)
PRODUCT_PRICE	Price of the product in the purchased currency.	NUMERIC(10,2)
PRODUCT_TYPE	Type: 'a' for digital album, 't' for digital track, 'p' for physical items.	VARCHAR(1)

Example:

PRODUCT_URL	PRODUCT_DESCRIPTION	PRODUCT_IMAGE_URL	PRODUCT_ALBUM_TITLE	PRODUCT_PRICE	PRODUCT_TYPE
//girlbanddublin.bandcamp.com/album/live-at-vicar-street	Live at Vicar Street	https://f4.bcbits.com/img/a0206405257_7.jpg	Live at Vicar Street	9,99	a

Dimension: DIM_CURRENCIES

Provides information about currencies in transactions.

Column name	Description	Data Type
CURRENCY_PAID_SYMB	Amount paid in the purchased currency, with currency symbol.	VARCHAR(13)
CURRENCY_CODE	The seller's(artists) currency code (USD, EUR etc).	VARCHAR(3)

Example:

CURRENCY_PAID_SYMB	CURRENCY_CODE
$9.99	USD
£1	GBP

Dimension: DIM_ARTISTS

Provides information about artists – creators of the selling items.

Column name	Description	Data Type
ARTIST_NAME	Name of the artist.	VARCHAR(100)
ARTIST_GENRE	Current genre of music for the artist.	VARCHAR(30)

Example:

ARTIST_NAME	ARTIST_GENRE
Girl Band	Pop

Dimension: DIM_TIMES

Provides information about time periods. Attributes can be added according business requirements.

Column name	Description	Data Type
TIME_SRC_ID	The UTC timestamp of the sale datetime, natural key.	BIGINT
TIME_YYYY	Year of the transaction.	INT
TIME_MM	Month of the transaction.	INT
TIME_DT	Date of the transaction.	DATE
TIME_WEEK_NUM	Week number of the transaction.	INT
TIME_MONTH_NAME	Months name.	VARACHAR(9)

Example:

TIME_SRC_ID	TIME_YYYY	TIME_MM	TIME_DT	TIME_WEEK_NUM	TIME_MONTH_NAME
1.599689e+09	2022	12	20.12.2022	51	December

Dimension: DIM ADRESSES
Provides information about the physical location of the customers.

Column name	Description	Data Type
ADDRESS_TEXT	Address of the customer.	VARCHAR(255)
ADDRESS_COUNTRY_CODE	Country code of the buyer	VARCHAR(2)
ADDRESS_COUNTRY	Full country name	VARCHAR(32)
ADDRESS_REGION	World's region name	VARCHAR(15)

Example:

ADDRESS_COUNTRY_CODE	ADDRESS_COUNTRY	ADDRESS_REGION
gb	United Kingdom	Europe

Dimension: DIM_CUSTOMERS
Provides information about customers.

Column name	Description	Data Type
CUSTOMER_NAME	Full name of the customer.	VARCHAR(100)
CUSTOMER_GENDER	Gender of the customer, M or F	VARCHAR(1)
CUSTOMER_EMAIL	Email of the customer, can be used as natural key	VARCHAR(50)
CUSTOMER_BIRTHDATE	Day of the transaction.	DATE

Example:

CUSTOMER_NAME	CUSTOMER_GENDER	CUSTOMER_ADDRESS	CUSTOMER_EMAIL	CUSTOMER_BIRTHDATE
Jessica Patel	F	97832 Palmer Inlet Apt. 593\nWest Kathleenboro	christophermorgan@gmail.com	07.06.1972

Dimension: DIM_PROMOTIONS
Provides information about promotions.

Column name	Description	Data Type
PROMOTION_SRC_ID	Promotion campaign source id.	VARCHAR(15)
PROMOTION_NAME	Promotion campaign name.	VARCHAR(100)
PROMOTION_DESC	Promotion description.	VARCHAR(255)

Example:

PROMOTION_SRC_ID	PROMOTION_NAME	PROMOTION_DESC
1234	Free Shipping	Free shipping for merch

Dimension: DIM_SHIPMENTS
For physical items provides shipment details.

Column name	Description	Data Type
SHIPMENT_TRACK_NUMBER_SRC_ID	Unique shipment source id.	VARCHAR(30)
SHIPMENT_DT	Date of the shipment.	DATE
SHIPMENT_RECEIVED_DT	Date, when shipment was received.	DATE

Example:

SHIPMENT_TRACK_NUMBER_SRC_ID	SHIPMENT_DT	SHIPMENT_RECEIVED_DT
12138115	01.06.2023	09.06.2023

Fact table: FCT_SALES_DD
Provides information about sales amount in dollars.
References to dimension tables omitted (for better presentation).

Column name	Description	Data Type
SALE_SRC_ID	Unique identifier combining the sale's URL and UTC timestamp.	VARCHAR(100)
EVENT_DT	Date of the sale transaction.	DATE
SALE_AMOUNT_PAID_USD	Sale amount in dollars.	NUMERIC(10, 2)
SALE_AMOUNT_PAID	Sale amount in currency of the seller	NUMERIC(10, 2)
SALE_PRODUCT_PRICE	Sale amount in currency of the seller	NUMERIC(10, 2)
INSERT_DT	Date of fact row creation.	DATE
UPDATE_DT	Date of fact last time row modification.	DATE

Example:

SALE_SRC_ID	EVENT_DT	SALE_AMOUNT_PAID_USD	SALE_AMOUNT_PAID	SALE_PRODUCT_PRICE	INSERT_DT	UPDATE_DT
1599688803.5175&//girlbanddublin.bandcamp.com/..	01.01.2023	9,99	9,99	9,99	15.07.2023	18.07.2023

2 BUSINESS LAYER 3NF

On the previous step, the key business entities were identified. Based on these entities and their relationships, the base tables were designed.

The settlement for the grain was established as follows: each individual sale of a digital or physical merchandise product. Therefore, the Sale table will serve as the central table, containing key metrics such as the sale amount in both USD and the seller's currency, as well as most of the relationships with other tables.

All other tables were created to represent a single dimensional entity with attributes (whose data types were also defined in the previous step), and with a primary key that uniquely identifies each record.

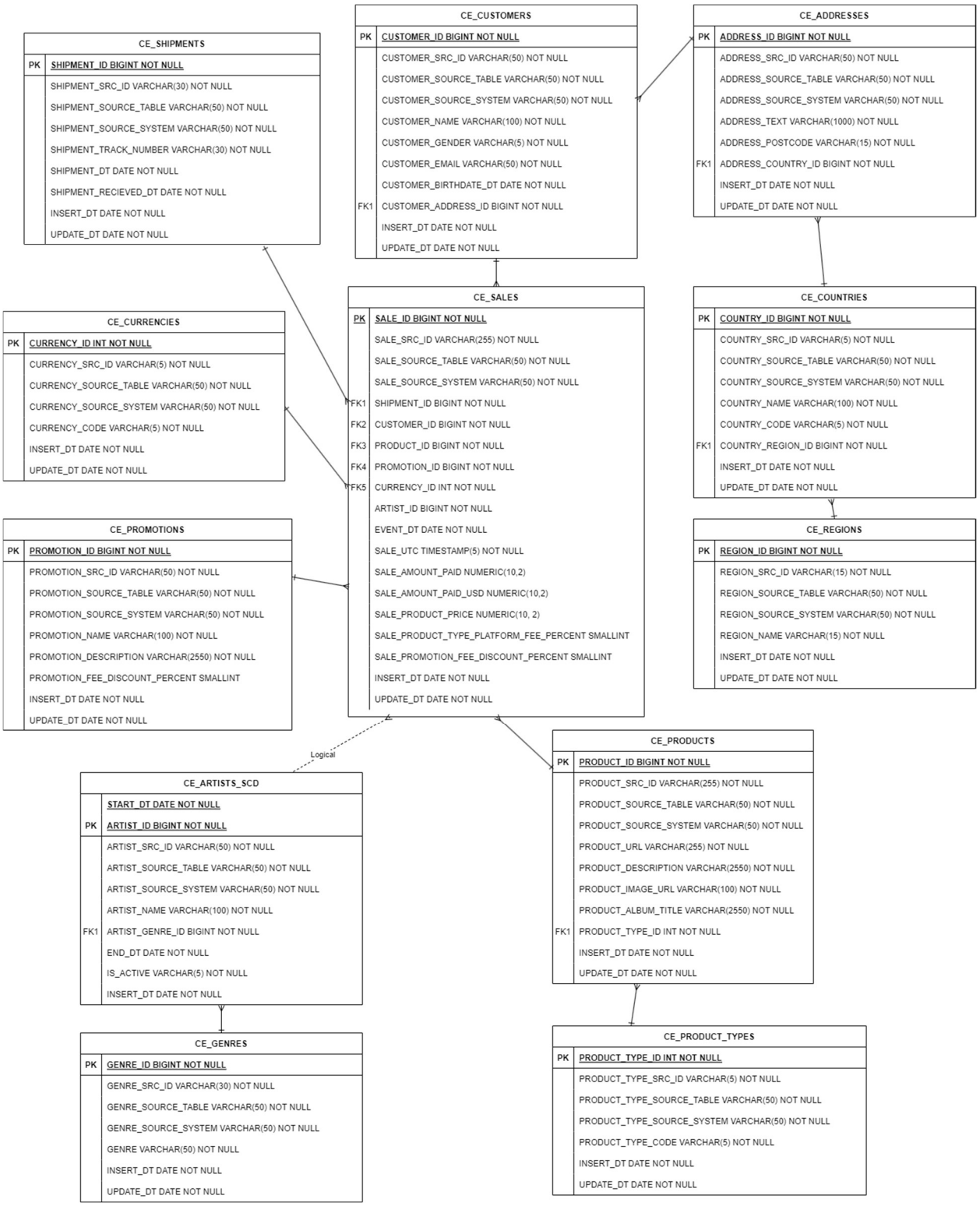
The next step was table normalization, which included splitting the tables to eliminate transitive dependencies. For example, the Addresses dimension was split, and foreign keys were added to parent tables.

On the next step, the table CE_SALES was created with foreign keys to many tables and with measure fields. The relations were displayed, and the UTC-timestamp was included in that table.

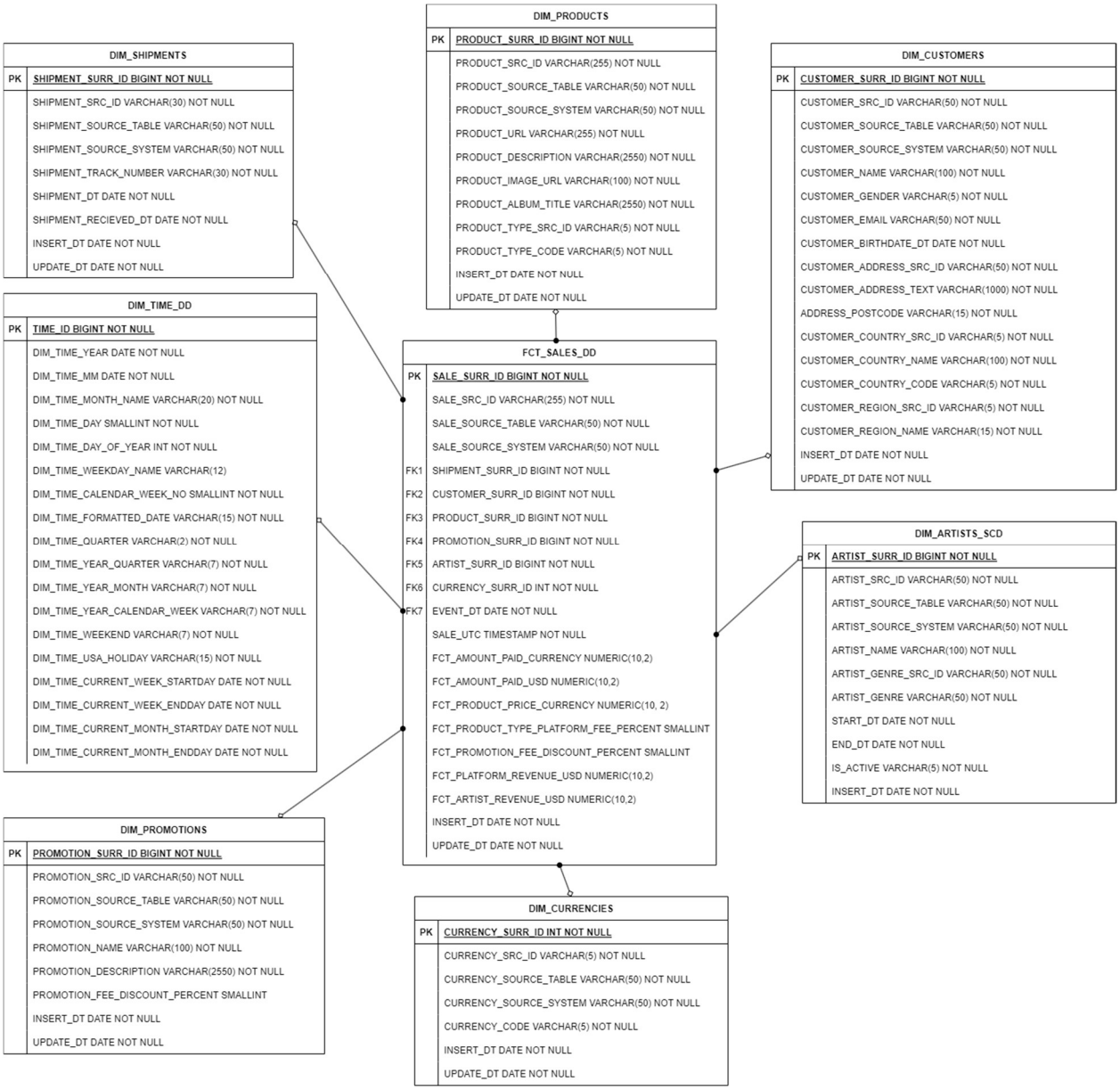
On the next step, source triplet fields were implemented in the tables.

The entity Artists have a Genre attribute that can be changed, but it is important to know the time period when the previous genre was active. Therefore, that table was tagged as an SCD2-type table (CE_ARTISTS_SCD), and necessary fields were added.

Next, fields INSERT_DT and UPDATE_DT were added to tables with SCD Type 1 design. As example, each customer is allowed to have only one address, but it is important to know how old it is. Therefore, the table CE_ADDRESSES was designed as an SCD Type1 table.


CE_SHIPMENTS
PK SHIPMENT_ID BIGINT NOT NULL
SHIPMENT_SRC_ID VARCHAR(30) NOT NULL
SHIPMENT_SOURCE_TABLE VARCHAR(50) NOT NULL
SHIPMENT_SOURCE_SYSTEM VARCHAR(50) NOT NULL
SHIPMENT_TRACK_NUMBER VARCHAR(30) NOT NULL
SHIPMENT_DT DATE NOT NULL
SHIPMENT_RECIEVED_DT DATE NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_CUSTOMERS
PK CUSTOMER_ID BIGINT NOT NULL
CUSTOMER_SRC_ID VARCHAR(50) NOT NULL
CUSTOMER_SOURCE_TABLE VARCHAR(50) NOT NULL
CUSTOMER_SOURCE_SYSTEM VARCHAR(50) NOT NULL
CUSTOMER_NAME VARCHAR(100) NOT NULL
CUSTOMER_GENDER VARCHAR(5) NOT NULL
CUSTOMER_EMAIL VARCHAR(50) NOT NULL
CUSTOMER_BIRTHDATE_DT DATE NOT NULL
FK1 CUSTOMER_ADDRESS_ID BIGINT NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_ADDRESSES
PK ADDRESS_ID BIGINT NOT NULL
ADDRESS_SRC_ID VARCHAR(50) NOT NULL
ADDRESS_SOURCE_TABLE VARCHAR(50) NOT NULL
ADDRESS_SOURCE_SYSTEM VARCHAR(50) NOT NULL
ADDRESS_TEXT VARCHAR(1000) NOT NULL
ADDRESS_POSTCODE VARCHAR(15) NOT NULL
FK1 ADDRESS_COUNTRY_ID BIGINT NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_CURRENCIES
PK CURRENCY_ID INT NOT NULL
CURRENCY_SRC_ID VARCHAR(5) NOT NULL
CURRENCY_SOURCE_TABLE VARCHAR(50) NOT NULL
CURRENCY_SOURCE_SYSTEM VARCHAR(50) NOT NULL
CURRENCY_CODE VARCHAR(5) NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_SALES
PK SALE_ID BIGINT NOT NULL
SALE_SRC_ID VARCHAR(255) NOT NULL
SALE_SOURCE_TABLE VARCHAR(50) NOT NULL
SALE_SOURCE_SYSTEM VARCHAR(50) NOT NULL
FK1 SHIPMENT_ID BIGINT NOT NULL
FK2 CUSTOMER_ID BIGINT NOT NULL
FK3 PRODUCT_ID BIGINT NOT NULL
FK4 PROMOTION_ID BIGINT NOT NULL
FK5 CURRENCY_ID INT NOT NULL
ARTIST_ID BIGINT NOT NULL
EVENT_DT DATE NOT NULL
SALE_UTC TIMESTAMP(5) NOT NULL
SALE_AMOUNT_PAID NUMERIC(10,2)
SALE_AMOUNT_PAID_USD NUMERIC(10,2)
SALE_PRODUCT_PRICE NUMERIC(10, 2)
SALE_PRODUCT_TYPE_PLATFORM_FEE_PERCENT SMALLINT
SALE_PROMOTION_FEE_DISCOUNT_PERCENT SMALLINT
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_COUNTRIES
PK COUNTRY_ID BIGINT NOT NULL
COUNTRY_SRC_ID VARCHAR(5) NOT NULL
COUNTRY_SOURCE_TABLE VARCHAR(50) NOT NULL
COUNTRY_SOURCE_SYSTEM VARCHAR(50) NOT NULL
COUNTRY_NAME VARCHAR(100) NOT NULL
COUNTRY_CODE VARCHAR(5) NOT NULL
FK1 COUNTRY_REGION_ID BIGINT NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_PROMOTIONS
PK PROMOTION_ID BIGINT NOT NULL
PROMOTION_SRC_ID VARCHAR(50) NOT NULL
PROMOTION_SOURCE_TABLE VARCHAR(50) NOT NULL
PROMOTION_SOURCE_SYSTEM VARCHAR(50) NOT NULL
PROMOTION_NAME VARCHAR(100) NOT NULL
PROMOTION_DESCRIPTION VARCHAR(2550) NOT NULL
PROMOTION_FEE_DISCOUNT_PERCENT SMALLINT
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_REGIONS
PK REGION_ID BIGINT NOT NULL
REGION_SRC_ID VARCHAR(15) NOT NULL
REGION_SOURCE_TABLE VARCHAR(50) NOT NULL
REGION_SOURCE_SYSTEM VARCHAR(50) NOT NULL
REGION_NAME VARCHAR(15) NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
Logical
CE_ARTISTS_SCD
START_DT DATE NOT NULL
PK ARTIST_ID BIGINT NOT NULL
ARTIST_SRC_ID VARCHAR(50) NOT NULL
ARTIST_SOURCE_TABLE VARCHAR(50) NOT NULL
ARTIST_SOURCE_SYSTEM VARCHAR(50) NOT NULL
ARTIST_NAME VARCHAR(100) NOT NULL
FK1 ARTIST_GENRE_ID BIGINT NOT NULL
END_DT DATE NOT NULL
IS_ACTIVE VARCHAR(5) NOT NULL
INSERT_DT DATE NOT NULL
CE_PRODUCTS
PK PRODUCT_ID BIGINT NOT NULL
PRODUCT_SRC_ID VARCHAR(255) NOT NULL
PRODUCT_SOURCE_TABLE VARCHAR(50) NOT NULL
PRODUCT_SOURCE_SYSTEM VARCHAR(50) NOT NULL
PRODUCT_URL VARCHAR(255) NOT NULL
PRODUCT_DESCRIPTION VARCHAR(2550) NOT NULL
PRODUCT_IMAGE_URL VARCHAR(100) NOT NULL
PRODUCT_ALBUM_TITLE VARCHAR(2550) NOT NULL
FK1 PRODUCT_TYPE_ID INT NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_GENRES
PK GENRE_ID BIGINT NOT NULL
GENRE_SRC_ID VARCHAR(30) NOT NULL
GENRE_SOURCE_TABLE VARCHAR(50) NOT NULL
GENRE_SOURCE_SYSTEM VARCHAR(50) NOT NULL
GENRE VARCHAR(50) NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL
CE_PRODUCT_TYPES
PK PRODUCT_TYPE_ID INT NOT NULL
PRODUCT_TYPE_SRC_ID VARCHAR(5) NOT NULL
PRODUCT_TYPE_SOURCE_TABLE VARCHAR(50) NOT NULL
PRODUCT_TYPE_SOURCE_SYSTEM VARCHAR(50) NOT NULL
PRODUCT_TYPE_CODE VARCHAR(5) NOT NULL
INSERT_DT DATE NOT NULL
UPDATE_DT DATE NOT NULL

3 BUSINESS LAYER DIMENSIONAL MODEL



The fact table FCT_SALES_DD contains the following metrics from the sources data:

- **FCT_AMOUNT_PAID_CURRENCY**: The *amount, paid by the customer paid for an item in the seller's currency.*
- **FCT_AMOUNT_PAID_USD**: *amount, paid by the customer for an item in US dollars.*
- **FCT_PRODUCT_PRICE_CURRENCY**: *The price of the item in the seller's currency.*
- **FCT_PRODUCT_TYPE_PLATFORM_FEE_PERCENT**: The *percentage of the sales amount that is intended for the platform; the percentage varies by product type.*

- **FCT_PROMOTION_FEE_DISCOUNT_PERCENT:** *The percentage by which the platform fee is reduced during promotional events for certain goods. The default value is 0.*

In addition, the following calculated metrics have been added:

1. **FCT_PLATFORM_REVENUE_USD**: *The amount of revenue in US dollars earned by the site, calculated using the following formula:*

   FCT_AMOUNT_PAID_USD *

   FCT_PRODUCT_TYPE_PLATFORM_FEE_PERCENT/100 *

   (1 - FCT_PROMOTION_FEE_DISCOUNT_PERCENT/100)

2. **FCT_ARTIST_REVENUE_USD**: *The amount of revenue in US dollars earned by the artist, calculated using the following formula:*

   FCT_AMOUNT_PAID_USD *

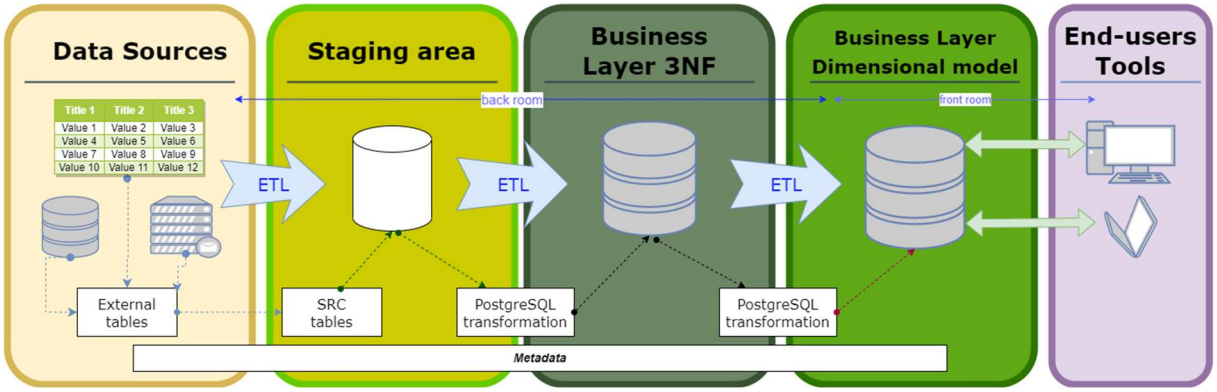
   (1 – FCT_PRODUCT_TYPE_PLATFORM_FEE_PERCENT/100 *

   (1 - FCT_PROMOTION_FEE_DISCOUNT_PERCENT/100)

)

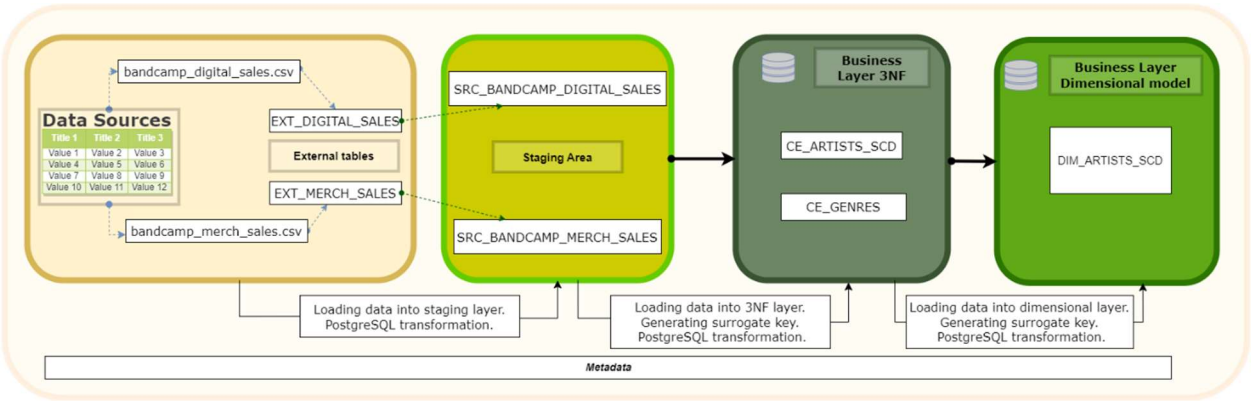
4 LOGICAL SCHEME

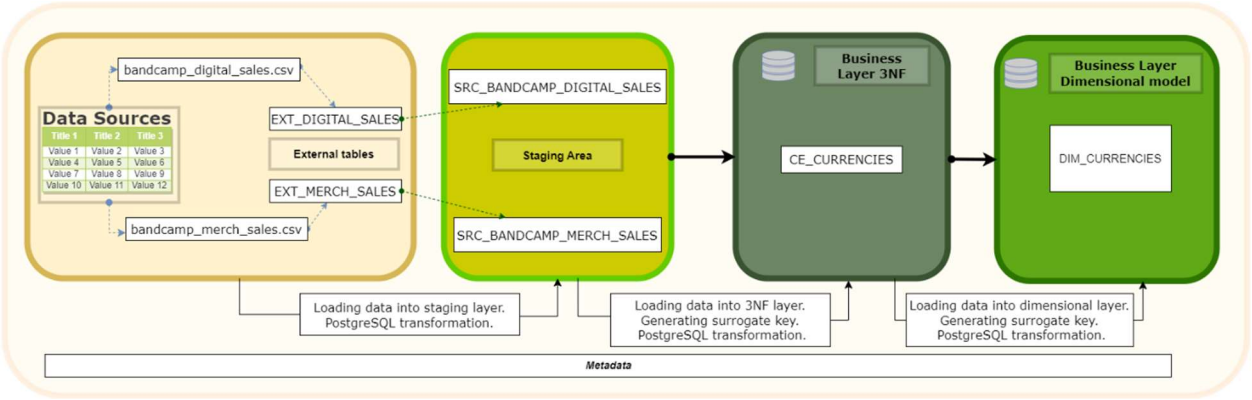
The process of loading the data to the Data Warehouse looks like this series of steps:

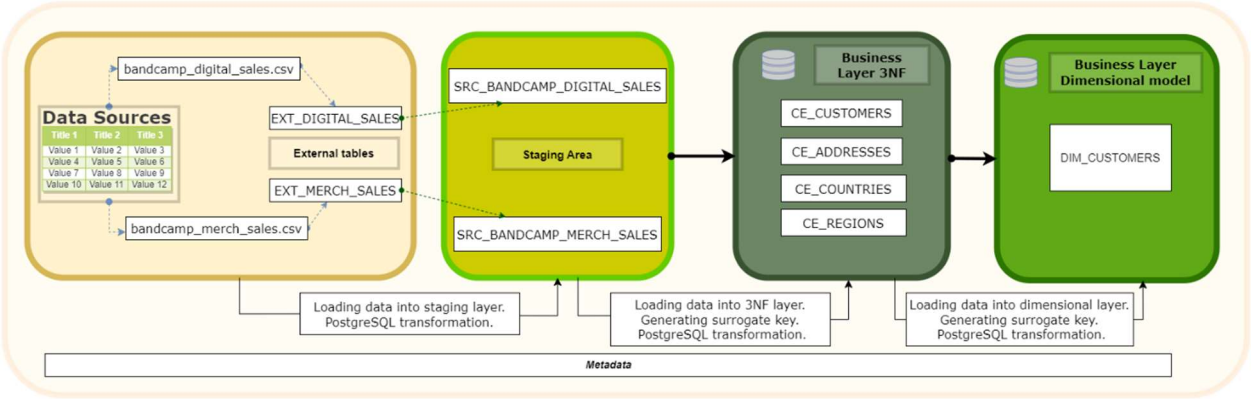


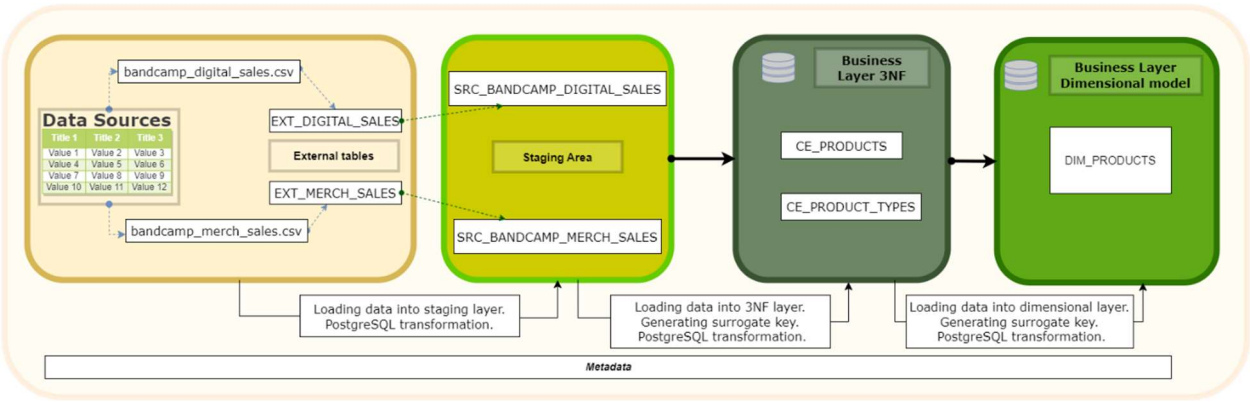
5 DATA FLOW

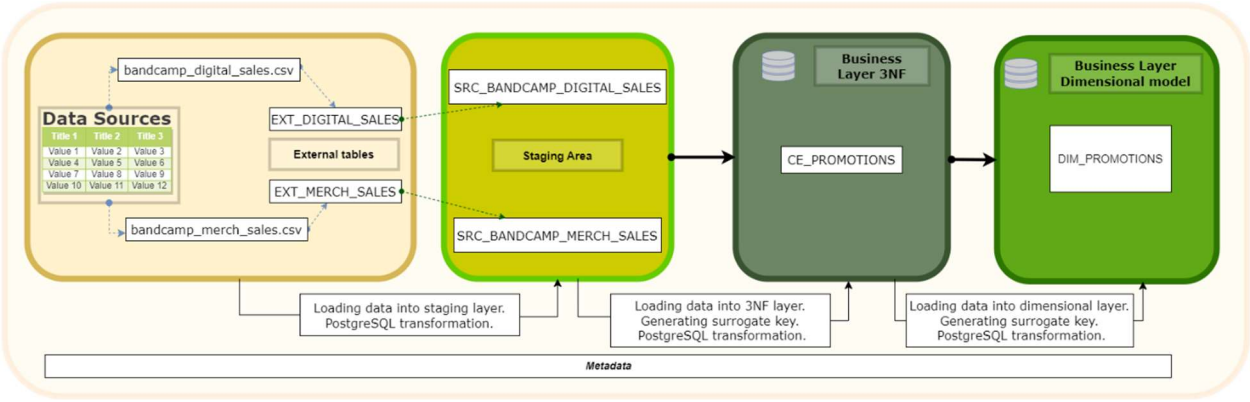
The next slides represent the data flow for the data warehouse loading process:

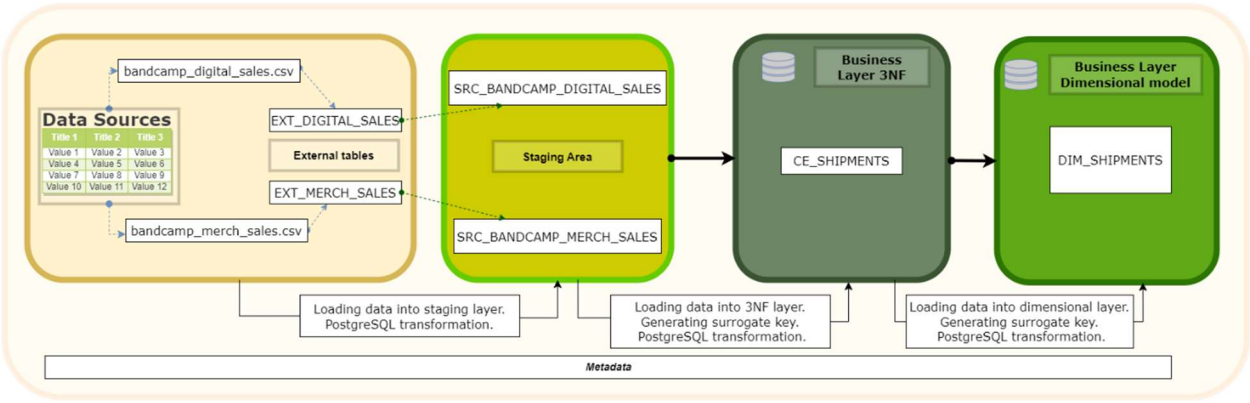


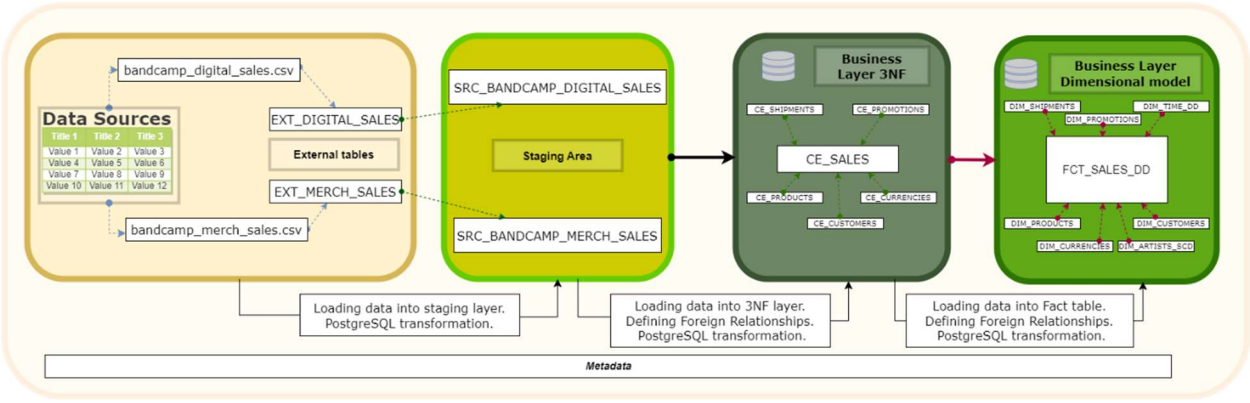





bandcamp_digital_sales.csv
Data Sources
EXT_DIGITAL_SALES
External tables
EXT_MERCH_SALES
bandcamp_merch_sales.csv
SRC_BANDCAMP_DIGITAL_SALES
Staging Area
SRC_BANDCAMP_MERCH_SALES
Business Layer 3NF
CE_PRODUCTS
CE_PRODUCT_TYPES
Business Layer Dimensional model
DIM_PRODUCTS
Loading data into staging layer. PostgreSQL transformation.
Loading data into 3NF layer. Generating surrogate key. PostgreSQL transformation.
Loading data into dimensional layer. Generating surrogate key. PostgreSQL transformation.
Metadata


bandcamp_digital_sales.csv
Data Sources
EXT_DIGITAL_SALES
External tables
EXT_MERCH_SALES
bandcamp_merch_sales.csv
SRC_BANDCAMP_DIGITAL_SALES
Staging Area
SRC_BANDCAMP_MERCH_SALES
Business Layer 3NF
CE_PROMOTIONS
Business Layer Dimensional model
DIM_PROMOTIONS
Loading data into staging layer. PostgreSQL transformation.
Loading data into 3NF layer. Generating surrogate key. PostgreSQL transformation.
Loading data into dimensional layer. Generating surrogate key. PostgreSQL transformation.
Metadata


bandcamp_digital_sales.csv
Data Sources
EXT_DIGITAL_SALES
External tables
EXT_MERCH_SALES
bandcamp_merch_sales.csv
SRC_BANDCAMP_DIGITAL_SALES
Staging Area
SRC_BANDCAMP_MERCH_SALES
Business Layer 3NF
CE_SHIPMENTS
Business Layer Dimensional model
DIM_SHIPMENTS
Loading data into staging layer. PostgreSQL transformation.
Loading data into 3NF layer. Generating surrogate key. PostgreSQL transformation.
Loading data into dimensional layer. Generating surrogate key. PostgreSQL transformation.
Metadata


bandcamp_digital_sales.csv
Data Sources
EXT_DIGITAL_SALES
External tables
EXT_MERCH_SALES
bandcamp_merch_sales.csv
SRC_BANDCAMP_DIGITAL_SALES
Staging Area
SRC_BANDCAMP_MERCH_SALES
Business Layer 3NF
CE_SHIPMENTS
CE_PROMOTIONS
CE_SALES
CE_PRODUCTS
CE_CURRENCIES
CE_CUSTOMERS
Business Layer Dimensional model
DIM_SHIPMENTS
DIM_TIME_DD
DIM_PROMOTIONS
FCT_SALES_DD
DIM_PRODUCTS
DIM_CUSTOMERS
DIM_CURRENCIES
DIM_ARTISTS_SCD
Loading data into staging layer. PostgreSQL transformation.
Loading data into 3NF layer. Defining Foreign Relationships. PostgreSQL transformation.
Loading data into Fact table. Defining Foreign Relationships. PostgreSQL transformation.
Metadata

6 FACT TABLE PARTITIONING STRATEGY

The fact table, **fct_sales_dd**, in the data warehouse on the BL_DM layer is designed with a partitioning strategy based on year and quarter. This means that the table is divided into multiple partitions, with each partition representing a specific combination of year and quarter.

In this particular strategy, there are four partitions created for each year, corresponding to the four quarters: Q1, Q2, Q3, and Q4. This allows for efficient data storage and retrieval by segregating the data based on its temporal characteristics.

Partitioning the fact table by year and quarter offers several benefits. It can enhance query performance by limiting the amount of data that needs to be scanned or accessed when retrieving information for a specific time period. It can also facilitate data maintenance operations such as loading, archiving, or purging data, as these operations can be targeted at specific partitions rather than the entire table.