

## New Prediction

Data analysis guides and step-by-step tutorials



# Know Your Customers: Step-by-Step Guide to Customer Segmentation Analysis with SQL (Plus The Code To Get Started)

by [Brian Graves](#) / [1 Comment](#)




Today I'm sharing three steps to create a customer segmentation analysis from scratch using SQL.

Customer segmentation with SQL is super important to level up your marketing campaign game. By following this guide, you'll unlock the power of SQL to better understand your customers. That

means you can create ads and campaigns that really speak to them.

And guess what? Happier customers mean more sales for your business!

***Thanks to Gigasheet for sponsoring this post.***

	<p><b>Data too big for Excel? Try Gigasheet.</b></p> <p>Quickly sort, filter, and group multi-gigabyte files with no database or coding. From huge CSVs to complex JSON files, Gigasheet makes exploratory data analysis easy. <a href="#">Sign up and get 3GB free.</a></p>
---	--

I've personally used segmentation in large-scale marketing campaigns for personalization and really super-targeted email campaigns. I've also used it to figure out which customers are most profitable so that my clients can offer discounts and rewards.

But built-in vendor dashboards really suck at this.

## If you don't have control over your data with SQL, you're missing out.

- Wasting time and money on generic marketing campaigns
- Stuck with basic, vendor dashboards with built-in segmentation
- Failing to automate and scale your data-driven marketing efforts
- Trusting your gut instead of relying on the data to make decisions

Fortunately, there's a better way.

This guide will take you step-by-step through the entire process, including the code you need to be successful. By the end of the guide, you'll be out of "Excel hell" and understand your customers better than anyone on your team.

Take a deep breath and let's dive in together:

## Step 1: Set up the tables

First, let's build a solid foundation for your data-driven marketing journey.

Here's the SQL code to set up a sample set of customer records that would be useful for customer segmentation.

Your specific tables will be different, but I created an [online database for you to work with for free](#).

```
-- Table 1: customers
CREATE TABLE customers (
  customer_id SERIAL PRIMARY KEY,
  first_name VARCHAR(255) NOT NULL,
  last_name VARCHAR(255) NOT NULL,
  email VARCHAR(255) UNIQUE NOT NULL,
  age INT,
  gender VARCHAR(50),
  location VARCHAR(255),
  education VARCHAR(255),
  income DECIMAL(10, 2),
  channel_preference VARCHAR(255),
  lifecycle_stage VARCHAR(255)
);

-- Table 2: customer_behavior
CREATE TABLE customer_behavior (
  behavior_id SERIAL PRIMARY KEY,
  customer_id INT NOT NULL,
  visit_date DATE NOT NULL,
  pages_visited INT,
  time_spent INT,
  FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);

-- Table 3: customer_purchase_history
CREATE TABLE customer_purchase_history (
  purchase_id SERIAL PRIMARY KEY,
  customer_id INT NOT NULL,
  order_date DATE NOT NULL,
  product_category VARCHAR(255),
  brand VARCHAR(255),
  order_value DECIMAL(10, 2),
  FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);

-- Table 4: customer_reviews
CREATE TABLE customer_reviews (
```

```

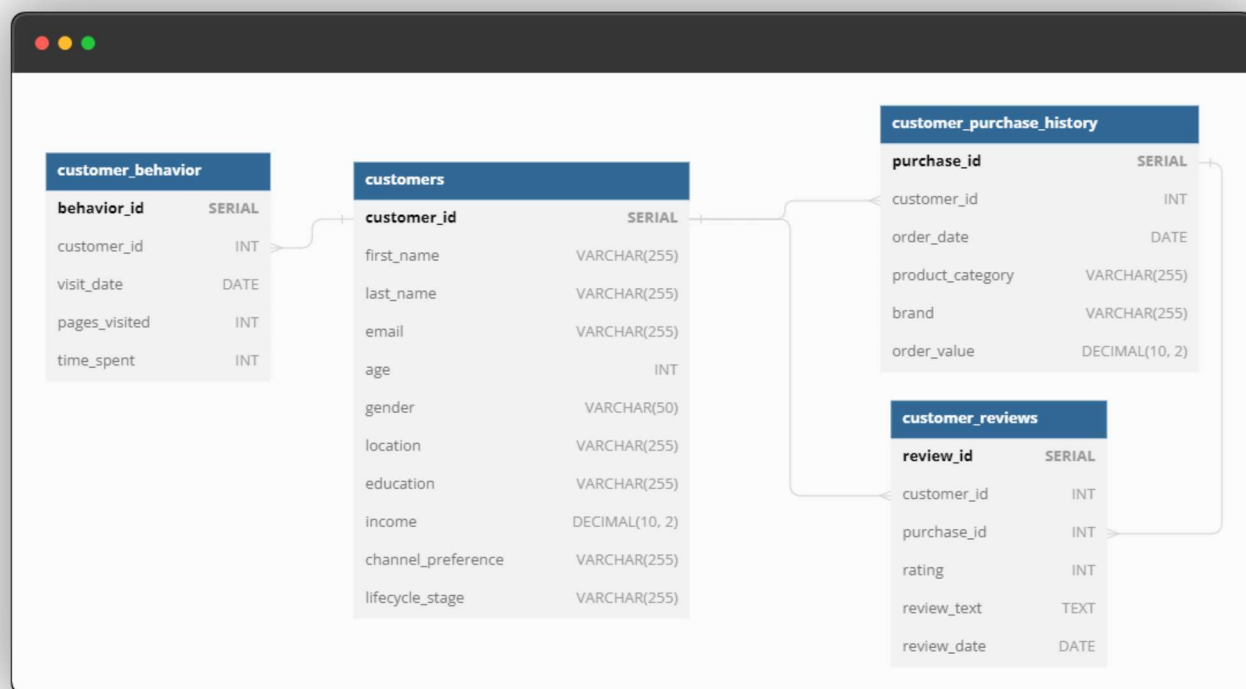
review_id SERIAL PRIMARY KEY,
customer_id INT NOT NULL,
purchase_id INT NOT NULL,
rating INT,
review_text TEXT,
review_date DATE,
FOREIGN KEY (customer_id) REFERENCES customers(customer_id),
FOREIGN KEY (purchase_id) REFERENCES
customer_purchase_history(purchase_id)
);

```

Here's what's inside:

1. **customers**: Demographics, channel preference, and customer lifecycle stage.
2. **customer\_behavior**: Browsing behavior.
3. **customer\_purchase\_history**: Purchase history, including product categories, brands, and order value.
4. **customer\_reviews**: Customer feedback and reviews.

And here's what it looks like visually:



## Step 2: Add your data

Next, we'll collect and store your most valuable customer data.

You have two options here:

- Option 1: Import from a CSV file
- Option 2: Import using INSERT statements

If you have your data in a CSV file, then go with option 1. If not, go with option 2.

### Option 1: Import from a CSV file:

1. Create a CSV file named `customers.csv` with your customer data (sample below):

```
first_name,last_name,email,age,gender,location,education,income,channel_preference,lifecycle_stage
John,Doe,john.doe@example.com,35,Male,New York,Bachelor,55000,Email,Active
Jane,Smith,jane.smith@example.com,28,Female,Los Angeles,Master,75000,Social Media,Active
Tom,Brown,tom.brown@example.com,42,Male,Chicago,PhD,90000,Email,At Risk
Sara,Johnson,sara.johnson@example.com,25,Female,Miami,Bachelor,48000,Mobile,New
Mike,Williams,mike.williams@example.com,37,Male,San Francisco,Master,110000,Mobile,Active
```

2. Connect to your PostgreSQL database using a tool like `psql`, `pgAdmin`, or any other client you prefer. I wrote more about this process recently.

3. Make sure the CSV file is accessible from the location where you are running the PostgreSQL client.

4. Run the following command to import the CSV data into the `customers` table:

```
COPY customers (first_name, last_name, email, age, gender, location, education, income, channel_preference, lifecycle_stage)
```

```
FROM '/path/to/customers.csv'  
DELIMITER ','  
CSV HEADER;
```

Remember to replace `/path/to/customers.csv` with the actual file path to your CSV file.

Repeat these steps for the other tables (`customer_behavior`, `customer_purchase_history`, and `customer_reviews`) by creating CSV files for each one and then running the `COPY` command for each table, updating the table name, column names, and file path accordingly.

## Option 2: Import using INSERT statements

Sometimes it can be easier just to write the code to insert records directly.

For example, if you already have your data in another database, you can write code to insert the records directly instead of using CSV (which is slower).

Here's the code to insert some sample records ([online example](#)):

```
-- Insert customer records  
INSERT INTO customers (first_name, last_name, email, age, gender, location,  
education, income, channel_preference, lifecycle_stage)  
VALUES ('John', 'Doe', 'john.doe@example.com', 35, 'Male', 'New York',  
'Bachelor', 55000, 'Email', 'Active');  
  
INSERT INTO customers (first_name, last_name, email, age, gender, location,  
education, income, channel_preference, lifecycle_stage)  
VALUES ('Jane', 'Smith', 'jane.smith@example.com', 28, 'Female', 'Los  
Angeles', 'Master', 75000, 'Social Media', 'Active');  
  
INSERT INTO customers (first_name, last_name, email, age, gender, location,  
education, income, channel_preference, lifecycle_stage)  
VALUES ('Tom', 'Brown', 'tom.brown@example.com', 42, 'Male', 'Chicago',  
'PhD', 90000, 'Email', 'At Risk');  
  
INSERT INTO customers (first_name, last_name, email, age, gender, location,  
education, income, channel_preference, lifecycle_stage)  
VALUES ('Sara', 'Johnson', 'sara.johnson@example.com', 25, 'Female', 'Miami',  
'Bachelor', 48000, 'Mobile', 'New');  
  
INSERT INTO customers (first_name, last_name, email, age, gender, location,  
education, income, channel_preference, lifecycle_stage)  
VALUES ('Mike', 'Williams', 'mike.williams@example.com', 37, 'Male', 'San  
Francisco', 'Master', 110000, 'Mobile', 'Active');  
  
-- Insert customer behavior records
```

```
INSERT INTO customer_behavior (customer_id, visit_date, pages_visited,
time_spent)
VALUES (1, '2023-04-01', 5, 300);

INSERT INTO customer_behavior (customer_id, visit_date, pages_visited,
time_spent)
VALUES (2, '2023-04-02', 8, 400);

-- Insert customer purchase history records
INSERT INTO customer_purchase_history (customer_id, order_date,
product_category, brand, order_value)
VALUES (1, '2023-03-29', 'Electronics', 'Apple', 1200.00);

INSERT INTO customer_purchase_history (customer_id, order_date,
product_category, brand, order_value)
VALUES (2, '2023-03-30', 'Fashion', 'Nike', 200.00);

INSERT INTO customer_purchase_history (customer_id, order_date,
product_category, brand, order_value)
VALUES (3, '2023-03-27', 'Home Appliances', 'Samsung', 600.00);

INSERT INTO customer_purchase_history (customer_id, order_date,
product_category, brand, order_value)
VALUES (4, '2023-03-25', 'Fashion', 'Adidas', 150.00);

-- Insert customer review records
INSERT INTO customer_reviews (customer_id, purchase_id, rating, review_text,
review_date)
VALUES (1, 1, 5, 'Great product and fast delivery!', '2023-03-31');

INSERT INTO customer_reviews (customer_id, purchase_id, rating, review_text,
review_date)
VALUES (2, 2, 4, 'Nice shoes, but delivery was slow.', '2023-04-04');

INSERT INTO customer_reviews (customer_id, purchase_id, rating, review_text,
review_date)
VALUES (3, 3, 3, 'Product works well, but packaging was damaged.', '2023-03-
29');

INSERT INTO customer_reviews (customer_id, purchase_id, rating, review_text,
review_date)
VALUES (4, 4, 4, 'Good quality clothing and comfortable fit.', '2023-03-28');
```

## Step 3: Develop some insights

After that, we'll hunt for some "hidden gems" by analyzing the data.

This step can take some time, thinking and "playing around" with the data. But after a while, [you'll get the hang of it as your SQL skills develop](#).

Here are two example insights based on the data ([follow along online here if you don't want to set up your own database](#))

**Insight #1:** Most valuable customers based on their total order value.

SQL code:

```
SELECT
    customers.customer_id,
    customers.first_name,
    customers.last_name,
    SUM(customer_purchase_history.order_value) AS total_order_value
FROM
    customers
    JOIN customer_purchase_history ON customers.customer_id =
customer_purchase_history.customer_id
GROUP BY
    customers.customer_id
ORDER BY
    total_order_value DESC
```

Explanation:

This query will calculate the total order value for each customer by joining the `customers` and `customer_purchase_history` tables on the `customer_id` column and summing the `order_value` column. The results are then grouped by customer and ordered by total order value in descending order.

customer_id	first_name	last_name	total_order_value
1	John	Doe	1200.00
3	Tom	Brown	600.00
2	Jane	Smith	200.00
4	Sara	Johnson	150.00

With this information, you could develop a customer loyalty program and offer discounts and other perks for top customers.

**Insight #2:** Most popular product categories among different age groups.



## SQL code:

```
SELECT
    customers.age,
    customer_purchase_history.product_category,
    COUNT(*) AS total_purchases
FROM
    customers
    JOIN customer_purchase_history
        ON customers.customer_id = customer_purchase_history.customer_id
WHERE
    customer_purchase_history.order_date >= '2023-01-01'
GROUP BY
    customers.age,
    customer_purchase_history.product_category
ORDER BY
    customers.age,
    total_purchases DESC
```

## Explanation:

This query shows the total purchases by product category for each age group by joining the `customers` and `customer_purchase_history` tables on the `customer_id` column and filtering for purchases made within a certain time period.

The results are then grouped by age and product category and ordered by age and total purchases in descending order.

age	product_category	total_purchases
25	Fashion	1
28	Fashion	1
35	Electronics	1
42	Home Appliances	1

## Now it's your turn:

1. [Use the online SQL provided](https://newprediction.com/customer-segmentation-with-sql/)

2. Develop insights using the `customer_reviews` table

3. [Share your insights on twitter](#)

That's it!

Follow these steps to get rid of guesswork in your digital marketing campaigns:

1. Set up the tables: Build a solid foundation for your data.
2. Add your data: Power up your database with customer info.
3. Develop insights: Find hidden gems to boost your strategies.

## 1 thought on “Know Your Customers: Step-by-Step Guide to Customer Segmentation Analysis with SQL (Plus The Code To Get Started)”

Pingback: Data Skills Into Dollars: Kickstart Your Freelancing Journey With These 5 In-Demand Data Analytics Services - New Prediction

### Leave a Reply

Your email address will not be published. Required fields are marked \*

Name \*

Email \*

Website

Comment \*

**Post Comment**

## More Guides & Tutorials



**BUILD YOUR PORTFOLIO**

**How To Write About Your Data Analytics Projects Without Being Boring - Even If You Don't Know What To Say**


 @NewPrediction



**SQL FOR DIGITAL MARKETERS**

**SQL Case Study: Create A Website Visitor Analysis From Scratch Using SQL - Including The Code You Need To Be Successful**

 @NewPrediction



**STEP-BY-STEP SQL CASE STUDY**

**Stop Losing Customers Now: Master the Art of Customer Churn Analysis Using PostgreSQL (Sample Code Included)**

 @NewPrediction



**SQL FOR DIGITAL MARKETERS**

**3 Simple Steps To Get Started With Social Media Analytics: A Step-by-Step SQL Case Study With Example Code**

 @NewPrediction





[Terms of Service](#) – [Privacy Policy](#)

**Get started on your data analytics career journey:**

The Data Analytics Portfolio Playbook

Solving with SQL: Master the Fundamentals of SQL in 30 Days So You Can Solve Real-World Business Problems and Increase Your Earning Potential

**Starting with Data?**  
**Get my weekly Data Analytics newsletter**

One actionable data analytics tip, delivered every Saturday.

First Name

Best email

**Subscribe For Free**

I hate spam as much as you do.

Unsubscribe any time.

Neve | Powered by WordPress