Lab: Gaining Insight with Sentiment Analysis

In this lab, you will use Hive's text processing features to analyze customers' comments and product ratings. You will uncover problems and propose potential solutions.

IMPORTANT: Since this lab builds on the previous one, it is important that you successfully complete the previous lab before starting this lab.

Background Information

Customer ratings and feedback are great sources of information for both customers and retailers like Dualcore. However, customer comments are typically free-form text and must be handled differently. Fortunately, Hive provides extensive support for text processing.

Step #1: Analyze Numeric Product Ratings

Before delving into text processing, you will begin by analyzing the numeric ratings customers have assigned to various products.

1. Change to the directory for this lab:

```
$ cd ~/training_materials/analyst/exercises/sentiment
```

2. Start Hive and use the `DESCRIBE` command to remind yourself of the table's structure.

3. We want to find the product that customers like most, but must guard against being misled by products that have few ratings assigned. Run the following query to find the product with the highest average among all those with at least 50 ratings:

```
SELECT prod_id, FORMAT_NUMBER(avg_rating, 2) AS
avg_rating
    FROM (SELECT prod_id, AVG(rating) AS avg_rating,
            COUNT(*) AS num
            FROM ratings
            GROUP BY prod_id) rated
    WHERE num >= 50
    ORDER BY avg_rating DESC
    LIMIT 1;
```

4. Rewrite, and then execute, the query above to find the product with the *lowest* average among products with at least 50 ratings. You should see that the result is product ID 1274673 with an average rating of 1.10. Store this query in `rated_lowest.hql`.

Step #2: Analyze Rating Comments

We observed earlier that customers are very dissatisfied with one of the products that Dualcore sells. Although numeric ratings can help identify *which* product that is, they don't tell Dualcore *why* customers don't like the product. We could simply read through all the comments associated with that product to learn this information, but that approach doesn't scale. Next, you will use Hive's text processing support to analyze the comments.

1. The following query normalizes all comments on that product to lowercase, breaks them into individual words using the `SENTENCES` function, and passes those to the `NGRAMS` function to find the five most common bigrams (two-word combinations). Run the query in Hive:

```
SELECT EXPLODE(NGRAMS(SENTENCES(LOWER(message)), 2, 5))
    AS bigrams
    FROM ratings
    WHERE prod_id = 1274673;
```

2. Most of these words are too common to provide much insight, though the word “expensive” does stand out in the list. Modify the previous query to find the five most common *trigrams* (three-word combinations), and then run that query in Hive. Store the query in `trigrams.hql`.

3. Among the patterns you see in the result is the phrase “ten times more.” This might be related to the complaints that the product is too expensive. Now that you’ve identified a specific phrase, look at a few comments that contain it by running this query:

```
SELECT message
     FROM ratings
     WHERE prod_id = 1274673
       AND message LIKE '%ten times more%'
     LIMIT 3;
```

   You should see three comments that say, “Why does the red one cost ten times more than the others?”

4. We can infer that customers are complaining about the price of this item, but the comment alone doesn’t provide enough detail. One of the words (“red”) in that comment was also found in the list of trigrams from the earlier query. Write and execute a query that will find all distinct comments containing the word “red” that are associated with product ID 1274673. Store this query in `messages_red.hql`.

5. The previous step should have displayed two comments:
   - What is so special aboutred?”
   - “Why does the red one cost ten times more than the others?”

   The second comment implies that this product is overpriced relative to similar products. Write and run a query that will display the record for product ID 1274673 in the `products` table. Store this query in `find_red_product.hql`.

6. Your query should have shown that the product was a "16GB USB Flash Drive (Red)" from the "Orion" brand. Next, run this query to identify similar products:

```
SELECT *
    FROM products
    WHERE name LIKE '%16 GB USB Flash Drive%'
      AND brand='Orion';
```

The query results show that there are three almost identical products, but the product with the negative reviews (the red one) costs about ten times as much as the others, just as some of the comments said.

Based on the cost and price columns, it appears that doing text processing on the product ratings has helped Dualcore uncover a pricing error.

This is the end of the lab.