Homework 5 - (Cohort 2 offset)

Homework 5: Expanding your Database

- Please upload your completed assignments to Google Drive.
- Due on Thursday, March 14 at 11:59pm
- Weight: 8% of total grade
- Upload one .sql file with your queries

String manipulations

1. Some product names in the product table have descriptions like "Jar" or "Organic". These are separated from the product name with a hyphen. Create a column using SUBSTR (and a couple of other commands) that captures these, but is otherwise NULL. Remove any trailing or leading whitespaces. Don't just use a case statement for each product!

product_name	description
Habanero Peppers - Organic	Organic

HINT: you might need to use INSTR(product_name,'-') to find the hyphens. INSTR will help split the column.

UNION

1. Using a UNION, write a query that displays the market dates with the highest and lowest total sales.

HINT: There are a possibly a few ways to do this query, but if you're struggling, try the following: 1) Create a CTE/Temp Table to find sales values grouped dates; 2) Create another CTE/Temp table with a rank windowed function on the previous query to create "best day" and "worst day"; 3) Query the second temp table twice, once for the best day, once for the worst day, with a UNION binding them.

Cross Join

1. Suppose every vendor in the `vendor_inventory` table had 5 of each of their products to sell to **every** customer on record. How much money would each vendor make per product? Show this by vendor_name and product name, rather than using the IDs.

HINT: Be sure you select only relevant columns and rows. Remember, CROSS JOIN will explode your table rows, so CROSS JOIN should likely be a subquery. Think a bit about the row counts: how many

distinct vendors, product names are there (x)? How many customers are there (y). Before your final group by you should have the product of those two queries (x*y).

2. Oh no! The price for banana peppers and sweet corn changed over time. If you didn't fix this in Question 1, fix it here (Hey, even instructors can miss something! 😅).

INSERT

1. Create a new table "product_units". This table will contain only products where the `product_qty_type = 'unit'`. It should use all of the columns from the product table, as well as a new column for the `CURRENT_TIMESTAMP`. Name the timestamp column `snapshot_timestamp`.
2. Using `INSERT`, add a new row to the table (with an updated timestamp). This can be any product you desire (e.g. add another record for Apple Pie).

DELETE

1. Delete the older record for the whatever product you added.

HINT: If you don't specify a WHERE clause, you are going to have a bad time.

UPDATE

1. We want to add the current_quantity to the product_units table. First, add a new column, `current_quantity` to the table using the following syntax.

```
ALTER TABLE product_units
ADD current_quantity INT;
```

Then, using `UPDATE`, change the current_quantity equal to the **last** `quantity` value from the vendor_inventory details.

HINT: This one is pretty hard. First, determine how to get the "last" quantity per product. Second, coalesce null values to 0 (if you don't have null values, figure out how to rearrange your query so you do.) Third, `SET current_quantity = (...your select statement...)`, remembering that WHERE can only accommodate one column. Finally, make sure you have a WHERE statement to update the right row, you'll need to use `product_units.product_id` to refer to the correct row within the product_units table. When you have all of these components, you can run the update statement.