Cleaning a PostgreSQL Database

In this project, you will work with data from a hypothetical Super Store to challenge and enhance your SQL skills in data cleaning. This project will engage you in identifying top categories based on the highest profit margins and detecting missing values, utilizing your comprehensive knowledge of SQL concepts.

Data Dictionary:

`orders`:

Column	Definition	Data type	Comments
`row_id`	Unique Record ID	`INTEGER`	
`order_id`	Identifier for each order in table	`TEXT`	Connects to `order_id` in `returned_orders` table
`order_date`	Date when order was placed	`TEXT`	
`market`	Market order_id belongs to	`TEXT`	
`region`	Region Customer belongs to	`TEXT`	Connects to `region` in `people` table
`product_id`	Identifier of Product bought	`TEXT`	Connects to `product_id` in `products` table
`sales`	Total Sales Amount for the Line Item	`DOUBLE PRECISION`	
`quantity`	Total Quantity for the Line Item	`DOUBLE PRECISION`	
`discount`	Discount applied for the Line Item	`DOUBLE PRECISION`	
`profit`	Total Profit earned on the Line Item	`DOUBLE PRECISION`	

`returned_orders`:

Column	Definition	Data type
`returned`	Yes values for Order / Line Item Returned	`TEXT`
`order_id`	Identifier for each order in table	`TEXT`
`market`	Market order_id belongs to	`TEXT`

`people`:

Column	Definition	Data type
`person`	Name of Salesperson credited with Order	`TEXT`
`region`	Region Salesperson in operating in	`TEXT`

`products`:

Column	Definition	Data type
`product_id`	Unique Identifier for the Product	`TEXT`
`category`	Category Product belongs to	`TEXT`
`sub_category`	Sub Category Product belongs to	`TEXT`
`product_name`	Detailed Name of the Product	`TEXT`

As you can see in the Data Dictionary above, date fields have been written to the `orders` table as `TEXT` and numeric fields like sales, profit, etc. have been written to the `orders` table as `Double Precision`. You will need to take care of these types in some of the queries. This project is an excellent opportunity to apply your SQL skills in a practical setting and gain valuable experience in data cleaning and analysis. Good luck, and happy querying!

Projects Data DataFrame as `top_five_products_each_category`

```sql
-- top_five_products_each_category
WITH ranked AS (
    SELECT
        category,
        product_name,
        ROUND(SUM(sales)::NUMERIC, 2) AS product_total_sales,
        ROUND(SUM(profit)::NUMERIC, 2) AS product_total_profit,
        RANK() OVER (PARTITION BY category ORDER BY SUM(sales) DESC) AS product_rank
    FROM orders o
    JOIN products p
        USING (product_id)
    GROUP BY category, product_name
)
SELECT *
FROM ranked
WHERE product_rank <= 5
ORDER BY category ASC, product_total_profit DESC, product_rank
```

in…	category	product_name	product_total_sales
0	Furniture	Harbour Creations Executive Leather Armchair, Adjustable	501
1	Furniture	SAFCO Executive Leather Armchair, Black	419
2	Furniture	Hon Executive Leather Armchair, Adjustable	581
3	Furniture	Novimex Executive Leather Armchair, Adjustable	405
4	Furniture	Office Star Executive Leather Armchair, Adjustable	51
5	Office Supplies	Hoover Stove, Red	326
6	Office Supplies	Eldon File Cart, Single Width	398
7	Office Supplies	Smead Lockers, Industrial	289
8	Office Supplies	Rogers File Cart, Single Width	295
9	Office Supplies	Hoover Stove, White	32
10	Technology	Canon imageCLASS 2200 Advanced Copier	615
11	Technology	Cisco Smart Phone, Full Size	764
12	Technology	Motorola Smart Phone, Full Size	73
13	Technology	Nokia Smart Phone, Full Size	719
14	Technology	Apple Smart Phone, Full Size	869

Rows: 15 Expand

Projects Data DataFrame as impute_

```sql
-- impute_missing_values
WITH empty_quantity AS (
    SELECT
        product_id,
        discount,
        market,
        region,
        sales,
        quantity
    FROM orders
    WHERE quantity IS NULL
),
unit_price_calculation AS (
    SELECT
        product_id,
        discount,
        AVG(sales/quantity) AS unit_price
    FROM orders
    WHERE quantity IS NOT NULL
    GROUP BY product_id, discount
)
SELECT DISTINCT
    e.*,
    e.sales/unit_price AS calculated_quantity
FROM empty_quantity e
JOIN unit_price_calculation up
    ON e.product_id = up.product_id AND e.discount = up.discount
```

	product_id						calculated_quan...
0	FUR-ADV-10000571	0	EMEA	EMEA	438.96		4
1	FUR-ADV-10004395	0	EMEA	EMEA	84.12		2
2	FUR-BO-10001337	0.15	US	West	308.499		3
3	TEC-STA-10003330	0	Africa	Africa	506.64		2
4	TEC-STA-10004542	0	Africa	Africa	160.32		4

Rows: 5 Expand