ku4il1z3k

January 6, 2025

[4]:
```python
import csv
from datetime import datetime

# Step 1: Extract Data

def read_csv(file_path):
  with open(file_path, mode='r') as file:
    reader = csv.DictReader(file)
    data = [row for row in reader]
    return data

# Step 2: Transform Data
def clean_sales_data(sales):
    """Cleans sales data by removing duplicates and handling missing values"""
    seen = set()
    cleaned_data = []
    for sale in sales:
      sale_id = sale['sale_id'] # Check for duplicates and missing values
      if sale_id not in seen and all(sale.values()):
        seen.add(sale_id)
        cleaned_data.append(sale)
    return cleaned_data


def integrate_data(sales, products, customers):
    """
    Integrates sales data with product and customer information.
    """
    product_lookup = {p['product_id']: p for p in products}
    customer_lookup = {c['customer_id']: c for c in customers}

    integrated_data = []
    for sale in sales:
        product = product_lookup.get(sale['product_id'], {})
        customer = customer_lookup.get(sale['customer_id'], {})

        if product and customer:  # Ensure both mappings exist#
```

```
            # Change the date format to '%m/%d/%Y' to match the data
            integrated_data.append({
              "sale_id": sale['sale_id'],
              "date": sale['date'],
              "customer_id": sale['customer_id'],
              "product_id": sale['product_id'],
              "quantity": int(sale["quantity"]),
              "price": float(sale['price']),
              "product_name": product['product_name'],
              "category": product['category'],
              "customer_name": customer['name'],
              "total_amount": int(sale['quantity']) * float(sale['price']),
              "day_of_week": datetime.strptime(sale['date'], '%m/%d/%Y').
 ↪strftime('%A'), # Updated format
              "month": datetime.strptime(sale['date'], '%m/%d/%Y').
 ↪strftime('%B'), # Updated format
              "quarter": (int(datetime.strptime(sale['date'], '%m/%d/%Y').month)␣
 ↪- 1) // 3 + 1 # Updated format
          })
    # Ensure that the `return` statement is properly aligned outside the loop
    return integrated_data

# Step 3: Load Data

def write_csv(file_path, data, headers):
    """Writes a list of dictionaries to a CSV file."""
    with open(file_path, mode='w', newline='') as file:
      writer = csv.DictWriter(file, fieldnames=headers)
      writer.writeheader()
      writer.writerows(data)

# Step 4: Analyze Data
def calculate_sales_by_category(sales):
  category_sales = {}
  for sale in sales:
    category = sale['category']
    amount = float(sale['total_amount'])
    category_sales[category] = category_sales.get(category, 0) + amount
  return category_sales


def identify_top_products(sales, n):
    """Identifies the top N products by total sales."""
    product_sales = {}
    for sale in sales:
        product = sale['product_name']
        amount = float(sale['total_amount'])
```

```
        product_sales[product] = product_sales.get(product, 0) + amount
        sorted_products = sorted(product_sales.items(), key=lambda x: x[1],␣
 ↪reverse=True)
    return sorted_products[:n]

def analyze_sales_trends(sales):
    """Analyzes sales trends over time."""
    monthly_sales = {}
    for sale in sales: # Corrected indentation
        month = sale['month']
        amount = float(sale['total_amount'])
        monthly_sales[month] = monthly_sales.get(month, 0) + amount
    return monthly_sales

def generate_summary_report(analysis):
    """Generates a summary report of the analysis."""
    report_lines=["sales_summary_report","=" *20]

    report_lines.append("\nSales by Catagory:")
    for category, amount in analysis['sales_by_category'].items():
        report_lines.append(f"{category}: ${amount:.2f}")

    report_lines.append("\nTop Product:")
    for product, amount in analysis['top_products']:
        report_lines.append(f"{product}: ${amount:.2f}")

    report_lines.append("\nMonthly Sales Trends:")
    for month, amount in analysis['monthly_sales_trends'].items():
        report_lines.append(f"{month}: ${amount:.2f}")

    return "\n".join(report_lines)

# Main Workflow

# File paths
customer_file = '/content/sample_data/Customer.csv'
employee_file = '/content/sample_data/employee.csv'
product_file = '/content/sample_data/Product.csv'
sales_file = '/content/sample_data/sales.csv'

# Extract
customers = read_csv(customer_file)
employees = read_csv(employee_file)
products = read_csv(product_file)
sales = read_csv(sales_file)

# Clean sales data
```

```
cleaned_sales=clean_sales_data(sales)


# Transform data
transformed_sales = integrate_data(cleaned_sales, products, customers)


# write transform data to CSV file
output_headers = list(transformed_sales[0].keys())
write_csv('processed_sales.csv', transformed_sales, output_headers)

# Analyze
top_products = identify_top_products(transformed_sales, n=5)
monthly_sales_trends = analyze_sales_trends(transformed_sales)
sales_by_category = calculate_sales_by_category(transformed_sales)

# Generate_& write summary_report
analysis = {
    'top_products': top_products,
    'monthly_sales_trends': monthly_sales_trends,
    'sales_by_category': sales_by_category
}
summary  = generate_summary_report(analysis)

with open('summary_report.txt', 'w') as report_file:
    report_file.write(summary)

# print summary to console
print(summary)
```

```
sales_summary_report
====================

Sales by Catagory:
Electronics: $12228.14
Home & Office: $4649.07
Stationery: $1439.10

Top Product:
Bluetooth Speaker: $9499.05
LED Desk Lamp: $4649.07
Wireless Mouse: $2729.09
Notebook Set: $1439.10

Monthly Sales Trends:
September: $4215.14
October: $4677.06
```

```
November: $4541.10
December: $4293.09
January: $589.92
```