```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "id": "b0f27e5c-ddb5-4dbf-b47a-867669b2991e",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "import matplotlib.pyplot as plt\n",
    "import seaborn as sns"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "a78b5af5-8158-43f2-bab7-d251b27d555a",
   "metadata": {},
   "source": [
    "## Loading the files"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "id": "78719d76-889b-487a-9299-fda4ae50ce93",
   "metadata": {},
   "outputs": [],
   "source": [
    "# Load CSV file\n",
    "purchase_behaviour = pd.read_csv('C:/Users/Lekhansh/Downloads/QVI_purchase_behaviour.csv')\n",
    "\n",
    "# Load XLSX file\n",
    "transaction_data = pd.read_excel('C:/Users/Lekhansh/Downloads/QVI_transaction_data.xlsx')\n"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "9de022f0-abb1-4559-a371-bbfd47d9b939",
   "metadata": {},
   "source": [
    "## Summary of the Data"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "id": "be81784d-4367-4ff6-a528-c722a0318297",
   "metadata": {
    "scrolled": true
   },
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "       LYLTY_CARD_NBR\n",
      "count    7.263700e+04\n",
      "mean     1.361859e+05\n",
      "std      8.989293e+04\n",
      "min      1.000000e+03\n",
      "25%      6.620200e+04\n",
      "50%      1.340400e+05\n",
      "75%      2.033750e+05\n",
      "max      2.373711e+06\n",
      "<class 'pandas.core.frame.DataFrame'>\n",
      "RangeIndex: 72637 entries, 0 to 72636\n",
      "Data columns (total 3 columns):\n",
      " #   Column            Non-Null Count  Dtype \n",
      "---  ------            --------------  ----- \n",
      " 0   LYLTY_CARD_NBR    72637 non-null  int64 \n",
      " 1   LIFESTAGE         72637 non-null  object\n",
      " 2   PREMIUM_CUSTOMER  72637 non-null  object\n",
      "dtypes: int64(1), object(2)\n",
      "memory usage: 1.7+ MB\n",
      "None\n",
      "                DATE     STORE_NBR  LYLTY_CARD_NBR        TXN_ID  \\\n",
      "count  264836.000000  264836.00000    2.648360e+05  2.648360e+05   \n",
      "mean    43464.036260     135.08011    1.355495e+05  1.351583e+05   \n",
      "std       105.389282      76.78418    8.057998e+04  7.813303e+04   \n",
      "min     43282.000000       1.00000    1.000000e+03  1.000000e+00   \n",
      "25%     43373.000000      70.00000    7.002100e+04  6.760150e+04   \n",
      "50%     43464.000000     130.00000    1.303575e+05  1.351375e+05   \n",
      "75%     43555.000000     203.00000    2.030942e+05  2.027012e+05   \n",
      "max     43646.000000     272.00000    2.373711e+06  2.415841e+06   \n",
      "\n",
      "            PROD_NBR       PROD_QTY      TOT_SALES  \n",
      "count  264836.000000  264836.000000  264836.000000  \n",
      "mean       56.583157       1.907309       7.304200  \n",
      "std        32.826638       0.643654       3.083226  \n",
      "min         1.000000       1.000000       1.500000  \n",
      "25%        28.000000       2.000000       5.400000  \n",
      "50%        56.000000       2.000000       7.400000  \n",
      "75%        85.000000       2.000000       9.200000  \n",
      "max       114.000000     200.000000     650.000000  \n",
      "<class 'pandas.core.frame.DataFrame'>\n",
      "RangeIndex: 264836 entries, 0 to 264835\n",
      "Data columns (total 8 columns):\n",
      " #   Column          Non-Null Count   Dtype  \n",
      "---  ------          --------------   -----  \n",
      " 0   DATE            264836 non-null  int64  \n",
      " 1   STORE_NBR       264836 non-null  int64  \n",
      " 2   LYLTY_CARD_NBR  264836 non-null  int64  \n",
      " 3   TXN_ID          264836 non-null  int64  \n",
      " 4   PROD_NBR        264836 non-null  int64  \n",
```

```
     " 5   PROD_NAME       264836 non-null  object \n",
     " 6   PROD_QTY        264836 non-null  int64  \n",
     " 7   TOT_SALES       264836 non-null  float64\n",
     "dtypes: float64(1), int64(6), object(1)\n",
     "memory usage: 16.2+ MB\n",
     "None\n"
    ]
   }
  ],
  "source": [
   "# Summarize the purchase behaviour data\n",
   "print(purchase_behaviour.describe())\n",
   "print(purchase_behaviour.info())\n",
   "\n",
   "# Summarize the transaction data\n",
   "print(transaction_data.describe())\n",
   "print(transaction_data.info())"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "c9e8da0d-9b42-42de-98c6-488f6efb6208",
  "metadata": {},
  "source": [
   "Purchase Behaviour Data\n",
   "1. The LYLTY_CARD_NBR variable shows a wide range, with a mean and median that suggest a fairly uniform distribution but with significant variability.\n",
   "2. The data contains only non-null values and seems to be well-structured for further analysis.\n",
   "\n",
   "Transaction Data\n",
   "1. The TOT_SALES and PROD_QTY variables show relatively low mean values with a wide range, indicating some high-value transactions.\n",
   "2. The PROD_NBR and STORE_NBR show a broad range, indicating multiple products and stores involved.\n",
   "3. The data is dense, suggesting a large number of transactions, making it suitable for detailed analysis and trend identification."
  ]
 },
 {
  "cell_type": "markdown",
  "id": "1de50209-f268-431f-ab83-743367f539b7",
  "metadata": {},
  "source": [
   "## Outlier Detection and Handling\n",
   "\n",
   "Now we will check for the outliers in columns \"TOT_SALES\" and \"PROD_QTY\""
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 4,
  "id": "a69045ec-af44-40e3-a719-64bafed423c2",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "image/png": "iVBORw0KGgoAAAANSUhEUgAAAx8AAAIhCAYAAAAvowfMAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjguMCwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy81sbWrAAA
     "text/plain": [
      "<Figure size 1000x600 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   },
   {
    "data": {
     "image/png": "iVBORw0KGgoAAAANSUhEUgAAAx8AAAIhCAYAAAAvowfMAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjguMCwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy81sbWrAAA
     "text/plain": [
      "<Figure size 1000x600 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   }
  ],
  "source": [
   "# Columns to check for outliers\n",
   "columns_to_check = ['TOT_SALES', 'PROD_QTY']\n",
   "\n",
   "# Plot box plots to visualize outliers\n",
   "for column in columns_to_check:\n",
   "    plt.figure(figsize=(10, 6))\n",
   "    sns.boxplot(x=transaction_data[column])\n",
   "    plt.title(f'Box Plot for {column}')\n",
   "    plt.show()\n"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 5,
  "id": "875036a9-dc66-4c97-8efe-2f2f8a156d71",
  "metadata": {},
  "outputs": [],
  "source": [
   "# Remove outliers in 'TOT_SALES'\n",
   "lower_bound_sales = transaction_data['TOT_SALES'].quantile(0.25) - 1.5 * (transaction_data['TOT_SALES'].quantile(0.75) - transaction_data['TOT_SALES'].qua
   "upper_bound_sales = transaction_data['TOT_SALES'].quantile(0.75) + 1.5 * (transaction_data['TOT_SALES'].quantile(0.75) - transaction_data['TOT_SALES'].qua
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 6,
  "id": "466231cc-358f-45d0-b1e7-3c0cb16ebd66",
  "metadata": {},
  "outputs": [],
  "source": [
   "# Cap outliers in 'TOT_SALES'\n",
   "transaction_data['TOT_SALES'] = np.clip(transaction_data['TOT_SALES'], lower_bound_sales, upper_bound_sales)"
  ]
 },
```

```
{
 "cell_type": "code",
 "execution_count": 9,
 "id": "c5173b6c-2475-45b6-989f-e7cc07bba370",
 "metadata": {},
 "outputs": [],
 "source": [
  "# Cap the outlier value to 10 (the upper range of normal values)\n",
  "transaction_data['PROD_QTY'] = np.clip(transaction_data['PROD_QTY'], None, 10)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 10,
 "id": "6b6afe33-9cb9-4e87-a417-f0a12cdb6c8c",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "image/png": "iVBORw0KGgoAAAANSUhEUgAAAx8AAAIhCAYAAAAvowfMAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjguMCwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy81sbWrAAA
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  },
  {
   "data": {
    "image/png": "iVBORw0KGgoAAAANSUhEUgAAAx8AAAIhCAYAAAAvowfMAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjguMCwgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy81sbWrAAA
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  }
 ],
 "source": [
  "# Columns to check for outliers\n",
  "columns_to_check = ['TOT_SALES', 'PROD_QTY']\n",
  "\n",
  "# Plot box plots to visualize outliers\n",
  "for column in columns_to_check:\n",
  "    plt.figure(figsize=(10, 6))\n",
  "    sns.boxplot(x=transaction_data[column])\n",
  "    plt.title(f'Box Plot for {column}')\n",
  "    plt.show()"
 ]
},
{
 "cell_type": "markdown",
 "id": "b8ec4fee-e3d6-4910-8d53-e41266345596",
 "metadata": {},
 "source": [
  "## Missing Values Check"
 ]
},
{
 "cell_type": "code",
 "execution_count": 12,
 "id": "3d293822-c2b4-4471-8a8e-c1cc2e378a9b",
 "metadata": {
  "scrolled": true
 },
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Missing values in purchase_behaviour table:\n",
    "LYLTY_CARD_NBR      0\n",
    "LIFESTAGE           0\n",
    "PREMIUM_CUSTOMER    0\n",
    "dtype: int64\n",
    "\n",
    "Missing values in transaction_data table:\n",
    "DATE              0\n",
    "STORE_NBR         0\n",
    "LYLTY_CARD_NBR    0\n",
    "TXN_ID            0\n",
    "PROD_NBR          0\n",
    "PROD_NAME         0\n",
    "PROD_QTY          0\n",
    "TOT_SALES         0\n",
    "dtype: int64\n"
   ]
  }
 ],
 "source": [
  "# Checking missing values in purchase_behaviour table\n",
  "print(\"Missing values in purchase_behaviour table:\")\n",
  "missing_values_purchase_behaviour = purchase_behaviour.isnull().sum()\n",
  "print(missing_values_purchase_behaviour)\n",
  "\n",
  "# Checking missing values in transaction_data table\n",
  "print(\"\\nMissing values in transaction_data table:\")\n",
  "missing_values_transaction_data = transaction_data.isnull().sum()\n",
  "print(missing_values_transaction_data)"
 ]
},
{
 "cell_type": "markdown",
 "id": "e95d351c-0f78-429d-8239-efab9542b0e8",
 "metadata": {},
 "source": [
```

```
  "## Additional Data Cleaning\n",
  "\n",
  "1. Data Formatting\n",
  "2. Correction of Date Column - from string to date format\n",
  "3. Removal of unnecessary values"
 ]
},
{
 "cell_type": "markdown",
 "id": "28b014ce-dd46-41e3-aa95-b0e9f21830b9",
 "metadata": {},
 "source": [
  "As we can see there are no missing values, but we will check the format of the data"
 ]
},
{
 "cell_type": "code",
 "execution_count": 15,
 "id": "4f3c57b5-7747-4c26-ae73-a94bedef9585",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Data types in transaction_data:\n",
    "DATE              int64\n",
    "STORE_NBR         int64\n",
    "LYLTY_CARD_NBR    int64\n",
    "TXN_ID            int64\n",
    "PROD_NBR          int64\n",
    "PROD_NAME        object\n",
    "PROD_QTY          int64\n",
    "TOT_SALES       float64\n",
    "dtype: object\n",
    "\n",
    "Data types in customer_data:\n",
    "LYLTY_CARD_NBR     int64\n",
    "LIFESTAGE         object\n",
    "PREMIUM_CUSTOMER  object\n",
    "dtype: object\n"
   ]
  }
 ],
 "source": [
  "# Check the data types (formats) of all variables in transaction_data\n",
  "print(\"Data types in transaction_data:\")\n",
  "print(transaction_data.dtypes)\n",
  "\n",
  "# Check the data types (formats) of all variables in customer_data\n",
  "print(\"\\nData types in customer_data:\")\n",
  "print(purchase_behaviour.dtypes)"
 ]
},
{
 "cell_type": "markdown",
 "id": "4a4f27a1-fc09-4d75-97ea-fb9d0b8c01a3",
 "metadata": {},
 "source": [
  "The DATE column is currently in integer format, so we need to convert it to a proper date format."
 ]
},
{
 "cell_type": "code",
 "execution_count": 16,
 "id": "c8bd5c72-1833-40c5-aaea-6b61f1476a8d",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "0   2018-10-17\n",
    "1   2019-05-14\n",
    "2   2019-05-20\n",
    "3   2018-08-17\n",
    "4   2018-08-18\n",
    "Name: DATE, dtype: datetime64[ns]\n"
   ]
  }
 ],
 "source": [
  "# Convert 'DATE' column to datetime format\n",
  "transaction_data['DATE'] = pd.to_datetime(transaction_data['DATE'], origin='1899-12-30', unit='D')\n",
  "\n",
  "# Check the conversion\n",
  "print(transaction_data['DATE'].head())"
 ]
},
{
 "cell_type": "markdown",
 "id": "3eb6cba9-7322-4014-af67-f3c8bebf629c",
 "metadata": {},
 "source": [
  "Exploring the PROD_NAME column to ensure only chip products are included."
 ]
},
{
 "cell_type": "code",
 "execution_count": 17,
 "id": "085dfcb5-eb2a-48d6-8404-da83cb5bcd5c",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
```

```
    "text": [
     "['Natural Chip        Compny SeaSalt175g' 'CCs Nacho Cheese    175g'\n",
     " 'Smiths Crinkle Cut  Chips Chicken 170g'\n",
     " 'Smiths Chip Thinly  S/Cream&Onion 175g'\n",
     " 'Kettle Tortilla ChpsHny&Jlpno Chili 150g'\n",
     " 'Old El Paso Salsa   Dip Tomato Mild 300g'\n",
     " 'Smiths Crinkle Chips Salt & Vinegar 330g'\n",
     " 'Grain Waves         Sweet Chilli 210g'\n",
     " 'Doritos Corn Chip Mexican Jalapeno 150g'\n",
     " 'Grain Waves Sour    Cream&Chives 210G'\n",
     " 'Kettle Sensations   Siracha Lime 150g' 'Twisties Cheese     270g'\n",
     " 'WW Crinkle Cut      Chicken 175g' 'Thins Chips Light&  Tangy 175g'\n",
     " 'CCs Original 175g' 'Burger Rings 220g'\n",
     " 'NCC Sour Cream &    Garden Chives 175g'\n",
     " 'Doritos Corn Chip Southern Chicken 150g' 'Cheezels Cheese Box 125g'\n",
     " 'Smiths Crinkle      Original 330g'\n",
     " 'Infzns Crn Crnchers Tangy Gcamole 110g'\n",
     " 'Kettle Sea Salt     And Vinegar 175g'\n",
     " 'Smiths Chip Thinly  Cut Original 175g' 'Kettle Original 175g'\n",
     " 'Red Rock Deli Thai  Chilli&Lime 150g' 'Pringles Sthrn FriedChicken 134g'\n",
     " 'Pringles Sweet&Spcy BBQ 134g' 'Red Rock Deli SR    Salsa & Mzzrlla 150g'\n",
     " 'Thins Chips         Originl saltd 175g'\n",
     " 'Red Rock Deli Sp    Salt & Truffle 150G'\n",
     " 'Smiths Thinly       Swt Chli&S/Cream175G' 'Kettle Chilli 175g'\n",
     " 'Doritos Mexicana    170g' 'Smiths Crinkle Cut  French OnionDip 150g'\n",
     " 'Natural ChipCo      Hony Soy Chckn175g'\n",
     " 'Dorito Corn Chp     Supreme 380g' 'Twisties Chicken270g'\n",
     " 'Smiths Thinly Cut   Roast Chicken 175g'\n",
     " 'Smiths Crinkle Cut  Tomato Salsa 150g'\n",
     " 'Kettle Mozzarella   Basil & Pesto 175g'\n",
     " 'Infuzions Thai SweetChili PotatoMix 110g'\n",
     " 'Kettle Sensations   Camembert & Fig 150g'\n",
     " 'Smith Crinkle Cut   Mac N Cheese 150g'\n",
     " 'Kettle Honey Soy    Chicken 175g' 'Thins Chips Seasonedchicken 175g'\n",
     " 'Smiths Crinkle Cut  Salt & Vinegar 170g'\n",
     " 'Infuzions BBQ Rib   Prawn Crackers 110g'\n",
     " 'GrnWves Plus Btroot & Chilli Jam 180g'\n",
     " 'Tyrrells Crisps     Lightly Salted 165g'\n",
     " 'Kettle Sweet Chilli And Sour Cream 175g'\n",
     " 'Doritos Salsa       Medium 300g' 'Kettle 135g Swt Pot Sea Salt'\n",
     " 'Pringles SourCream  Onion 134g' 'Doritos Corn Chips  Original 170g'\n",
     " 'Twisties Cheese     Burger 250g'\n",
     " 'Old El Paso Salsa   Dip Chnky Tom Ht300g'\n",
     " 'Cobs Popd Swt/Chlli &Sr/Cream Chips 110g'\n",
     " 'Woolworths Mild     Salsa 300g'\n",
     " 'Natural Chip Co     Tmato Hrb&Spce 175g'\n",
     " 'Smiths Crinkle Cut  Chips Original 170g'\n",
     " 'Cobs Popd Sea Salt  Chips 110g'\n",
     " 'Smiths Crinkle Cut  Chips Chs&Onion170g'\n",
     " 'French Fries Potato Chips 175g'\n",
     " 'Old El Paso Salsa   Dip Tomato Med 300g'\n",
     " 'Doritos Corn Chips  Cheese Supreme 170g'\n",
     " 'Pringles Original   Crisps 134g' 'RRD Chilli&         Coconut 150g'\n",
     " 'WW Original Corn    Chips 200g' 'Thins Potato Chips  Hot & Spicy 175g'\n",
     " 'Cobs Popd Sour Crm  &Chives Chips 110g'\n",
     " 'Smiths Crnkle Chip  Orgnl Big Bag 380g'\n",
     " 'Doritos Corn Chips  Nacho Cheese 170g'\n",
     " 'Kettle Sensations   BBQ&Maple 150g' 'WW D/Style Chip     Sea Salt 200g'\n",
     " 'Pringles Chicken    Salt Crips 134g' 'WW Original Stacked Chips 160g'\n",
     " 'Smiths Chip Thinly  CutSalt/Vinegr175g' 'Cheezels Cheese 330g'\n",
     " 'Tostitos Lightly    Salted 175g' 'Thins Chips Salt &  Vinegar 175g'\n",
     " 'Smiths Crinkle Cut  Chips Barbecue 170g' 'Cheetos Puffs 165g'\n",
     " 'RRD Sweet Chilli &  Sour Cream 165g' 'WW Crinkle Cut      Original 175g'\n",
     " 'Tostitos Splash Of  Lime 175g' 'Woolworths Medium   Salsa 300g'\n",
     " 'Kettle Tortilla ChpsBtroot&Ricotta 150g' 'CCs Tasty Cheese    175g'\n",
     " 'Woolworths Cheese   Rings 190g' 'Tostitos Smoked     Chipotle 175g'\n",
     " 'Pringles Barbeque   134g' 'WW Supreme Cheese   Corn Chips 200g'\n",
     " 'Pringles Mystery    Flavour 134g'\n",
     " 'Tyrrells Crisps     Ched & Chives 165g'\n",
     " 'Snbts Whlgrn Crisps Cheddr&Mstrd 90g' 'Cheetos Chs & Bacon Balls 190g'\n",
     " 'Pringles Slt Vingar 134g' 'Infuzions SourCream&Herbs Veg Strws 110g'\n",
     " 'Kettle Tortilla ChpsFeta&Garlic 150g'\n",
     " 'Infuzions Mango     Chutny Papadums 70g'\n",
     " 'RRD Steak &         Chimuchurri 150g' 'RRD Honey Soy       Chicken 165g'\n",
     " 'Sunbites Whlegrn    Crisps Frch/Onin 90g' 'RRD Salt & Vinegar  165g'\n",
     " 'Doritos Cheese      Supreme 330g' 'Smiths Crinkle Cut  Snag&Sauce 150g'\n",
     " 'WW Sour Cream &OnionStacked Chips 160g' 'RRD Lime & Pepper   165g'\n",
     " 'Natural ChipCo Sea  Salt & Vinegr 175g'\n",
     " 'Red Rock Deli Chikn&Garlic Aioli 150g'\n",
     " 'RRD SR Slow Rst     Pork Belly 150g' 'RRD Pc Sea Salt     165g'\n",
     " 'Smith Crinkle Cut   Bolognese 150g' 'Doritos Salsa Mild  300g']\n"
    ]
   }
  ],
  "source": [
   "# Check unique product names\n",
   "print(transaction_data['PROD_NAME'].unique())"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 18,
  "id": "e06041ca-a919-4494-ba2a-313617ab28c5",
  "metadata": {
   "scrolled": true
  },
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "['Natural Chip        Compny SeaSalt175g' 'CCs Nacho Cheese    175g'\n",
     " 'Smiths Crinkle Cut  Chips Chicken 170g'\n",
     " 'Smiths Chip Thinly  S/Cream&Onion 175g'\n",
     " 'Kettle Tortilla ChpsHny&Jlpno Chili 150g'\n",
     " 'Smiths Crinkle Chips Salt & Vinegar 330g'\n",
```

```
    " 'Grain Waves         Sweet Chilli 210g'\n",
    " 'Doritos Corn Chip Mexican Jalapeno 150g'\n",
    " 'Grain Waves Sour    Cream&Chives 210G'\n",
    " 'Kettle Sensations   Siracha Lime 150g' 'Twisties Cheese     270g'\n",
    " 'WW Crinkle Cut      Chicken 175g' 'Thins Chips Light&  Tangy 175g'\n",
    " 'CCs Original 175g' 'Burger Rings 220g'\n",
    " 'NCC Sour Cream &    Garden Chives 175g'\n",
    " 'Doritos Corn Chip Southern Chicken 150g' 'Cheezels Cheese Box 125g'\n",
    " 'Smiths Crinkle      Original 330g'\n",
    " 'Infzns Crn Crnchers Tangy Gcamole 110g'\n",
    " 'Kettle Sea Salt     And Vinegar 175g'\n",
    " 'Smiths Chip Thinly  Cut Original 175g' 'Kettle Original 175g'\n",
    " 'Red Rock Deli Thai  Chilli&Lime 150g' 'Pringles Sthrn FriedChicken 134g'\n",
    " 'Pringles Sweet&Spcy BBQ 134g' 'Thins Chips         Originl saltd 175g'\n",
    " 'Red Rock Deli Sp    Salt & Truffle 150G'\n",
    " 'Smiths Thinly       Swt Chli&S/Cream175G' 'Kettle Chilli 175g'\n",
    " 'Doritos Mexicana    170g' 'Smiths Crinkle Cut  French OnionDip 150g'\n",
    " 'Natural ChipCo      Hony Soy Chckn175g'\n",
    " 'Dorito Corn Chp     Supreme 380g' 'Twisties Chicken270g'\n",
    " 'Smiths Thinly Cut   Roast Chicken 175g'\n",
    " 'Kettle Mozzarella   Basil & Pesto 175g'\n",
    " 'Infuzions Thai SweetChili PotatoMix 110g'\n",
    " 'Kettle Sensations   Camembert & Fig 150g'\n",
    " 'Smith Crinkle Cut   Mac N Cheese 150g'\n",
    " 'Kettle Honey Soy    Chicken 175g' 'Thins Chips Seasonedchicken 175g'\n",
    " 'Smiths Crinkle Cut  Salt & Vinegar 170g'\n",
    " 'Infuzions BBQ Rib   Prawn Crackers 110g'\n",
    " 'GrnWves Plus Btroot & Chilli Jam 180g'\n",
    " 'Tyrrells Crisps     Lightly Salted 165g'\n",
    " 'Kettle Sweet Chilli And Sour Cream 175g' 'Kettle 135g Swt Pot Sea Salt'\n",
    " 'Pringles SourCream  Onion 134g' 'Doritos Corn Chips  Original 170g'\n",
    " 'Twisties Cheese     Burger 250g'\n",
    " 'Cobs Popd Swt/Chlli &Sr/Cream Chips 110g'\n",
    " 'Natural Chip Co     Tmato Hrb&Spce 175g'\n",
    " 'Smiths Crinkle Cut  Chips Original 170g'\n",
    " 'Cobs Popd Sea Salt  Chips 110g'\n",
    " 'Smiths Crinkle Cut  Chips Chs&Onion170g'\n",
    " 'French Fries Potato Chips 175g'\n",
    " 'Doritos Corn Chips  Cheese Supreme 170g'\n",
    " 'Pringles Original   Crisps 134g' 'RRD Chilli&         Coconut 150g'\n",
    " 'WW Original Corn    Chips 200g' 'Thins Potato Chips  Hot & Spicy 175g'\n",
    " 'Cobs Popd Sour Crm  &Chives Chips 110g'\n",
    " 'Smiths Crnkle Chip  Orgnl Big Bag 380g'\n",
    " 'Doritos Corn Chips  Nacho Cheese 170g'\n",
    " 'Kettle Sensations   BBQ&Maple 150g' 'WW D/Style Chip     Sea Salt 200g'\n",
    " 'Pringles Chicken    Salt Crips 134g' 'WW Original Stacked Chips 160g'\n",
    " 'Smiths Chip Thinly  CutSalt/Vinegr175g' 'Cheezels Cheese 330g'\n",
    " 'Tostitos Lightly    Salted 175g' 'Thins Chips Salt &  Vinegar 175g'\n",
    " 'Smiths Crinkle Cut  Chips Barbecue 170g' 'Cheetos Puffs 165g'\n",
    " 'RRD Sweet Chilli &  Sour Cream 165g' 'WW Crinkle Cut      Original 175g'\n",
    " 'Tostitos Splash Of  Lime 175g' 'Kettle Tortilla ChpsBtroot&Ricotta 150g'\n",
    " 'CCs Tasty Cheese    175g' 'Woolworths Cheese   Rings 190g'\n",
    " 'Tostitos Smoked     Chipotle 175g' 'Pringles Barbeque   134g'\n",
    " 'WW Supreme Cheese   Corn Chips 200g' 'Pringles Mystery    Flavour 134g'\n",
    " 'Tyrrells Crisps     Ched & Chives 165g'\n",
    " 'Snbts Whlgrn Crisps Cheddr&Mstrd 90g' 'Cheetos Chs & Bacon Balls 190g'\n",
    " 'Pringles Slt Vingar 134g' 'Infuzions SourCream&Herbs Veg Strws 110g'\n",
    " 'Kettle Tortilla ChpsFeta&Garlic 150g'\n",
    " 'Infuzions Mango     Chutny Papadums 70g'\n",
    " 'RRD Steak &         Chimuchurri 150g' 'RRD Honey Soy       Chicken 165g'\n",
    " 'Sunbites Whlegrn    Crisps Frch/Onin 90g' 'RRD Salt & Vinegar  165g'\n",
    " 'Doritos Cheese      Supreme 330g' 'Smiths Crinkle Cut  Snag&Sauce 150g'\n",
    " 'WW Sour Cream &OnionStacked Chips 160g' 'RRD Lime & Pepper   165g'\n",
    " 'Natural ChipCo Sea  Salt & Vinegr 175g'\n",
    " 'Red Rock Deli Chikn&Garlic Aioli 150g'\n",
    " 'RRD SR Slow Rst     Pork Belly 150g' 'RRD Pc Sea Salt     165g'\n",
    " 'Smith Crinkle Cut   Bolognese 150g']\n"
   ]
  }
 ],
 "source": [
  "# Remove salsa products\n",
  "transaction_data = transaction_data[~transaction_data['PROD_NAME'].str.contains('salsa', case=False)]\n",
  "\n",
  "# Verify the removal\n",
  "print(transaction_data['PROD_NAME'].unique())"
 ]
},
{
 "cell_type": "markdown",
 "id": "fbdbc5ad-8381-4a41-bbba-07d9ead08ca6",
 "metadata": {},
 "source": [
  "## Sales Trends Over Time"
 ]
},
{
 "cell_type": "code",
 "execution_count": 19,
 "id": "cfd50801-abff-4b8b-81eb-1b971d9013f3",
 "metadata": {},
 "outputs": [
  {
   "name": "stderr",
   "output_type": "stream",
   "text": [
    "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\seaborn\\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in\n",
    "  with pd.option_context('mode.use_inf_as_na', True):\n",
    "C:\\Users\\Lekhansh\\anaconda3\\Lib\\site-packages\\seaborn\\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in\n",
    "  with pd.option_context('mode.use_inf_as_na', True):\n"
   ]
  },
  {
   "data": {
    "image/png":
    "text/plain": [
```

```
      "<Figure size 1000x600 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   }
  ],
  "source": [
   "# Group by date to count transactions\n",
   "transactions_by_day = transaction_data.groupby('DATE').size().reset_index(name='transaction_count')\n",
   "\n",
   "# Plot transactions over time\n",
   "plt.figure(figsize=(10, 6))\n",
   "sns.lineplot(x='DATE', y='transaction_count', data=transactions_by_day)\n",
   "plt.title('Transactions Over Time')\n",
   "plt.xlabel('Date')\n",
   "plt.ylabel('Number of Transactions')\n",
   "plt.xticks(rotation=45)\n",
   "plt.show()"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "e41ee22c-91ab-49f3-8b85-5b277510cb35",
  "metadata": {},
  "source": [
   "## New Features\n",
   "\n",
   "Creating PACK_SIZE and BRAND Features"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 20,
  "id": "51d3da40-9316-47b7-adfb-c4ac00b6738e",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "   PACK_SIZE    BRAND\n",
     "0      175.0  Natural\n",
     "1      175.0      CCs\n",
     "2      170.0   Smiths\n",
     "3      175.0   Smiths\n",
     "4      150.0   Kettle\n"
    ]
   }
  ],
  "source": [
   "# Extract pack size by finding digits in 'PROD_NAME'\n",
   "transaction_data['PACK_SIZE'] = transaction_data['PROD_NAME'].str.extract('(\\d+)').astype(float)\n",
   "\n",
   "# Extract brand name as the first word of 'PROD_NAME'\n",
   "transaction_data['BRAND'] = transaction_data['PROD_NAME'].str.split().str[0]\n",
   "\n",
   "# Check the new columns\n",
   "print(transaction_data[['PACK_SIZE', 'BRAND']].head())"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 24,
  "id": "e1b3c0d2-8a2b-40d3-a8f2-71838c4e14ea",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "['Natural' 'CCs' 'Smiths' 'Kettle' 'Grain' 'Doritos' 'Twisties' 'WW'\n",
     " 'Thins' 'Burger' 'NCC' 'Cheezels' 'Infzns' 'Red' 'Pringles' 'Dorito'\n",
     " 'Infuzions' 'Smith' 'GrnWves' 'Tyrrells' 'Cobs' 'French' 'RRD' 'Tostitos'\n",
     " 'Cheetos' 'Woolworths' 'Snbts' 'Sunbites']\n"
    ]
   }
  ],
  "source": [
   "print(transaction_data['BRAND'].unique())"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "ef0a7219-07fd-4b9c-ad9a-2a01840d4c03",
  "metadata": {},
  "source": [
   "Some brands may need to be combined for consistency (e.g., \"RED\" and \"RRD\")."
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 26,
  "id": "d73ca9fe-1228-4520-b174-390ad07dc877",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "['Natural' 'CCs' 'Smiths' 'Kettle' 'Grain' 'Doritos' 'Twisties' 'WW'\n",
     " 'Thins' 'Burger' 'NCC' 'Cheezels' 'Infzns' 'RRD' 'Pringles' 'Dorito'\n",
     " 'Infuzions' 'Smith' 'GrnWves' 'Tyrrells' 'Cobs' 'French' 'Tostitos'\n",
     " 'Cheetos' 'Woolworths' 'Snbts' 'Sunbites']\n"
    ]
   }
```

```
 ],
 "source": [
  "# Clean brand names by combining similar names\n",
  "transaction_data['BRAND'] = transaction_data['BRAND'].replace({'Red': 'RRD'})\n",
  "\n",
  "# Check the updated brand names\n",
  "print(transaction_data['BRAND'].unique())"
 ]
},
{
 "cell_type": "markdown",
 "id": "bd299ef8-8c9e-49e0-89d1-aff463c6f4a5",
 "metadata": {},
 "source": [
  "## Merging Data\n",
  "\n",
  "Now, merging the transaction data with the customer data to analyze customer segments."
 ]
},
{
 "cell_type": "code",
 "execution_count": 29,
 "id": "40316141-c37e-4790-baef-43556ac98a7a",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "DATE                0\n",
    "STORE_NBR           0\n",
    "LYLTY_CARD_NBR      0\n",
    "TXN_ID              0\n",
    "PROD_NBR            0\n",
    "PROD_NAME           0\n",
    "PROD_QTY            0\n",
    "TOT_SALES           0\n",
    "PACK_SIZE           0\n",
    "BRAND               0\n",
    "LIFESTAGE           0\n",
    "PREMIUM_CUSTOMER    0\n",
    "dtype: int64\n"
   ]
  }
 ],
 "source": [
  "# Merge transaction data with customer data on loyalty card number\n",
  "merged_data = pd.merge(transaction_data, purchase_behaviour, on='LYLTY_CARD_NBR', how='left')\n",
  "\n",
  "# Check for missing customer details\n",
  "print(merged_data.isnull().sum())"
 ]
},
{
 "cell_type": "markdown",
 "id": "9465ff09-cfdf-412f-a002-3302001605bd",
 "metadata": {},
 "source": [
  "## Analyzing Sales by Customer Segments\n",
  "\n",
  "Let’s calculate total sales by LIFESTAGE and PREMIUM_CUSTOMER."
 ]
},
{
 "cell_type": "code",
 "execution_count": 30,
 "id": "ea7128be-042a-43df-8d23-9a9e2c9deba4",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "image/png":
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  }
 ],
 "source": [
  "# Calculate total sales by LIFESTAGE and PREMIUM_CUSTOMER\n",
  "sales_by_segment = merged_data.groupby(['LIFESTAGE', 'PREMIUM_CUSTOMER'])['TOT_SALES'].sum().reset_index()\n",
  "\n",
  "# Plot total sales by segment\n",
  "plt.figure(figsize=(10, 6))\n",
  "sns.barplot(x='LIFESTAGE', y='TOT_SALES', hue='PREMIUM_CUSTOMER', data=sales_by_segment)\n",
  "plt.title('Total Sales by Lifestage and Premium Customer Status')\n",
  "plt.xticks(rotation=45)\n",
  "plt.show()"
 ]
},
{
 "cell_type": "markdown",
 "id": "efcb4b1c-2ab1-49a1-b110-2121323ed509",
 "metadata": {},
 "source": [
  "## Number of Customers by Segment\n",
  "\n",
  "Next, we can analyze the number of customers in each segment."
 ]
},
{
 "cell_type": "code",
 "execution_count": 32,
 "id": "2d26a858-3046-4b2a-a389-3416c5794f61",
```

```
 "metadata": {},
 "outputs": [
  {
   "data": {
    "image/png":
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  }
 ],
 "source": [
  "# Count customers by LIFESTAGE and PREMIUM_CUSTOMER\n",
  "customers_by_segment = merged_data.groupby(['LIFESTAGE', 'PREMIUM_CUSTOMER'])['LYLTY_CARD_NBR'].nunique().reset_index(name='customer_count')\n",
  "\n",
  "# Plot customer count by segment\n",
  "plt.figure(figsize=(10, 6))\n",
  "sns.barplot(x='LIFESTAGE', y='customer_count', hue='PREMIUM_CUSTOMER', data=customers_by_segment)\n",
  "plt.title('Customer Count by Lifestage and Premium Customer Status')\n",
  "plt.xticks(rotation=45)\n",
  "plt.show()"
 ]
},
{
 "cell_type": "markdown",
 "id": "701ec5df-6c12-4453-bba7-b0bdb3675885",
 "metadata": {},
 "source": [
  "## Average Number of Units Purchased\n",
  "\n",
  "Let’s calculate the average number of chip units purchased per customer."
 ]
},
{
 "cell_type": "code",
 "execution_count": 34,
 "id": "4a9dcd9a-5ca1-4034-a9a8-37ec1f6f9365",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "image/png":
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  }
 ],
 "source": [
  "# Calculate average units purchased by LIFESTAGE and PREMIUM_CUSTOMER\n",
  "avg_units_by_segment = merged_data.groupby(['LIFESTAGE', 'PREMIUM_CUSTOMER'])['PROD_QTY'].mean().reset_index()\n",
  "\n",
  "# Plot average units purchased by segment\n",
  "plt.figure(figsize=(10, 6))\n",
  "sns.barplot(x='LIFESTAGE', y='PROD_QTY', hue='PREMIUM_CUSTOMER', data=avg_units_by_segment)\n",
  "plt.title('Average Units Purchased by Lifestage and Premium Customer Status')\n",
  "plt.xticks(rotation=45)\n",
  "plt.show()"
 ]
},
{
 "cell_type": "markdown",
 "id": "2c9e452d-4f26-480c-986e-310aa09d5055",
 "metadata": {},
 "source": [
  "## Average Price per Unit\n",
  "\n",
  "Finally, let's calculate the average price per unit for each segment."
 ]
},
{
 "cell_type": "code",
 "execution_count": 35,
 "id": "4b3e7fb2-4b7c-43d0-a3ce-96337d89dc41",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "image/png":
    "text/plain": [
     "<Figure size 1000x600 with 1 Axes>"
    ]
   },
   "metadata": {},
   "output_type": "display_data"
  }
 ],
 "source": [
  "# Calculate average price per unit by LIFESTAGE and PREMIUM_CUSTOMER\n",
  "avg_price_by_segment = merged_data.groupby(['LIFESTAGE', 'PREMIUM_CUSTOMER'])['TOT_SALES'].mean().reset_index()\n",
  "\n",
  "# Plot average price per unit by segment\n",
  "plt.figure(figsize=(10, 6))\n",
  "sns.barplot(x='LIFESTAGE', y='TOT_SALES', hue='PREMIUM_CUSTOMER', data=avg_price_by_segment)\n",
  "plt.title('Average Price per Unit by Lifestage and Premium Customer Status')\n",
  "plt.xticks(rotation=45)\n",
  "plt.show()"
 ]
},
{
 "cell_type": "markdown",
 "id": "a55e0d75-4f70-4063-9d7f-b3018c159baf",
```

```
   "metadata": {},
   "source": [
    "## Saving Cleaned Data"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 36,
   "id": "471b1b0f-9b31-4dc8-b031-afe720b6e2d2",
   "metadata": {},
   "outputs": [],
   "source": [
    "# Save the cleaned merged data to a CSV\n",
    "merged_data.to_csv('QVI_cleaned_data.csv', index=False)"
   ]
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.11.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```