Water Main Break Analysis Report

1. Introduction

This project analyzes a dataset about water main breaks. The goal is to follow the data science process, which includes **loading, cleaning, transforming, visualizing the data, and extracting insights.** We will use Power BI for visualization and Python in a Jupyter Notebook for this work. The analysis aims to find patterns and trends that can help improve water supply management.

2. Data Science Workflow

2.1. Dataset Source and Description

The dataset utilized for this project was obtained from the **City of Kitchener's Open Data Portal**. The "Water Main Breaks" dataset offers real-time records of pipe break incidents throughout the city. It includes detailed information such as the type of break, location, material, repair method, date of occurrence, and more.

Dataset Link:
Water Main Breaks – Kitchener GeoHub

2.2 Data Exploration

The dataset was first examined in its raw form using **Python in Jupyter Notebook**. This phase aimed to understand the structure, content, and quality of the data.

```python
#Basic Info
print("Shape:", df.shape)

print("\nInfo:\n")
print(df.info())
```

```
Shape: (2839, 18)

Info:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2839 entries, 0 to 2838
Data columns (total 18 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   OBJECTID                                      2839 non-null   int64
 1   Wat Break Incident ID                         2839 non-null   int64
 2   incident-date                                 2839 non-null   int64
 3   Type of Asset Broken                          2839 non-null   object
 4   Current status of the break                   2839 non-null   object
 5   UPDATE_DATE                                   2447 non-null   datetime64[ns]
 6   Date operations was returned to normal service  106 non-null   datetime64[ns]
 7   Nature of Break                               2636 non-null   object
 8   Apparent cause of break                       2558 non-null   object
 9   Repair Type                                   191 non-null    object
 10  Street                                        2824 non-null   object
```

```python
# Value Counts (Categorical Columns)
categorical_cols = df.select_dtypes(include=["object"]).columns

for col in categorical_cols:
    print(f"\nColumn: {col}")
    print(df[col].value_counts())
    print("\nPercentage:")
    print(df[col].value_counts(normalize=True) * 100)
    print("-" * 50)
```

```
Column: incident-date
incident-date
15/12/2011    59
12/01/2008    21
23/04/2015     9
06/09/2016     7
03/05/2015     6
              ..
26/10/2006     1
22/10/2006     1
17/10/2006     1
18/09/2006     1
08/12/2024     1
Name: count, Length: 2103, dtype: int64

Percentage:
incident-date
15/12/2011    2.078197
12/01/2008    0.739697
23/04/2015    0.317013
```

2.3 Data Cleaning and Preparation

2.3.1. Data Loading

The dataset was imported using the `pandas` library:

df = pd.read_excel("water-main-breaks.xlsx")

A preview was displayed to verify successful loading.

```
print(df.head())
```

```
   OBJECTID  Wat Break Incident ID  incident-date Type of Asset Broken  \
0      8419                   1677          31048                 MAIN
1      9577                   1667          31048                 MAIN
2     10350                   1668          31048                 MAIN
3      8788                   1669          31413                 MAIN
4      9260                   1688          31778                 MAIN

  Current status of the break UPDATE_DATE  \
0            REPAIR COMPLETED  2011-12-15
1            REPAIR COMPLETED  2011-12-15
2            REPAIR COMPLETED  2019-07-18
3            REPAIR COMPLETED  2011-12-15
4            REPAIR COMPLETED  2011-12-15

  Date operations was returned to normal service Nature of Break  \
0                                            NaT         UNKNOWN
1                                            NaT         UNKNOWN
2                                            NaT         UNKNOWN
3                                            NaT         UNKNOWN
4                                            NaT         UNKNOWN

  Apparent cause of break Repair Type        Street  Related Asset ID  \
0                   OTHER         NaN    BRANT CRES            136130
1                   OTHER         NaN     WALTER ST             97716
2                   OTHER         NaN   VANCAMP AVE            135674
3                   OTHER         NaN  SPADINA RD W             35450
```

2.3.2. Date Format Standardization

The dataset had inconsistent and non-standard date formats, especially in the **`incident-date`** column. To resolve this:

- Non-numeric characters were removed.
- Excel serial date numbers were converted into proper datetime format.
- Dates were then formatted as **DD/MM/YYYY**.

The **UPDATE_DATE** column was also standardized using:

df["UPDATE_DATE"] = pd.to_datetime(df["UPDATE_DATE"], format="%d/%m/%Y", errors="coerce")

```
# Step 1: Data Cleaning
# Extract numbers
df['incident-date'] = df['incident-date'].astype(str).str.extract('(\d+)')
# Convert to numbers
df['incident-date'] = pd.to_numeric(df['incident-date'], errors='coerce')
df['incident-date'] = pd.to_datetime(df['incident-date'], origin='1899-12-30', unit='D', errors='coerce')
df['incident-date'] = df['incident-date'].dt.strftime("%d/%m/%Y")  # Format as dd/mm/yyyy

print(df[['incident-date']].head())
```

```
  incident-date
0    01/01/1985
1    01/01/1985
2    01/01/1985
3    01/01/1986
4    01/01/1987
```

2.3.3. Missing and Duplicate Data Analysis

To understand data quality, missing and duplicate values were identified:

missing_values = df.isnull().sum()
duplicate_count = df.duplicated().sum()

```
print("Missing Values:\n", missing_values)
print("\nNumber of Duplicate Rows:", duplicate_count)
```

```
Missing Values:
 OBJECTID                                            0
Wat Break Incident ID                                0
incident-date                                        0
Type of Asset Broken                                 0
Current status of the break                          0
UPDATE_DATE                                        392
Date operations was returned to normal service    2733
Nature of Break                                    203
Apparent cause of break                            281
Repair Type                                       2648
Street                                              15
Related Asset ID                                     0
Related Asset Depth (m)                           2824
Depth of Frost (m)                                2758
Asset Size (cm)                                    278
Year Asset Installed                               282
Asset Material                                     278
Asset Exists                                         0
dtype: int64

Number of Duplicate Rows: 0
```

The percentage of missing values per column was calculated to guide the cleaning process:

```
#missing values percentage
missing_values = df.isnull().sum()
missing_percentage = (missing_values / len(df)) * 100
print(missing_percentage)
```

```
OBJECTID                                           0.000000
Wat Break Incident ID                              0.000000
incident-date                                      0.000000
Type of Asset Broken                               0.000000
Current status of the break                        0.000000
UPDATE_DATE                                       13.807679
Date operations was returned to normal service    96.266291
Nature of Break                                    7.150405
Apparent cause of break                            9.897851
Repair Type                                       93.272279
Street                                             0.528355
Related Asset ID                                   0.000000
Related Asset Depth (m)                           99.471645
Depth of Frost (m)                                97.146883
Asset Size (cm)                                    9.792180
Year Asset Installed                               9.933075
Asset Material                                     9.792180
Asset Exists                                       0.000000
dtype: float64
```

2.3.4. Data Cleaning and Imputation

Columns with more than **90% missing data** were considered for removal. Essential columns were retained and cleaned using imputation:

- **Datetime Columns:** Missing values were forward-filled.
- **Categorical Columns:** Filled using the mode (most frequent value).
- **Numerical Columns:** Filled using the median.
- **Custom Handling:** Some values were replaced with "Unknown" where appropriate:

A final check was performed to confirm all missing values had been handled.

```
file_path = r"C:\Users\DELL\water-main-breaks.xlsx"

print("Missing Values Before Imputation:")
print(df.isnull().sum())

#  Forward fill for datetime columns
df['UPDATE_DATE'] = df['UPDATE_DATE'].ffill()

#  Categorical columns - fill with mode (most frequent value)
categorical_cols = ['Nature of Break', 'Apparent cause of break', 'Repair Type', 'Street', 'Asset Material']
for col in categorical_cols:
    df[col] = df[col].fillna(df[col].mode()[0])

# Numerical columns - fill with median
numerical_cols = ['Asset Size (cm)', 'Year Asset Installed']
for col in numerical_cols:
    df[col] = df[col].fillna(df[col].median())

# fill empty rows with Unknown
df["Repair Type"].fillna("Unknown", inplace=True)

# Display missing values after imputation
print("\nMissing Values After Imputation:")
print(df.isnull().sum())
# Display the DataFrame
print(df.head())
```

```
Missing Values After Imputation:
OBJECTID                                          0
Wat Break Incident ID                             0
incident-date                                     0
Type of Asset Broken                              0
Current status of the break                       0
UPDATE_DATE                                       0
Date operations was returned to normal service    2733
Nature of Break                                   0
Apparent cause of break                           0
Repair Type                                       0
Street                                            0
Related Asset ID                                  0
Related Asset Depth (m)                           2824
Depth of Frost (m)                                2758
Asset Size (cm)                                   0
Year Asset Installed                              0
Asset Material                                    0
Asset Exists                                      0
dtype: int64
```

```python
# Remove spaces in column names
df.columns = df.columns.str.strip()

# Drop columns with excessive missing values (>90%)
columns_to_drop = ['Date operations was returned to normal service','Related Asset Depth (m)', 'Depth of Frost (m)']
df = df.drop(columns=[col for col in columns_to_drop if col in df.columns], axis=1)

print("Columns after dropping:", df.columns)

#save the dataset
df.to_csv(r"C:\Users\DELL\cleaned_water_main_breaks.csv", index=False)
print("\nCleaned dataset saved as 'cleaned_water_main_breaks.csv'.")
```

```
Columns after dropping: Index(['OBJECTID', 'Wat Break Incident ID', 'incident-date',
       'Type of Asset Broken', 'Current status of the break', 'UPDATE_DATE',
       'Nature of Break', 'Apparent cause of break', 'Repair Type', 'Street',
       'Related Asset ID', 'Asset Size (cm)', 'Year Asset Installed',
       'Asset Material', 'Asset Exists'],
      dtype='object')

Cleaned dataset saved as 'cleaned_water_main_breaks.csv'.
```

2.3.5. Descriptive Analysis and Date Dimension Columns

To enable time-based analysis, date dimension features were created:

- Extracted day, month, year from `incident-date` and `UPDATE_DATE`.

- This helped support deeper insights during visualization and trend analysis.

```
# Convert date columns to datetime format
df_main['incident-date'] = pd.to_datetime(df_main['incident-date'], errors='coerce', dayfirst=True)
df_main['UPDATE_DATE'] = pd.to_datetime(df_main['UPDATE_DATE'], errors='coerce')

# Function to create date dimension attributes
def create_date_dimension(df, date_column):
    df[f'{date_column}_Year'] = df[date_column].dt.year
    df[f'{date_column}_Month'] = df[date_column].dt.month
    df[f'{date_column}_Month_Name'] = df[date_column].dt.strftime('%B')
    df[f'{date_column}_Day'] = df[date_column].dt.day
    df[f'{date_column}_Day_of_Week'] = df[date_column].dt.dayofweek
    df[f'{date_column}_Day_Name'] = df[date_column].dt.strftime('%A')
    df[f'{date_column}_Quarter'] = df[date_column].dt.quarter
    df[f'{date_column}_Week_of_Year'] = df[date_column].dt.isocalendar().week
    return df

# Apply the function to both date columns
df_main = create_date_dimension(df_main, 'incident-date')
df_main = create_date_dimension(df_main, 'UPDATE_DATE')

# Define output file path and save the updated DataFrame
output_file_path = (r"C:\Users\DELL\date-dimension.csv")
df_main.to_csv(output_file_path, index=False)
# Return the output file path
output_file_path
```

```
'C:\\Users\\DELL\\date-dimension.csv'
```

Descriptive analysis was performed to summarize and understand our data.

This overview shows the basic structure of the dataset. We can see it contains **2,839 entries** and 15 columns of information about each one. The data types are a mix of numbers (integers and decimals) and text (objects). A key positive insight is that all columns show 2,839 non-null values, meaning there are no missing entries in this dataset, which is excellent for ensuring a complete analysis. The columns contain a variety of information, from dates and causes to asset materials and repair types, providing a comprehensive view for the investigation.

```
General Info about the dataset:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2839 entries, 0 to 2838
Data columns (total 15 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   OBJECTID                     2839 non-null   int64
 1   Wat Break Incident ID        2839 non-null   int64
 2   incident-date                2839 non-null   object
 3   Type of Asset Broken         2839 non-null   object
 4   Current status of the break  2839 non-null   object
 5   UPDATE_DATE                  2839 non-null   object
 6   Nature of Break              2839 non-null   object
 7   Apparent cause of break      2839 non-null   object
 8   Repair Type                  2839 non-null   object
 9   Street                       2839 non-null   object
 10  Related Asset ID             2839 non-null   int64
 11  Asset Size (cm)              2839 non-null   float64
 12  Year Asset Installed         2839 non-null   float64
 13  Asset Material               2839 non-null   object
 14  Asset Exists                 2839 non-null   object
dtypes: float64(2), int64(3), object(10)
memory usage: 332.8+ KB
None
```

Missing values were handled by dropting columns with over 90% missing values and performing imputation to fill out missing values in columns essential for analysis.

```
 Missing Values:
 OBJECTID                     0
Wat Break Incident ID         0
incident-date                 0
Type of Asset Broken          0
Current status of the break   0
UPDATE_DATE                   0
Nature of Break               0
Apparent cause of break       0
Repair Type                   0
Street                        0
Related Asset ID              0
Asset Size (cm)               0
Year Asset Installed          0
Asset Material                0
Asset Exists                  0
dtype: int64
```

Statistical Summary of Numerical columns

This summary highlights key statistics for the numerical data. The most common "Asset Size" is **150 cm,** based on the median. The maximum size is **1200 cm,** suggesting some unusually large values. Most assets are old, with an average year of installation is **1968** and a **median of 1965.**

Summary Statistics for Numerical Columns:

	OBJECTID	Wat Break Incident ID	Related Asset ID	Asset Size (cm)
count	2839.000000	2839.000000	2.839000e+03	2839.000000
mean	15140.839028	23044.647411	1.189747e+05	178.797112
std	16788.102088	53317.794691	5.589350e+05	83.895756
min	1.000000	1.000000	0.000000e+00	0.000000
25%	8697.500000	712.500000	1.756000e+04	150.000000
50%	9409.000000	1428.000000	3.354000e+04	150.000000
75%	10133.500000	2161.500000	7.616400e+04	150.000000
max	88001.000000	197291.000000	4.769517e+06	1200.000000

	Year Asset Installed
count	2839.000000
mean	1968.829517
std	20.103498
min	1889.000000
25%	1958.000000
50%	1965.000000
75%	1971.000000
max	2018.000000

Statistical Summary of Categorical Columns

This table provides key statistics for the 'Year Asset Installed' column. A major insight is that the infrastructure is, on average, very old. The mean installation year is **approximately 1969,** and the **median is 1965.** This indicates that half of all the assets in the dataset were installed **before 1965.** The fact that a significant portion of the network is **over 50 years old** is a crucial finding, as older assets are generally more prone to failures and breaks.

Summary Statistics for Categorical Columns:

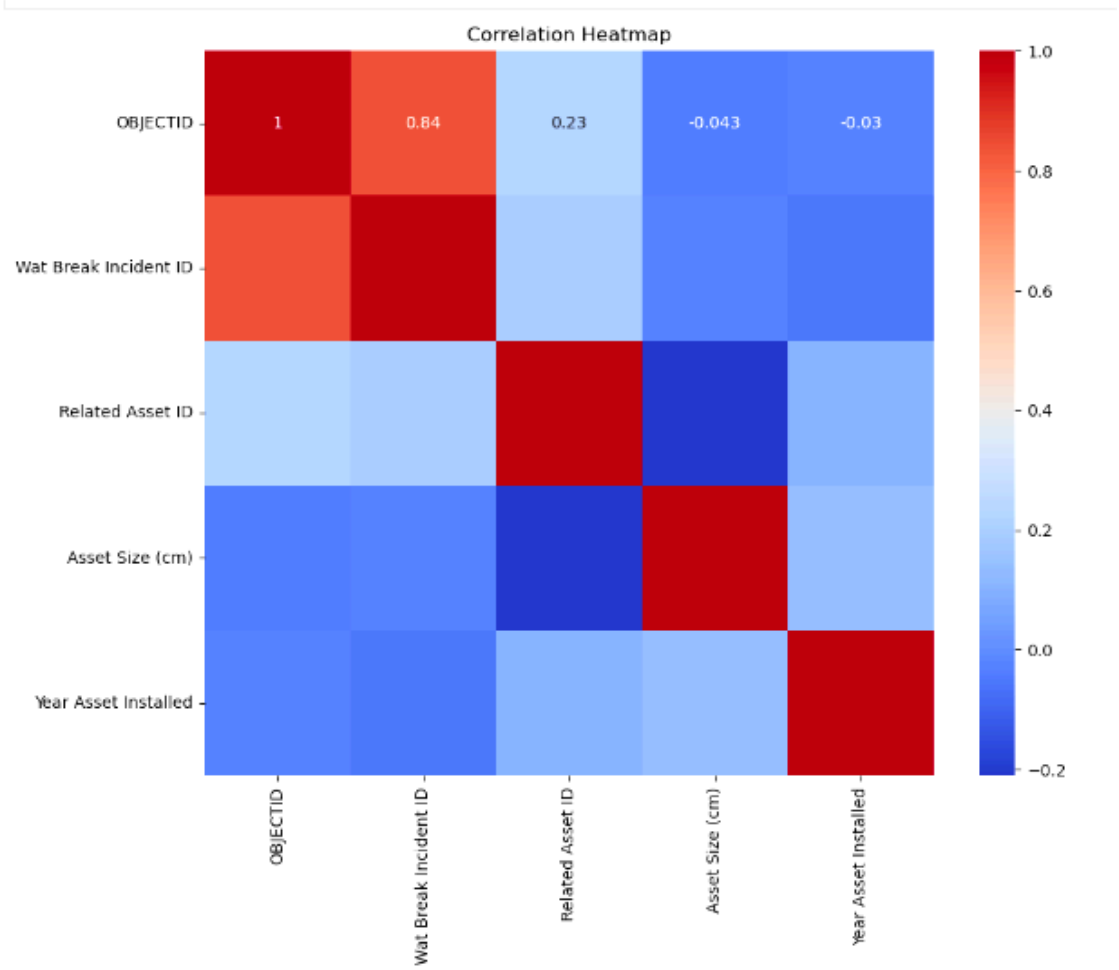
	OBJECTID	Wat Break Incident ID	Related Asset ID	Asset Size (cm)
count	2839.000000	2839.000000	2.839000e+03	2839.000000
mean	15140.839028	23044.647411	1.189747e+05	178.797112
std	16788.102088	53317.794691	5.589350e+05	83.895756
min	1.000000	1.000000	0.000000e+00	0.000000
25%	8697.500000	712.500000	1.756000e+04	150.000000
50%	9409.000000	1428.000000	3.354000e+04	150.000000
75%	10133.500000	2161.500000	7.616400e+04	150.000000
max	88001.000000	197291.000000	4.769517e+06	1200.000000

	Year Asset Installed
count	2839.000000
mean	1968.829517
std	20.103498
min	1889.000000
25%	1958.000000
50%	1965.000000
75%	1971.000000
max	2018.000000

3. Correlation Heatmap

Python Code: Correlation heatmap for numeric columns

```
# Correlation Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation, annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```



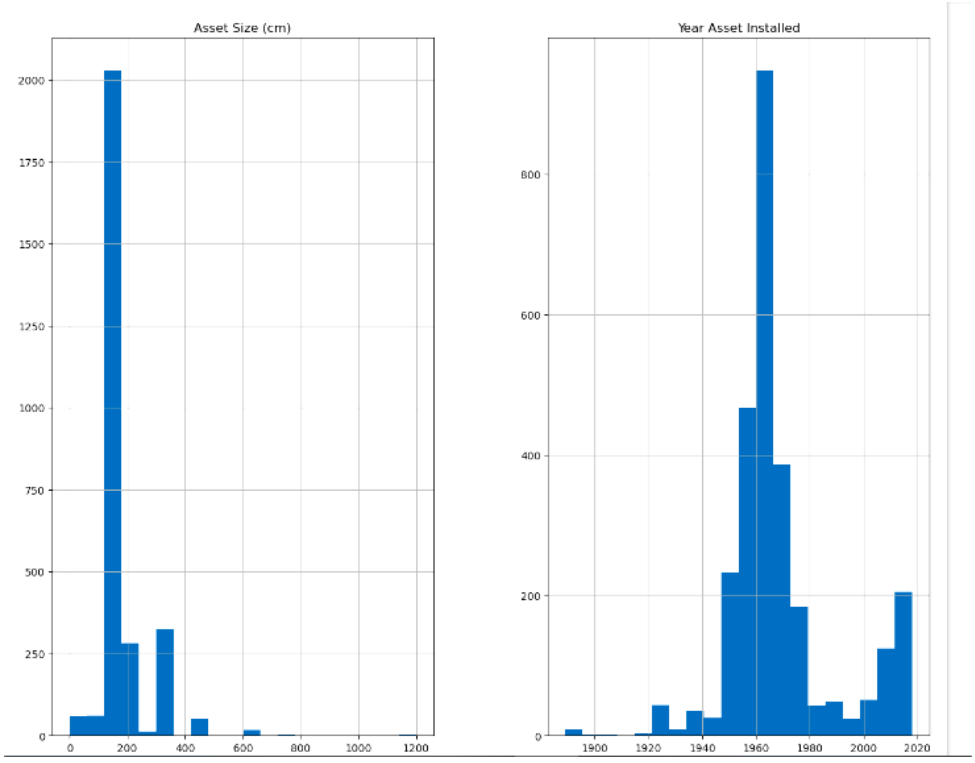
The water breaks heatmap highlights the frequency and distribution of breaks across different times and conditions, making it easy to spot patterns. Darker areas show where breaks happen most often, while lighter areas indicate fewer occurrences. This helps identify peak times or situations when water breaks are more common, pointing to possible underlying causes such as workload, scheduling, or environmental factors that may need attention.

4. Histograms

Python code:

```
# Histograms
numeric_cols = ["Asset Size (cm)", "Year Asset Installed"]

df[numeric_cols].hist(figsize=(15, 12), bins=20)
plt.suptitle("Histograms of Selected Columns")
plt.show()
```



The histograms show that most assets are relatively small, with a high concentration below 200 cm and very few exceeding that range. This suggests that the dataset is mainly made up of smaller assets. Regarding installation year, most assets were installed between the 1940s and 1970s, with a notable peak in the 1960s. Installations decreased afterward but experienced a smaller resurgence in the 2000s and 2010s. These patterns imply that the asset base is mostly old and may need maintenance or replacement, while the recent additions point to some modernization in the system.

5. Skewness and Kurtosis

```
# Skewness & Kurtosis
for col in numeric_cols:
    print(f"{col}: Skewness={df[col].skew():.2f}, Kurtosis={df[col].kurt():.2f}")
```

```
Asset Size (cm): Skewness=4.26, Kurtosis=35.92
Year Asset Installed: Skewness=0.79, Kurtosis=1.54
```

- The **Asset Size (cm)** distribution, with a skewness of 4.26 and kurtosis of 35.92, is highly positively skewed and extremely peaked, indicating that most assets are clustered at smaller sizes with a few very large outliers.
- In contrast, the **Year Asset Installed** distribution, with a skewness of 0.79 and kurtosis of 1.54, is moderately right-skewed and slightly more peaked than a normal distribution, suggesting that most assets were installed around a central time period, with some leaning toward more recent years.

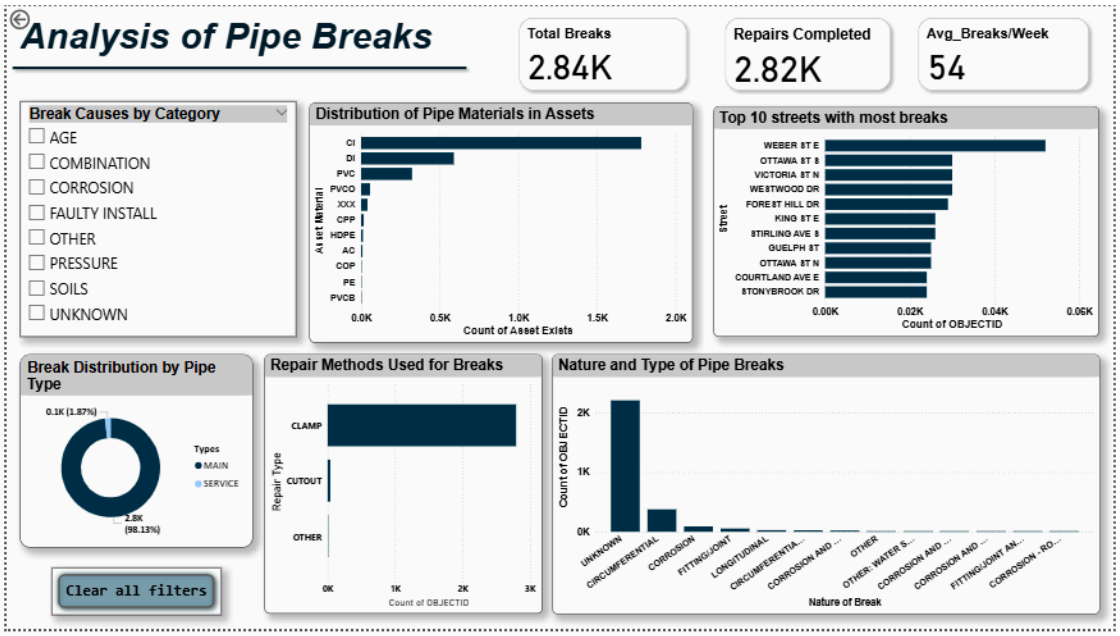
---

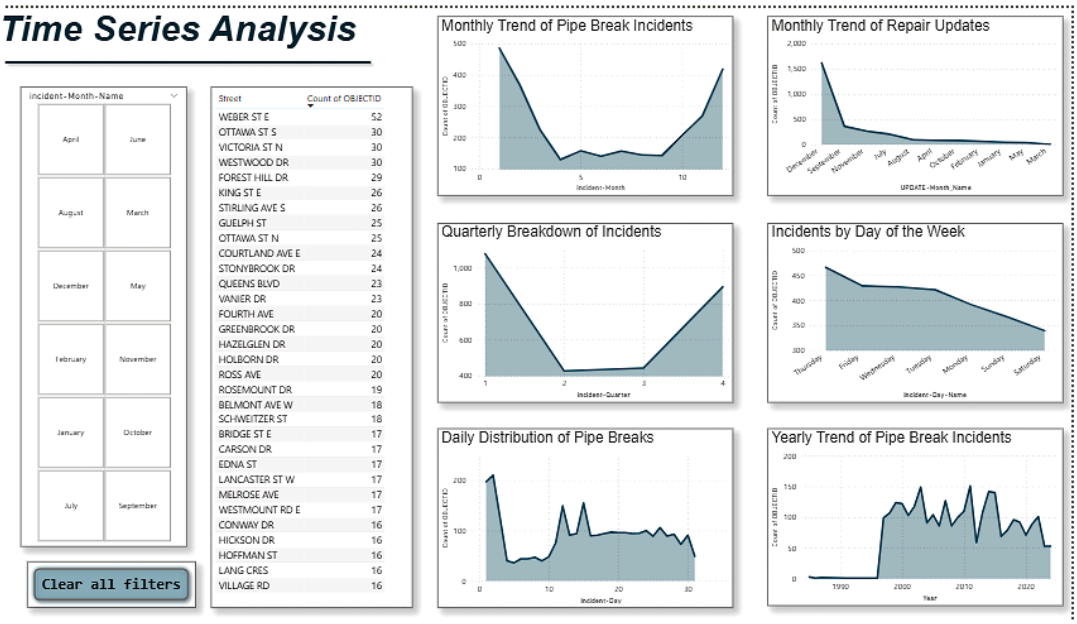
6. Data Analysis and Visualization

After cleaning, the dataset was imported into **Power BI** for interactive visualization and deeper insight extraction. Key visuals were created to represent the data across time, location, material, and other relevant dimensions.

6.1 Power BI Dashboard Overview

This Power BI dashboard provides an in-depth analysis of **485 water main break incidents**, focusing on location, frequency, causes, materials, and time-based patterns. It includes breakdowns by **pipe material**, **type of break**, **repair method**, **geography**, and **temporal trends** (year, month, day).





7. Visual-wise Insight Breakdown

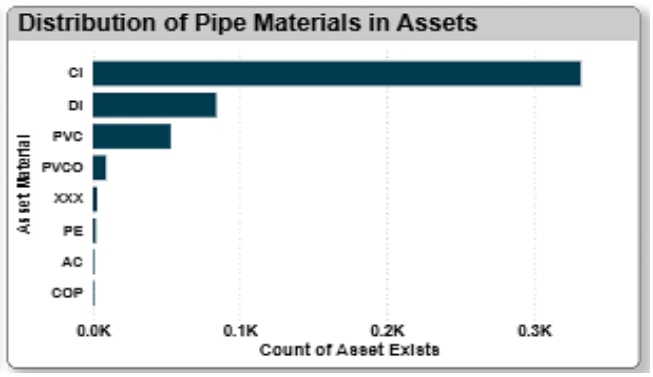
1. Key Metrics (KPI Cards)

- **Total Breaks:** 2.84k
- **Repairs Completed:** 2.82k

- **Average Breaks per Week:** 54

Insight: The KPIs provided give key insights into the performance and frequency of pipe break incidents and how they are resolved. A total of **2.84K breaks** have been reported, with **2.82K successfully repaired**, showing a high repair success rate and effective maintenance response. Additionally, an average of **54 breaks per week** emphasizes the ongoing issue and highlights the need for continuous monitoring and proactive maintenance strategies to decrease break frequency over time.

2. Distribution of Pipe Materials



A bar chart displaying the count of assets by material type shows that the most commonly used pipe materials are:

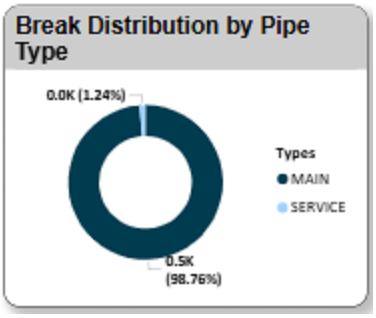
- **Cast Iron (CI)**
- **Ductile Iron (DI)**
- **PVC and PVCO**

Key Insight:

The use of older materials like Cast Iron may correlate with higher break rates. Gradually phase out older materials like CI in favor of modern, durable alternatives like PVC or PE.

---

3. Break Distribution by Pipe Type



A pie chart reveals that:

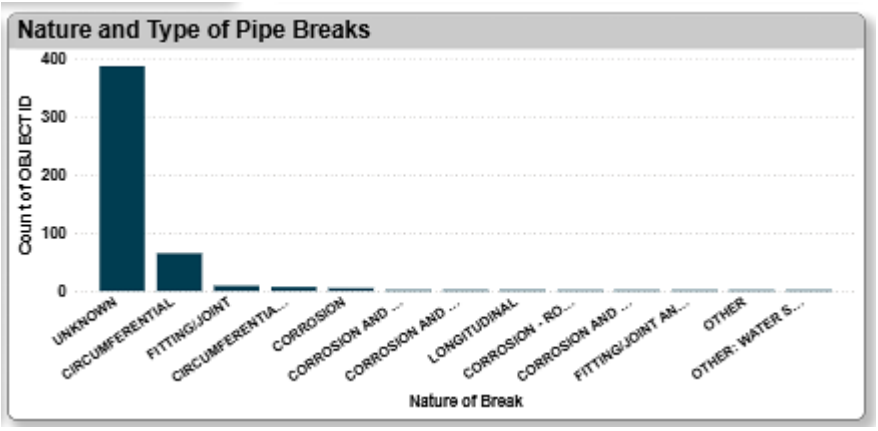
- **98.13%** of break incidents occurred in the **main pipes**
- Only **1.87%** were in **service lines**

Key Insight:

Major infrastructure issues are concentrated in the main distribution network, highlighting the need for focused maintenance efforts on these critical pipelines.

---

4. Nature and Cause of Breaks



Break causes are categorized as:

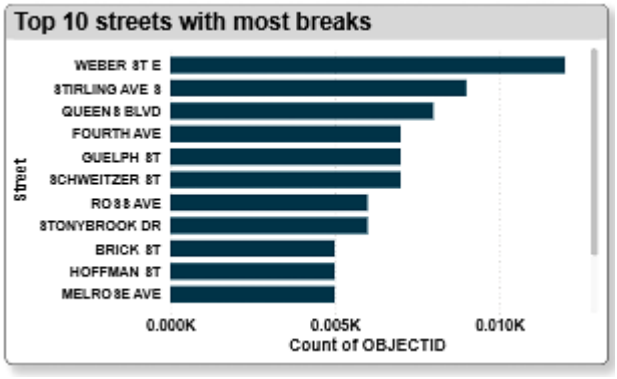
- **Corrosion**
- **Circumferential and Longitudinal Cracks**
- **Fitting/Joint Failures**
- **Unknown Causes**

Key Insight:

Corrosion is a leading cause of pipe failure. A significant number of cases are marked as **"Unknown"**, indicating a need to improve data collection during field inspections.

---

5. Top Streets with Most Breaks



The streets with the highest number of reported breaks include:

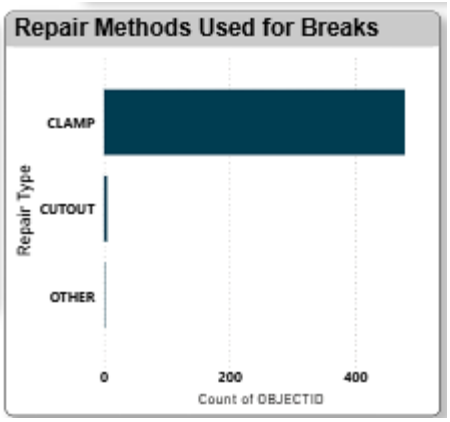
- **Weber Street East** (highest)
- **Stirling Avenue South**
- **Queens Boulevard**
- **Courtland Avenue East**

Key Insight:

These streets consistently appear in the top ranks for incidents. This suggests a localized infrastructure problem and points to areas needing urgent repair or pipe replacement.

---

6. Repair Methods Used



The most common repair methods include:

- **Clamp**
- **Cutout**
- **Other minor fixes**

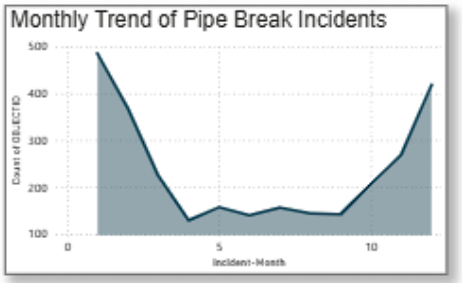
Key Insight:

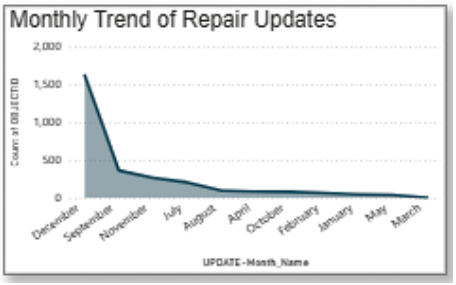
Most breaks are often fixed through simple repairs, suggesting that many failures could be manageable if identified early. However, this also indicates a lack of long-term solutions, which may result in recurring problems.

---

7. Temporal Analysis of Breaks

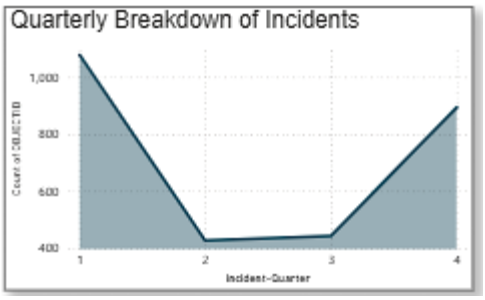
a. Monthly Trends





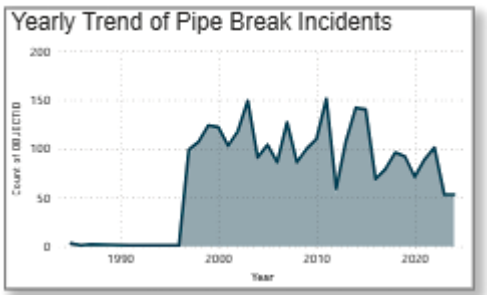
Break incidents increase noticeably towards the **end of the year**, particularly in **November and December**.

b. Quarterly Trends



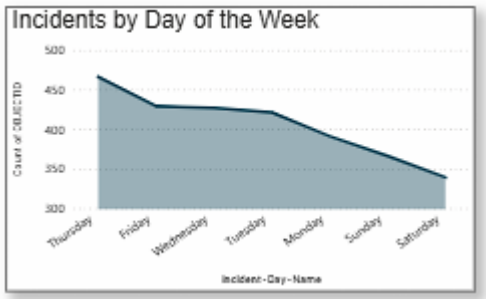
Most incidents occur in **Q1 and Q4**, with Q4 showing the highest frequency.

c. Yearly Trend



There is a clear **upward trend in break incidents** since the year 2000, peaking after **2010**, which may reflect aging infrastructure and increased stress on the network.

d. Day-of-Week Analysis



The highest number of breaks are reported on:

- **Thursdays**
- **Fridays**

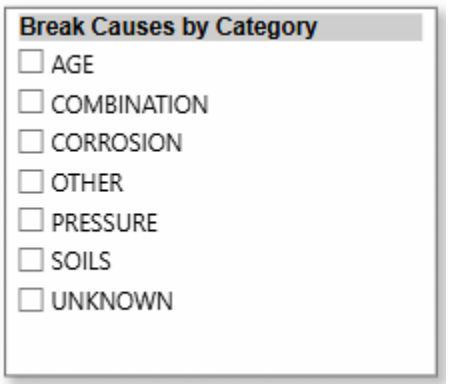
Key Insight:

Time-based patterns suggest that **weather**, **water pressure variations**, and **increased weekday usage** may contribute to higher break incidents. The spike at year-end may relate to freezing conditions or system overload.

8. Filters Available

To enhance interactivity and allow for more focused analysis, the Power BI dashboard includes several slicers and filters. These enable users to dynamically adjust the visuals based on specific attributes or time frames. The available filters include:

- **Break causes by category:**



  - 219 breaks were caused by **pipe age**, and 217 of them were repaired.
  - 137 breaks were due to a **combination of factors**, and all were repaired.
  - 111 breaks happened due to **corrosion**, with 109 repaired.
  - 9 breaks were caused by **faulty installation**, and all were repaired.
  - 23 breaks occurred due to **pressure**, all were repaired.
  - 20 breaks were related to **soil conditions**, all were repaired.
  - 2.29K breaks were due to **other reasons**, with 2.27K repaired.
  - 35 breaks had an **unknown cause**, and all were repaired.

  This analysis shows that most pipe breaks happen because of three main factors: the age of the pipes, corrosion, and other unspecified issues. Despite these different causes, repairs are usually completed quickly and effectively, indicating a strong maintenance system. However, the high number of breaks related to old pipes and corrosion highlights the need for regular inspections and timely replacements to prevent future breaks.

- **Break Incident Month:**

  - 485 pipe breaks were recorded in January.
  - 368 pipe breaks were recorded in February.
  - 224 pipe breaks were recorded in March.
  - 129 pipe breaks were recorded in April.
  - 157 pipe breaks were recorded in May.
  - 140 pipe breaks were recorded in June.
  - 156 pipe breaks were recorded in July.
  - 144 pipe breaks were recorded in August.
  - 142 pipe breaks were recorded in September.
  - 207 pipe breaks were recorded in October.

- 269 pipe breaks were recorded in November.
- 418 pipe breaks were recorded in December.

The data clearly shows that pipe breaks in Kitchener peak during the winter months. January and December experience the highest number of breaks, with 485 and 418 incidents, respectively. In contrast, the fewest breaks occur in the spring and early summer, particularly in April (129) and June (140). This seasonal pattern suggests that cold temperatures, ground frost, and possibly freeze-thaw cycles significantly contribute to the stresses placed on pipeline infrastructure.

Understanding the Seasonal Influence on Pipe Breaks

Kitchener's climate is classified as humid continental, with cold, snowy winters. Average high temperatures range from about −3 °C in January to around −2 °C in February, with average lows between −11 °C and −10 °C. Wikipediacurrentresults.comWeather Atlas. Snowfall is substantial during winter—Kitchener sees roughly 44 cm in January and 30 cm in February currentresults.comWeather Atlas. Persistent frost and low temperatures can deepen ground freezing well below the typical pipe depth, causing soil shifts that stress or break pipes CityNews Kitchener.

City officials note that extended periods of –20 °C to –30 °C lasting 10 to 20 days are especially damaging, as they drive frost deeper into the ground and increase the likelihood of main breaks CityNews Kitchener. In response, they often recommend letting taps drip—especially basement taps—to keep water flowing and reduce the risk of freezing CityNews KitchenerHomes and GardensReal Simple.

This analysis highlights that winter weather significantly affects pipe break ratios. It reinforces our observation that sustained extreme cold and deep frost are key factors contributing to pipe failures in Kitchener. To reduce break rates during winter, it is crucial to implement mitigation strategies such as proper insulation, methods to prevent dripping, infrastructure upgrades, and monitoring ground frost.

9. Recommendations

1. Replace High-Risk Pipe Materials

- Prioritize replacement of **Cast Iron** and other outdated materials.
- Focus initial upgrades on streets with repeated failures.

2. Strengthen Preventive Maintenance

- Monitor pipes in **Weber St E, Stirling Ave S, and Queens Blvd** proactively.
- Use pressure sensors and flow monitors to predict stress points.

3. Improve Field Data Collection

- Train staff to record **detailed and accurate break causes**.
- Introduce digital data entry tools for real-time updates.

4. Focus on Main Line Rehabilitation

- Allocate funds to upgrade **main distribution lines**, as they represent 98% of failures.

5. Prepare for Seasonal Peaks

- Increase inspection and maintenance teams in **Q4 and Q1**, especially before winter.

6. Reevaluate Repair Methods

- Review the success rates of **clamp and cutout** methods.
- Where failures recur, prioritize full pipe replacement over temporary repairs.

---

10. Conclusion

This project highlights the benefits of using Python for data cleaning in combination with Power BI for data visualization to analyze infrastructure data. This data-driven approach revealed significant patterns, identifying aging materials, seasonal stress, and critical hotspot locations as the main contributors to water pipe breaks.

The insights gained from this analysis can assist utility managers in making informed decisions about preventive maintenance, infrastructure investments, and policy changes. Ultimately, this will contribute to a safer and more reliable water distribution system for the community.
