TEXAS ADVANCED COMPUTING CENTER

WWW.TACC.UTEXAS.EDU

The University of Texas at Austin

Intro to Pandas

The Chef's Knife

PRESENTED BY:

Intro to Pandas + Lambda Functions + Advanced Techniques

🎯 Learning Goals:

- Ultimate Goal: Understand what Pandas is and why it’s useful.
- Load and inspect real-world data (Austin Traffic).
- Perform basic data exploration and statistics.
- Query and filter data (using conditions and lambda functions).
- Connect lambda functions to their broader use in Machine Learning.
- Explore advanced Pandas techniques like groupby, sorting, and aggregation.

Let's Do This!

Fire up Eureka!

Download Austin Real-Time Traffic Incident Reports from:

https://data.austintexas.gov/Transportation-and-Mobility/Real-Time-Traffic-Incident-Reports/dx9v-zd7x/about_data

GOOGLE: Austin Real-Time Traffic Incidents

Let's Do This!

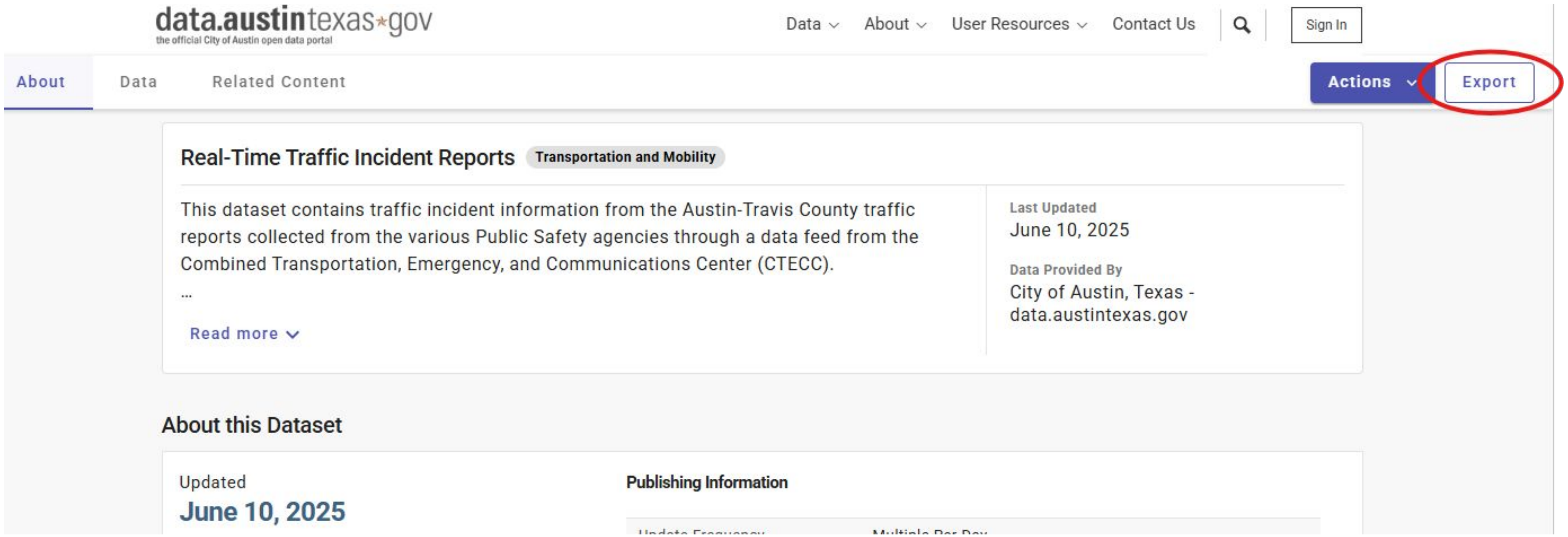
Fire up Eureka!

Download Austin Real-Time Traffic Incident Reports from:

https://data.austintexas.gov/Transportation-and-Mobility/Real-Time-Traffic-Incident-Reports/dx9v-zd7x/about_data

GOOGLE: Austin Real-Time Traffic Incidents

Get The Data



Real-Time Traffic Incident Reports Transportation and Mobility

This dataset contains traffic incident information from the Austin-Travis County traffic reports collected from the various Public Safety agencies through a data feed from the Combined Transportation, Emergency, and Communications Center (CTECC).
...

Read more

Last Updated
June 10, 2025

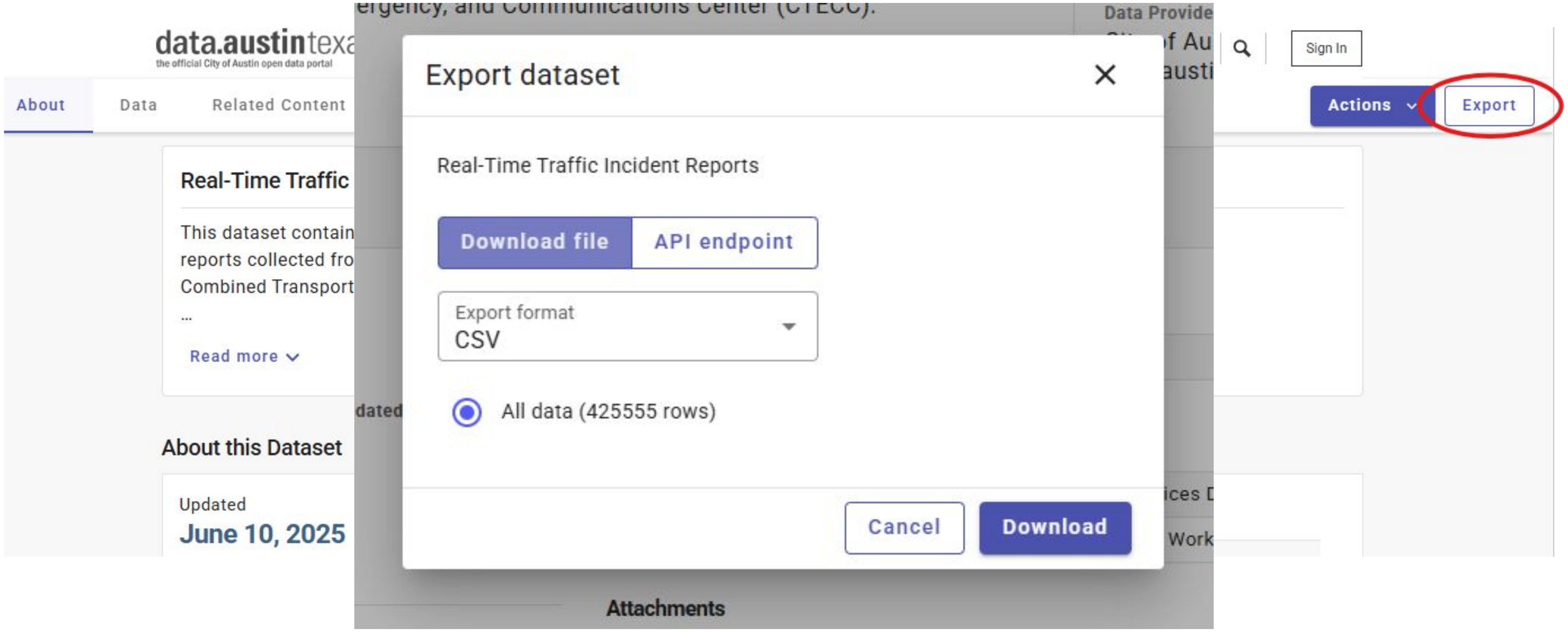
Data Provided By
City of Austin, Texas - data.austintexas.gov

About this Dataset

Updated
June 10, 2025

Publishing Information

Get The Data


Export dataset

Real-Time Traffic Incident Reports

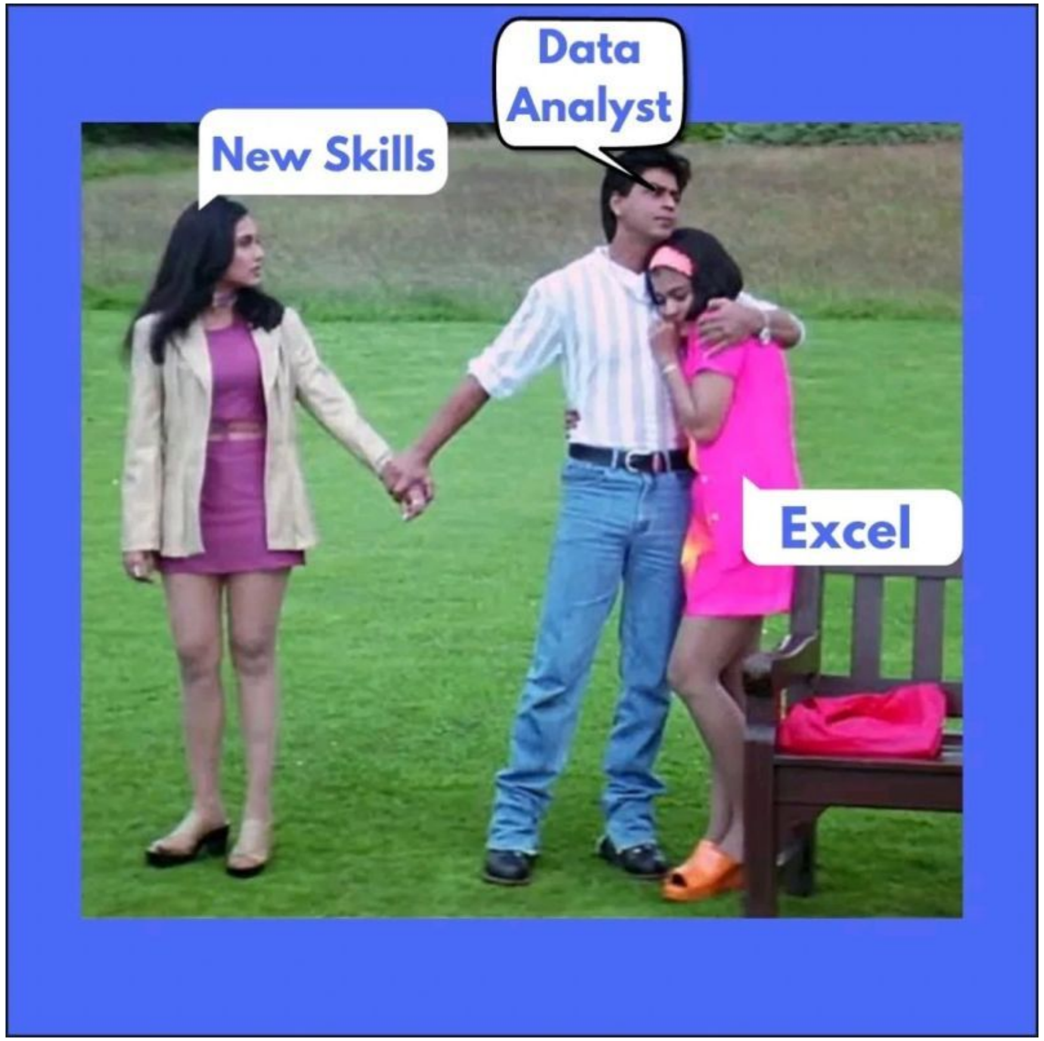
Export format: CSV

All data (425555 rows)

🧭 Part 1: Introduction to Pandas

What Is Pandas?

- Python library for working with tabular data.
- Like Excel, but way more powerful and programmable.



TACC

✅ Hands-on Steps:

```
import pandas as pd

# Load the dataset
df = pd.read_csv("Austin_traffic_data.csv")

# Preview
df.head()
```

🔍 Inspecting the Data:
What do these commands do

```
df.shape  # (rows, columns)
df.columns
df.info()
df.describe()
```

🔍 Inspecting the Data:

```
# Shape of the data (rows, columns)
df.shape

# Column names
df.columns

# Quick summary of the dataframe
df.info()

# Summary statistics (only numerical columns)
df.describe()
```

💬 Mini Challenge:

You may have to Google or ask ChatGPT

Print how many unique values are in the 'Issue Reported' column.

💬 Mini Challenge: A solution

```
# How many unique types of issues are reported?
df['Issue Reported'].nunique()

# Optional: list the top 10 most frequent issues
df['Issue Reported'].value_counts().head(10)
```

🧭 Part 2: Querying and Filtering Data

How Do We Ask Questions of Our Data?

```
# Simple query
df[df['Issue Reported'] == 'Crash Urgent']

# Incidents on a specific street
df[df['Address'].str.contains('IH 35',
na=False)]

# Multiple conditions
df[(df['Issue Reported'] == 'Crash Urgent') &
(df['Address'].str.contains('IH 35', na=False))]
```

💬 Mini Challenge:

You may have to Google or ask ChatGPT

How many “Traffic Hazard” incidents occurred in 2023?

How about in 2024?

💬 Mini Challenge:

You may have to Google or ask ChatGPT

```
# Convert date column to datetime
df['Published Date'] = pd.to_datetime(df['Published Date'])

# Filter for year 2023 and "Traffic Hazard"
hazards_2023 = df[
    (df['Published Date'].dt.year == 2023) &
    (df['Issue Reported'] == 'Traffic Hazard')
]

# Count
len(hazards_2023)
```

🧭 Part 3: Lambda Functions + Side Quest

What Is a Lambda Function?

- Anonymous one-line function: `lambda x: x + 1`
- Useful for short operations in data transformations

```
# Clean up issue descriptions
df['Issue Lower'] = df['Issue Reported'].apply
(lambda x: x.lower() if pd.notnull(x) else x)


# Extract year from date
df['Year'] = pd.to_datetime
(df['Published Date']).apply(lambda x: x.year)
```

⚔️ 🧭 Side Quest: Lambda in Machine Learning

What Is a Lambda Function?

lambda functions are often used for quick feature engineering or inside pipelines

```
from sklearn.preprocessing import FunctionTransformer

# Use lambda inside a pipeline step (optional preview)
transformer = FunctionTransformer(lambda x: x**2)
```

💬 Macro Challenge:

You may have to Google or ask ChatGPT

Create a new column that returns True if the incident occurred in rush hour (e.g., 7–9am or 4–6pm).

💬 Macro Challenge:

You may have to Google or ask ChatGPT

```
# Extract hour from datetime

df['Hour'] = df['Published Date'].dt.hour


# Define rush hour times (7–9am and 4–6pm)

def is_rush_hour(hour):

    return (7 <= hour <= 9) or (16 <= hour <= 18)


df['Rush Hour'] = df['Hour'].apply(is_rush_hour)


# Preview

df[['Published Date', 'Hour', 'Rush Hour']].head()
```

🧭 Part 4: Advanced Pandas Techniques

Level Up with Pandas

```
# Grouping
df.groupby('Issue Reported').size().sort_values(ascending=False).head(10)

# Time-based grouping
df['Date'] = pd.to_datetime(df['Published Date'])
df['Month'] = df['Date'].dt.month
monthly_counts = df.groupby('Month').size()

# Aggregations
df.groupby('Issue Reported')['Traffic Report ID'].count()

# Sorting
df.sort_values(by='Published Date', ascending=False).head()
```

💬 Mini Challenge:

You may have to Google or ask ChatGPT

Which month has the highest number of “Crash Urgent” reports?

💬 Mini Challenge:

You may have to Google or ask ChatGPT

```
crash_urgent = df[df['Issue Reported'] == 'Crash Urgent']
crash_urgent.groupby('Month').size().sort_values(ascending=False)
```

Question? Comments?