Python and SQL: intro / SQL platforms

Ewa Weychert

Class 11: Django data explorer (CSV)

Learning goals (today)

By the end of 90 minutes, students will be able to:

- Explain how Streamlit UI concepts map to Django (server-rendered) web apps
- Build a Django app that:
  - uploads a CSV,
  - shows overview + missingness + types,
  - renders plots for numeric/categorical columns
- Understand key web concepts: HTTP, forms, templates, sessions, static/media

Agenda (90 minutes)

1. (0–10) Why Django instead of Streamlit (architecture + tradeoffs)
2. (10–25) Project setup + URL routing + templates
3. (25–40) File upload flow (POST + CSV parsing)
4. (40–55) Data panels: overview + missingness + dtypes + unique counts
5. (55–70) Tabs + controls (Bootstrap + form inputs)
6. (70–85) Charts (Matplotlib → base64 images in HTML)
7. (85–90) Wrap-up + next steps

Step 0: Fix pip on macOS (PEP 668) using a venv

On Homebrew Python, install packages inside a virtual environment:

```
# 1) go to your class folder
cd ~/Desktop/class_11

# 2) create a venv inside the folder (recommended name: .venv)
python3 -m venv .venv

# 3) activate it (zsh/macOS)
source .venv/bin/activate

# 4) upgrade pip (optional but recommended)
python -m pip install --upgrade pip

# 5) install requirements
python -m pip install django pandas matplotlib
```

Step 1: Create Django project + app

```
# still inside the venv (.venv) and inside ~/Desktop/class_11

# create the project
python -m django startproject dataexplorer

cd dataexplorer

# create the app
python manage.py startapp explorer

# run migrations (recommended)
python manage.py migrate

# start server
python manage.py runserver
```

Open `http://127.0.0.1:8000/`.

Step 2: Add app to settings.py

Edit: `dataexplorer/settings.py`

```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",

    # add this:
    "explorer",
]
```

Save the file (server auto-reloads).

Step 3: Project URLs (include explorer URLs)

Edit: `dataexplorer/urls.py`

```python
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path("admin/", admin.site.urls),
    path("", include("explorer.urls")),
]
```

Step 4: App URLs (route / to the view)

Create: `explorer/urls.py`

```
from django.urls import path
from . import views

urlpatterns = [
    path("", views.upload_and_explore, name="upload_and_explore")
        ,
]
```

Important: two common bugs you hit

- **Bug A:** `from __future__ import annotations` must be **first line** in the file.
  - Easiest: **remove it** (Python 3.13 is fine without it).
- **Bug B:** `TemplateDoesNotExist explorer/page.html`
  - Your template file must be named **page.html**, not **pages.html**.
  - And it must be located at: `explorer/templates/explorer/page.html`.

Step 5: views.py (FULL correct file) 1

Replace all content of: `explorer/views.py` with:

```
import base64
import io
from typing import Any

import matplotlib
matplotlib.use("Agg")
import matplotlib.pyplot as plt
import pandas as pd
from django.shortcuts import render

def fig_to_base64_png(fig) -> str:
    buf = io.BytesIO()
    fig.tight_layout()
    fig.savefig(buf, format="png", dpi=150)
    plt.close(fig)
    buf.seek(0)
    return base64.b64encode(buf.read()).decode("utf-8")
```

Step 5: views.py (FULL correct file) 2

```
def df_to_table_html(df: pd.DataFrame) -> str:
    return df.to_html(
        classes="table table-sm table-striped table-bordered",
        border=0)

def safe_int(value: Any, default: int, min_v: int, max_v: int) ->
     int:
    try:
        v = int(value)
    except (TypeError, ValueError):
        return default
    return max(min_v, min(max_v, v))
```

Step 5: views.py (FULL correct file) 3

```
def overview_context(df: pd.DataFrame, n_head: int) -> dict[str,
    Any]:
    num_df = df.select_dtypes(include="number")
    cat_df = df.select_dtypes(exclude="number")

    ctx: dict[str, Any] = {
        "shape": df.shape,
        "columns": list(df.columns),
        "head_table": df_to_table_html(df.head(n_head)),
        "num_desc": None,
        "cat_desc": None,
    }
    if not num_df.empty:
        ctx["num_desc"] = df_to_table_html(num_df.describe())
    if not cat_df.empty:
        ctx["cat_desc"] = df_to_table_html(cat_df.describe(include
            ="all"))
    return ctx
```

Step 5: views.py (FULL correct file) 4

```
def missing_types_context(df: pd.DataFrame) -> dict[str, Any]:
    miss_count = df.isna().sum().to_frame("missing_count")
    miss_pct = (df.isna().mean() * 100).round(2).to_frame("
        missing_%")
    dtypes = df.dtypes.astype(str).to_frame("dtype")
    nunique = df.nunique(dropna=True).to_frame("n_unique")

    return {
        "miss_count": df_to_table_html(miss_count),
        "miss_pct": df_to_table_html(miss_pct),
        "dtypes": df_to_table_html(dtypes),
        "nunique": df_to_table_html(nunique),
    }
```

Step 5: views.py (FULL correct file) 5

```python
def continuous_plots(df: pd.DataFrame, bins: int) -> list[dict[
    str, str]]:
    images: list[dict[str, str]] = []
    for col in df.select_dtypes(include="number").columns:
        data = df[col].dropna()
        if data.empty:
            continue
        fig, ax = plt.subplots()
        ax.hist(data, bins=bins)
        ax.set_title(str(col))
        ax.set_xlabel(str(col))
        ax.set_ylabel("Count")
        images.append({"name": str(col), "png": fig_to_base64_png(
            fig)})
    return images
```

Step 5: views.py (FULL correct file) 6

```
def categorical_plots(df: pd.DataFrame, top_n: int) -> list[dict[
    str, str]]:
    images: list[dict[str, str]] = []
    for col in df.select_dtypes(exclude="number").columns:
        vc = df[col].astype(str).value_counts().head(top_n)
        if vc.empty:
            continue
        fig, ax = plt.subplots()
        ax.bar(vc.index.astype(str), vc.values)
        ax.set_title(str(col))
        ax.set_xlabel(str(col))
        ax.set_ylabel("Count")
        ax.tick_params(axis="x", rotation=45)
        images.append({"name": str(col), "png": fig_to_base64_png(
            fig)})
    return images
```

Step 5: views.py (FULL correct file) 7

```
def build_context(df: pd.DataFrame, n_head: int, bins: int, top_n
    : int) -> dict[str, Any]:
    ctx: dict[str, Any] = {"empty": False}
    ctx.update(overview_context(df, n_head))
    ctx.update(missing_types_context(df))
    ctx["continuous_imgs"] = continuous_plots(df, bins)
    ctx["categorical_imgs"] = categorical_plots(df, top_n)
    ctx["controls"] = {"n_head": n_head, "bins": bins, "top_n":
        top_n}
    return ctx
```

Step 5: views.py (FULL correct file) 8

```
def upload_and_explore(request):
    default_controls = {"n_head": 10, "bins": 20, "top_n": 10}
    if request.method == "POST" and request.FILES.get("csv_file")
        :
        n_head = safe_int(request.POST.get("n_head"),
            default_controls["n_head"], 1, 200)
        bins = safe_int(request.POST.get("bins"), default_controls
            ["bins"], 5, 50)
        top_n = safe_int(request.POST.get("top_n"),
            default_controls["top_n"], 3, 30)
        csv_file = request.FILES["csv_file"]
```

Step 5: views.py (FULL correct file) 9

```
    try:
        df = pd.read_csv(csv_file)
    except Exception as e:
        return render(
            request,
            "explorer/page.html",
            {"empty": True, "error": f"Could not read CSV: {e}
                ","controls": default_controls},
        )

    return render(
        request,
        "explorer/page.html",
        build_context(df, n_head, bins, top_n),
    )
```

Step 5: views.py (FULL correct file) 10

```
    return render(
        request,
        "explorer/page.html",
        {"empty": True, "controls": default_controls},
    )
```

Step 6: Create template folders + page.html

Create this exact path:

```
explorer/
  templates/
    explorer/
      page.html
```

Important: your file must be named `page.html` (not `pages.html`).

Step 7: page.html (minimal working template)1

Create: `explorer/templates/explorer/page.html`

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>CSV Explorer</title>
  <!-- Bootstrap (simple CDN for demo) -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/
      css/bootstrap.min.css" rel="stylesheet">
</head>
<body class="container py-4">

<h2 class="mb-3">CSV Explorer</h2>

{% if error %}
  <div class="alert alert-danger">{{ error }}</div>
{% endif %}
```

Step 7: page.html (minimal working template) 2

```
<form method="post" enctype="multipart/form-data" class="mb-4">
  {% csrf_token %}
  <div class="mb-2">
    <input type="file" class="form-control" name="csv_file"
        accept=".csv" required>
  </div>

  <div class="row g-2">
    <div class="col-md-4">
      <label class="form-label">Rows in preview</label>
      <input class="form-control" type="number" name="n_head"
          value="{{ controls.n_head }}" min="1" max="200">
    </div>
    <div class="col-md-4">
      <label class="form-label">Histogram bins</label>
      <input class="form-control" type="number" name="bins" value
          ="{{ controls.bins }}" min="5" max="50">
    </div>
```

Step 7: page.html (minimal working template) 3

```
    <div class="col-md-4">
      <label class="form-label">Top N categories</label>
      <input class="form-control" type="number" name="top_n"
          value="{{ controls.top_n }}" min="3" max="30">
    </div>
  </div>

  <button class="btn btn-primary mt-3" type="submit">Explore</
      button>
</form>

{% if not empty %}

<ul class="nav nav-tabs" role="tablist">
  <li class="nav-item">
    <button class="nav-link active" data-bs-toggle="tab" data-bs-
        target="#overview" type="button">Overview</button>
```

Step 7: page.html (minimal working template) 4

```
  </li>
  <li class="nav-item">
    <button class="nav-link" data-bs-toggle="tab" data-bs-target=
        "#missing" type="button">Missing & Types</button>
  </li>
  <li class="nav-item">
    <button class="nav-link" data-bs-toggle="tab" data-bs-target=
        "#cont" type="button">Continuous</button>
  </li>
  <li class="nav-item">
    <button class="nav-link" data-bs-toggle="tab" data-bs-target=
        "#cat" type="button">Categorical</button>
  </li>
</ul>
```

Step 7: page.html (minimal working template) 5

```
<div class="tab-content pt-3">

  <div class="tab-pane fade show active" id="overview">
    <p><strong>Shape:</strong> {{ shape }}</p>

    <h5>Preview</h5>
    {{ head_table|safe }}

    {% if num_desc %}
      <h5 class="mt-4">Numeric describe()</h5>
      {{ num_desc|safe }}
    {% endif %}

    {% if cat_desc %}
      <h5 class="mt-4">Categorical describe()</h5>
      {{ cat_desc|safe }}
    {% endif %}
  </div>
```

Step 7: page.html (minimal working template) 6

```
  <div class="tab-pane fade" id="missing">
    <div class="row">
      <div class="col-md-6">
        <h5>Missing count</h5>
        {{ miss_count|safe }}
      </div>
      <div class="col-md-6">
        <h5>Missing %</h5>
        {{ miss_pct|safe }}
      </div>
    </div>
    <div class="row mt-3">
      <div class="col-md-6">
        <h5>Dtypes</h5>
        {{ dtypes|safe }}
      </div>
      <div class="col-md-6">
        <h5>Unique values</h5>
        {{ nunique|safe }}
```

Step 7: page.html (minimal working template) 4

```
    </div>
  </div>
</div>

<div class="tab-pane fade" id="cont">
  <div class="row">
  {% for img in continuous_imgs %}
    <div class="col-md-6 mb-3">
      <h6>{{ img.name }}</h6>
      <img class="img-fluid" src="data:image/png;base64,{{ img.
          png }}" alt="Histogram {{ img.name }}">
    </div>
  {% empty %}
    <p><em>No numeric columns found.</em></p>
  {% endfor %}
  </div>
</div>

<div class="tab-pane fade" id="cat">
```

Step 7: page.html (minimal working template) 7

```
    <div class="row">
    {% for img in categorical_imgs %}
      <div class="col-md-6 mb-3">
        <h6>{{ img.name }}</h6>
        <img class="img-fluid" src="data:image/png;base64,{{ img.
            png }}" alt="Bar chart {{ img.name }}">
      </div>
    {% empty %}
      <p><em>No categorical columns found.</em></p>
    {% endfor %}
    </div>
  </div>

</div>
{% endif %}

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js
    /bootstrap.bundle.min.js"></script>
</body>
```

Step 8: Run and verify

```
# make sure venv is active
source .venv/bin/activate

# go to the folder with manage.py
cd ~/Desktop/class_11/dataexplorer

# run server
python manage.py runserver
```

Checkpoints:

- Visit `http://127.0.0.1:8000/` and see upload form
- Upload a CSV → tabs appear
- Overview table renders + plots render

Wrap-up: typical mistakes (and fixes)

- **TemplateDoesNotExist:** filename/path mismatch
  - Must be `explorer/templates/explorer/page.html`
  - And render it as `"explorer/page.html"`
- **AttributeError in urls.py:** view function name mismatch
  - urls.py must call exactly `views.upload_and_explore`
- **Future import SyntaxError:** delete it or put it first line

Q&A