Full Datasette Table Exporter

This script is designed to download and export full tables from a Datasette instance using streaming. It is especially useful when dealing with large tables that exceed the default row limit.

Purpose:

To automate the export of large-scale structured data (like `endpoint_dataset_issue_type_summary`) from the Open Digital Planning Datasette platform to local CSV files.

What the Script Does:

1. **Command-line Argument Parsing**:
   - Requires an `--output-dir` argument to specify where CSV files should be saved.
2. **Datasette Streaming Download**:
   - Reads the full table using the `?_stream=on` parameter to bypass row limits.
3. **DataFrame Conversion & Saving**:
   - Converts streamed data into a Pandas DataFrame.
   - Saves the result as a `.csv` file named after the table.
4. **Error Handling**:
   - Catches and logs download issues for any failed tables.

Example Table Handled:

- `endpoint_dataset_issue_type_summary` from the `performance` database.

This tool is modular—additional tables can be added to the `tables` dictionary and will be downloaded in batch.

Usage (Command Line):

```bash python full_datasette_export.py --output-dir ./outputs

In [ ]:
```
```python
import pandas as pd
import os
import argparse

def full_datasette_table(tables, output_dir):
    """
    Downloads full tables from Datasette in CSV format using streaming.

    Args:
```

```
        tables (dict): A dictionary where keys are table names and values are their
        output_dir (str): The directory to save the exported CSV files.
    """
    os.makedirs(output_dir, exist_ok=True)  # Ensure output directory exists

    for name, url in tables.items():
        full_url = f"{url}.csv?_stream=on"  # Enable full streaming of rows
        try:
            df = pd.read_csv(full_url)  # Load full dataset
            csv_name = f"{name}.csv"
            save_path = os.path.join(output_dir, csv_name)
            df.to_csv(save_path, index=False)  # Save to CSV without index
            print(f"Saved: {save_path}")
        except Exception as e:
            print(f"[ERROR] Failed to fetch {name}: {e}")

def parse_args():
    """
    Parses command-line arguments for specifying the output directory.

    Returns:
        argparse.Namespace: Parsed arguments containing the output directory path.
    """
    parser = argparse.ArgumentParser(description="Datasette batch exporter")
    parser.add_argument(
        "--output-dir",
        type=str,
        required=True,
        help="Directory to save exported CSVs"
    )
    return parser.parse_args()

if __name__ == "__main__":
    # Parse command-line arguments
    args = parse_args()

    # Dictionary of table names and their Datasette URLs
    tables = {
        "endpoint-dataset-issue-type-summary":
            "https://datasette.planning.data.gov.uk/performance/endpoint_dataset_is
    }

    # Run export
    full_datasette_table(tables, args.output_dir)
```