To write Python code that reads multiple CSV files, compares the values in the first column of each CSV, and then creates new CSV files containing only the rows with values that appear in the first column of all the CSVs, follow these steps:

### Steps:

1. \*\*Read multiple CSV files.\*\*

2. \*\*Extract the values in the first column from each CSV.\*\*

3. \*\*Find the intersection of those values across all CSVs.\*\*

4. \*\*Create new CSV files that contain only the rows where the first column value is in the intersection.\*\*

Here's an example Python code that performs these tasks:

```python

import csv

import os

def read\_first\_column(csv\_file):

"""Reads the first column of a CSV file and returns a set of its values."""

first\_column\_values = set()

with open(csv\_file, newline='', encoding='utf-8') as file:

reader = csv.reader(file)

for row in reader:

if row: # Ensure the row is not empty

first\_column\_values.add(row[0])

return first\_column\_values

def get\_intersection\_of\_first\_columns(csv\_files):

"""Finds the intersection of the first column values across multiple CSVs."""

if not csv\_files:

return set()

# Initialize the intersection with the first CSV file's first column

intersection\_set = read\_first\_column(csv\_files[0])

for csv\_file in csv\_files[1:]:

intersection\_set &= read\_first\_column(csv\_file)

return intersection\_set

def filter\_and\_write\_csv(csv\_file, intersection\_set, output\_dir):

"""Filters the rows of the CSV based on the intersection set and writes to a new file."""

output\_file = os.path.join(output\_dir, f"filtered\_{os.path.basename(csv\_file)}")

with open(csv\_file, newline='', encoding='utf-8') as infile, open(output\_file, mode='w', newline='', encoding='utf-8') as outfile:

reader = csv.reader(infile)

writer = csv.writer(outfile)

for row in reader:

if row and row[0] in intersection\_set:

writer.writerow(row)

def main(input\_files, output\_dir):

# Ensure the output directory exists

os.makedirs(output\_dir, exist\_ok=True)

# Get the intersection of the first column across all CSVs

intersection\_set = get\_intersection\_of\_first\_columns(input\_files)

# For each input file, filter and write the result to a new file

for csv\_file in input\_files:

filter\_and\_write\_csv(csv\_file, intersection\_set, output\_dir)

# Example usage:

input\_files = ['file1.csv', 'file2.csv', 'file3.csv'] # List of input CSV files

output\_dir = 'output\_csvs' # Directory to store filtered CSVs

main(input\_files, output\_dir)

```

### Explanation of the code:

1. \*\*`read\_first\_column`\*\*: This function reads the first column of a given CSV file and returns a set of its values. Using a set makes it easy to compare unique values.

2. \*\*`get\_intersection\_of\_first\_columns`\*\*: This function takes a list of CSV file paths, reads the first column from each, and finds the intersection of all the values. This intersection is the set of values that appear in the first column of all CSV files.

3. \*\*`filter\_and\_write\_csv`\*\*: This function filters the rows of a CSV based on whether the first column value is in the intersection set and writes the matching rows to a new file.

4. \*\*`main`\*\*: This is the main driver function that:

- Ensures the output directory exists.

- Computes the intersection of first-column values across all input CSVs.

- Creates filtered CSV files based on the intersection.

### How to Use:

1. Replace `input\_files` with the list of your actual CSV file paths.

2. Replace `output\_dir` with the directory where you want the filtered CSVs to be saved.

3. Run the script, and the filtered CSV files will be created in the specified output directory.

### Example Output:

If you have three CSV files:

- `file1.csv`:

```

id,name

1,John

2,Jane

3,Bob

```

- `file2.csv`:

```

id,name

2,Jane

3,Bob

4,Tom

```

- `file3.csv`:

```

id,name

3,Bob

4,Tom

5,Alice

```

After running the script, the output directory will contain a filtered version of each CSV file where only rows with the intersection of first column values are kept:

- `filtered\_file1.csv`:

```

id,name

2,Jane

3,Bob

```

- `filtered\_file2.csv`:

```

id,name

3,Bob

```

- `filtered\_file3.csv`:

```

id,name

3,Bob

```