Problem 1: Data Deduplication with Limited Memory

In data pipelines, a common issue is encountering duplicate records, especially when data originates from multiple sources. Write a Python function to remove duplicates from a **massive dataset** (too large to fit into memory). You have only 10% of the data size as available memory.

Task:

- You are given a generator function `get_data_stream()` that returns a stream of data (each data element is a tuple `(id, value)`).
- Implement a deduplication function that outputs the unique values without exceeding memory limits.

Signature:

```
python
Copy code
def deduplicate_large_dataset(data_stream: Iterable[Tuple[int, Any]]) ->
Set[int]:
    pass
```

Example:

```
python
Copy code
data_stream = [(1, "a"), (2, "b"), (1, "a"), (3, "c")]
unique_ids = deduplicate_large_dataset(data_stream)
# output should be {1, 2, 3}
```

Constraints:

- You can assume only a limited amount of memory is available.

Problem 2: Optimal Partitioning of a Dataset

Given a large dataset of integers, partition the dataset into `n` partitions such that the sum of elements in each partition is as **balanced as possible**.

Task:

- Write a Python function to split the input list into `n` partitions that have roughly equal sums.
- You need to return a list of partitions where the sum of the numbers in each partition is as close as possible to the others.

Signature:

```
python
Copy code
def partition_dataset(data: List[int], n: int) -> List[List[int]]:
    pass
```

Example:

```
python
Copy code
data = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
n = 3
partitions = partition_dataset(data, n)
# The output should be a list of 3 partitions with roughly equal sums.
```

Problem 3: Handling Duplicate Data in a Transactional Database

You are working with a table that stores transactional data for a system, but the data contains duplicates. Write a SQL query to **identify and remove duplicate rows** from a table while keeping the most recent record based on a timestamp.

Table: `transactions`

id	user_id	amount	transaction_date
1	101	500	2023-08-01 10:00:00
2	101	500	2023-08-01 10:00:00
3	102	250	2023-08-02 11:30:00
4	102	250	2023-08-02 11:30:00
5	103	750	2023-08-03 14:15:00

Task: Write a SQL query that removes duplicate records based on `user_id` and `transaction_date` while keeping the latest entry by `id`.

Expected Result:

- After the query, the table should only contain unique records with the latest `id` for each `user_id` and `transaction_date`.

Problem 4: Partitioning Data and Distributing Workloads

You have a table with user data, and you need to **partition the table** based on the geographic region of the users. Additionally, within each region, calculate the **total number of active users**. Develop the solution in both SQL and Python.

Table: `users`

user_id	region	active	join_date
1	North	1	2023-01-01
2	South	0	2023-02-10
3	East	1	2023-01-15
4	West	1	2023-03-22
5	North	1	2023-05-30

Task: Write a query to partition users by their `region` and calculate the number of active users (`active = 1`) in each region. Afterward, suggest an appropriate partitioning strategy for optimizing queries on this table based on region.

Expected Result:

region	active_users
North	2
South	0
East	1
West	1

Problem 5: Employee and Department Database (develop in both, SQL and Python)

You are given two tables: Employees and Departments.

Employees Table:

Table

EmployeeID	FirstName	LastName	DepartmentID	Salary
1	John	Doe	101	60000
2	Jane	Smith	102	75000
3	Emily	Davis	101	50000
4	Michael	Brown	103	80000
5	Sarah	Wilson	102	72000

Departments Table:

Table

DepartmentID	DepartmentName
101	HR
102	IT
103	Finance

Tasks:

1. **List all employees along with their department names.**
    - **Expected Output:**

Table

EmployeeID	FirstName	LastName	DepartmentName
1	John	Doe	HR
2	Jane	Smith	IT
3	Emily	Davis	HR
4	Michael	Brown	Finance
5	Sarah	Wilson	IT

2. **Find the total salary expenditure for each department.**
    - **Expected Output:**

Table

DepartmentName	TotalSalary
HR	110000
IT	147000
Finance	80000

3. **List departments that have more than one employee.**
    - **Expected Output:**

Table

DepartmentName
HR
IT