

## Input File Format

The test input file is a CSV file and follows the format described in **Section 6.4** of the lab description. The server IDs are [S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12] and Client records have IDs ranging from [1, 2, 3, ..., 3000]. These will remain consistent across all test cases.

## Example Test Cases

Set Number	Transactions	Live Servers	Contact Servers	Byzantine Servers
1	(21, 700, 1)	[S1, S2, ...]	[S1, S5, S9]	[S6, S11]
	(1301, 1302, 3)			
	(2001, 2650, 7)			
2	(301, 1302, 9)	[S1, S3, ...]	[S1, S5, S9]	[S6, S11]
	(501, 502, 3)			

Table 1: Example Test Cases

## Transaction Processing

- **Initial State:** Each client record starts with an initial balance of 10 units. Each server maintains all the client balances in the datastore.
- **Transaction Handling:** Your client must process and execute transactions in the order they are listed in the input file. For each transaction, the client determines the corresponding shard based on the Shard Mapping and sends the transaction to the Contact server of the appropriate cluster.

## NOTE

- Make sure your client does not process the next set of transactions unless explicitly instructed using the user prompt.
- Keep in mind that test case sets may be interdependent, meaning that server states and the datastore are carried forward from one set to the next.
- Furthermore, your implementation must strictly expose the functions outlined in Section 2.2 of the lab description, as these functions will be used during this period to verify the state of your servers.

**NOTE:** Even if a server is disconnected, the balance of the specified Client ID must still be displayed for that server when requested using the `PrintBalance()` function. Similarly, the datastore of all servers should be shown, regardless of their connection status and byzantine behavior.

# Test Case Descriptions

## Test Set 1 (Intra Shard)

- Test 1: Concurrent transaction in multiple clusters commit - Intra Shard
- Test 2: Transactions wait for lock - Intra Shard
- Test 3: Skip due to insufficient balance (Part 1) - Intra Shard
- Test 4: Skip due to insufficient balance (Part 2) - Intra Shard
- Test 5: No quorum in the cluster - Intra Shard
- Test 6: No execution due to incomplete previous set - Intra Shard

## Test Set 2 (Cross Shard)

- Test 1: Concurrent transaction in multiple clusters commit - Cross Shard
- Test 2: Transactions wait for lock - Cross Shard
- Test 3: Skip due to insufficient balance in coordinator cluster - Cross Shard
- Test 4: No quorum in the coordinator cluster - Cross Shard
- Test 5: No quorum in the participant cluster - Cross Shard
- Test 6: No execution due to incomplete previous set - Cross Shard

You can execute the two test files independently by running all test cases from **Test Set 1**, terminating the program, and then running all test cases from **Test Set 2**.

The descriptions provided above outline the test cases from the CSV input file. These test cases aim to cover all major scenarios and should be sufficient for validating your implementation.