

# Coding Pattern

## 1 Benchmark Function (main function)

**run\_benchmark()**

- **READ** the configuration parameters into the memory (config params `file.configuration.name ...`)
- save the current time as the start time of the 'benchmark' action
- **READ** the bulk file data into the partitioned collection bulk\_data\_partitions (config param 'file.bulk.name')  
partition key = modulo (ASCII value of 1st byte of key  
                          \* 251  
                          + ASCII value of 2nd byte of key),  
                          number partitions (config param 'benchmark.number.partitions')
- create a separate database connection (without auto commit behaviour) for each partition
- trial\_max = 0  
  trial\_min = 0  
  trial\_no = 0  
  trial\_sum = 0  
  **WHILE** trial\_no < config\_param 'benchmark.trials'  
    duration\_trial = **DO** **run\_trial**(database connections,  
                                    trial\_no,  
                                    bulk\_data\_partitions)  
    **IF** trial\_max == 0 OR duration\_trial > trial\_max  
      trial\_max = duration\_trial  
    **END IF**  
    **IF** trial\_min == 0 OR duration\_trial < trial\_min  
      trial\_min = duration\_trial  
    **END IF**  
    trial\_sum + duration\_trial  
  **ENDWHILE**
- partition\_key = 0  
  **WHILE** partition\_key < config\_param 'benchmark.number.partitions'  
    close the database connection  
  **ENDWHILE**

# Coding Pattern

- **WRITE** an entry for the action 'benchmark' in the result file (config param 'file.result.name')
- INFO Duration (ms) trial min. : trial\_min  
INFO Duration (ms) trial max. : trial\_max  
INFO Duration (ms) trial average : trial\_sum / config\_param 'benchmark.trials'
- INFO Duration (ms) benchmark run : duration\_benchmark

# Coding Pattern

## 2 Trial Function

```
run_trial(database connections,
          trial_no,
          bulk_data_partitions)

- save the current time as the start time of the 'trial' action

- INFO Start trial no. Trial_no

- create the database table (config param 'sql.create')
  IF error
    drop the database table (config param 'sql.drop')
    create the database table (config param 'sql.create')
  ENDIF

- DO run_insert(database connections,
               trial_no,
               bulk_data_partitions)

- DO run_select(database connections,
               trial_no,
               bulk_data_partitions)

- drop the database table (config param 'sql.drop')

- WRITE an entry for the action 'trial' in the result file (config param 'file.result.name')

- INFO Duration (ms) trial          : duration_trial

- RETURN duration = end time - start time
```

# Coding Pattern

## 3 Insert Control Function

```
run_insert(database connections,
           trial_no,
           bulk_data_partitions)

- save the current time as the start time of the 'query' action

- partition_key = 0
  WHILE partition_key < config_param 'benchmark.number.partitions'
    IF config_param 'benchmark.core.multiplier' = 0
      DO run_insert_helper(database connections(partition_key),
                          bulk_data_partitions(partition_key),
                          partition_key)

    ELSE
      DO run_insert_helper (database connections(partition_key),
                          bulk_data_partitions(partition_key,
                          partition_key)) as a thread

    ENDIF
  ENDWHILE

- WRITE an entry for the action 'query' in the result file (config param 'file.result.name')
```

# Coding Pattern

## 4 Insert Helper Function

```
run_insert_helper (database connection,
                  bulk_data_partition,
                  partition_key)

- INFO Start insert partition_key=partition_key

- count = 0
  collection batch_collection = empty
  WHILE iterating through the collection bulk_data_partition
    count + 1

    add the SQL statement in config param 'sql.insert' with the current bulk_data entry to the collection
    batch_collection

    IF config_param 'benchmark.batch.size' > 0
      IF count modulo config param 'benchmark.batch.size' = 0
        execute the SQL statements in the collection batch_collection
        batch_collection = empty
      ENDIF
    ENDIF

    IF config param 'benchmark.transaction.size' > 0
      AND count modulo config param 'benchmark.transaction.size' = 0
        commit
      ENDIF
    ENDWHILE

- IF collection batch_collection is not empty
  execute the SQL statements in the collection batch_collection
ENDIF

- commit

- INFO End    insert partition_key=partition_key
```

# Coding Pattern

## 5 Select Control Function

```
run_select(database connections,
           trial_no,
           bulk_data_partitions)

- save the current time as the start time of the 'query' action

- partition_key = 0
  WHILE partition_key < config_param 'benchmark.number.partitions'
    IF config_param 'benchmark.core.multiplier' = 0
      DO run_select_helper(database connections(partition_key),
                          bulk_data_partitions(partition_key,
                          partition_key)

    ELSE
      DO run_select_helper (database connections(partition_key),
                          bulk_data_partitions(partition_key,
                          partition_key) as a thread

    ENDIF
  ENDWHILE

- WRITE an entry for the action 'query' in the result file (config param 'file.result.name')
```

# Coding Pattern

## 6 Select Helper Function

```
run_select_helper (database connection,  
                  bulk_data_partition,  
                  partition_key)  
  
- INFO Start select partition_key=partition_key  
  
- execute the SQL statement in config param 'sql.select'  
  
- count = 0  
  WHILE iterating through the result set  
    count + 1  
  ENDWHILE  
  
- IF NOT count = size(bulk_data_partition)  
  display an error message  
ENDIF  
  
- INFO End    select partition_key=partition_key
```