OraBench - Coding Pattern.

Table of Contents

1 `Benchmark Function` (main function)

```
    run_benchmark()

        save the current time as the start of the 'benchmark' action

        READ the configuration parameters into the memory (config params
`file.configuration.name ...`)
        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 * 256 + 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_no = 0
        WHILE trial_no < config_param 'benchmark.trials'
            DO run_trial(database connections, trial_no, bulk_data_partitions)
        ENDWHILE

        partition_no = 0
        WHILE partition_no < config_param 'benchmark.number.partitions'
            close the database connection
        ENDWHILE

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

2 `Trial Function`

```
    run_trial(database connections, trial_no, bulk_data_partitions)
    INPUT: the database connections
           the current trial number
```

```
               the partitioned bulk data

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

            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_benchmark_insert(database connections, trial_no,
    bulk_data_partitions)
            DO run_benchmark_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')
```

3 Insert Control Function

```
        run_insert(database connections, trial_no, bulk_data_partitions)
        INPUT: the database connections
               the current trial number
               the partitioned bulk data

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

            partition_no = 0
            WHILE partition_no < config_param 'benchmark.number.partitions'
                IF config_param 'benchmark.core.multiplier' = 0
                    DO Insert(database connections(partition_no),
    bulk_data_partitions(partition_no))
                ELSE
                    DO Insert(database connections(partition_no),
    bulk_data_partitions(partition_no)) as a thread
                ENDIF
            ENDWHILE

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

4 Insert Function

```
        insert(database connection, bulk_data_partition)
        INPUT: the database connection
               the bulk data partition
            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
```

5 Select Control Function

```
    run_select(database connections, trial_no, bulk_data_partitions)
    INPUT: the database connections
           the current trial number
           the partitioned bulk data

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

        partition_no = 0
        WHILE partition_no < config_param 'benchmark.number.partitions'
            IF config_param 'benchmark.core.multiplier' = 0
                DO select(database connections(partition_no),
bulk_data_partitions(partition_no, partition_no)
            ELSE
                DO select(database connections(partition_no),
bulk_data_partitions(partition_no, partition_no) as a thread
            ENDIF
        ENDWHILE

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

6 Select Function

```
select(database connection, bulk_data_partition, partition_no)
INPUT: the database connection
       the bulk data partition
       the current partition number

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

    count = 0

    execute the SQL statement in config param 'sql.select'

    WHILE iterating through the result set
        count + 1
    ENDWHILE

    IF NOT count = size(bulk_data_partition)
        display an error message
    ENDIF
```