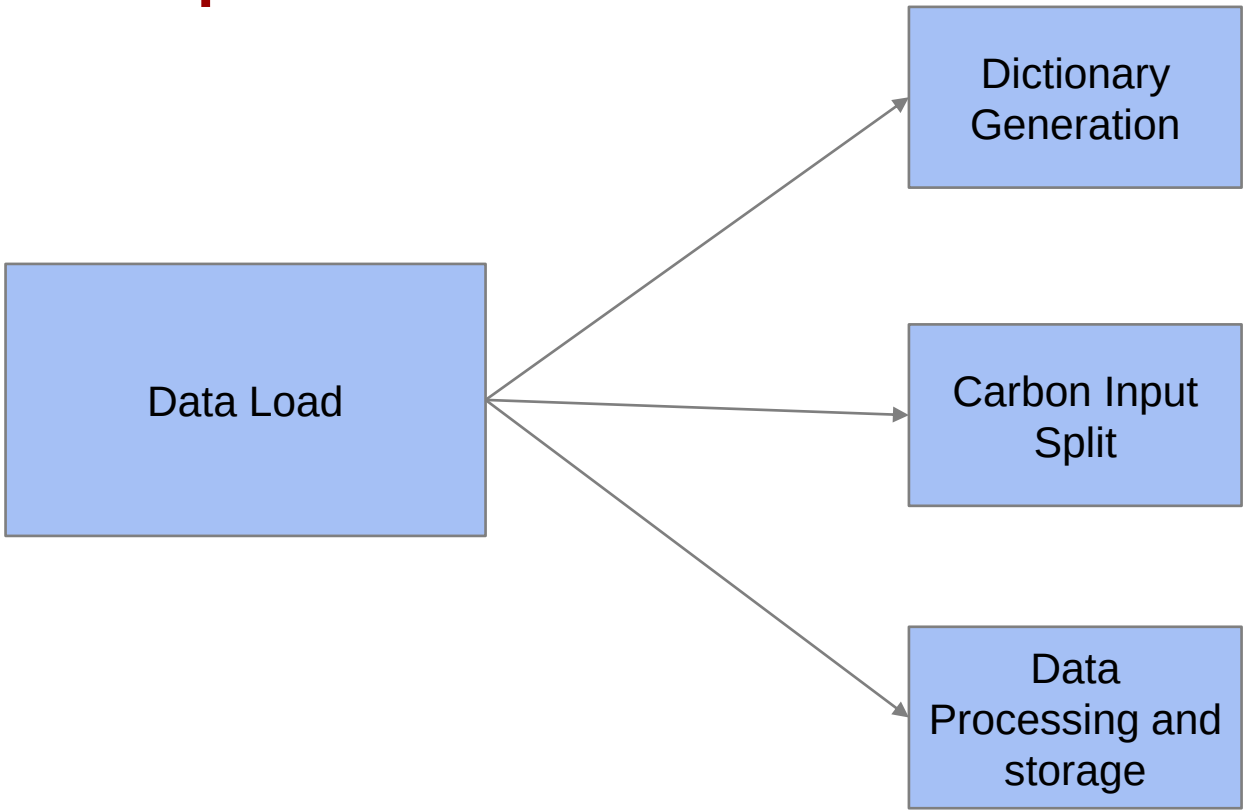
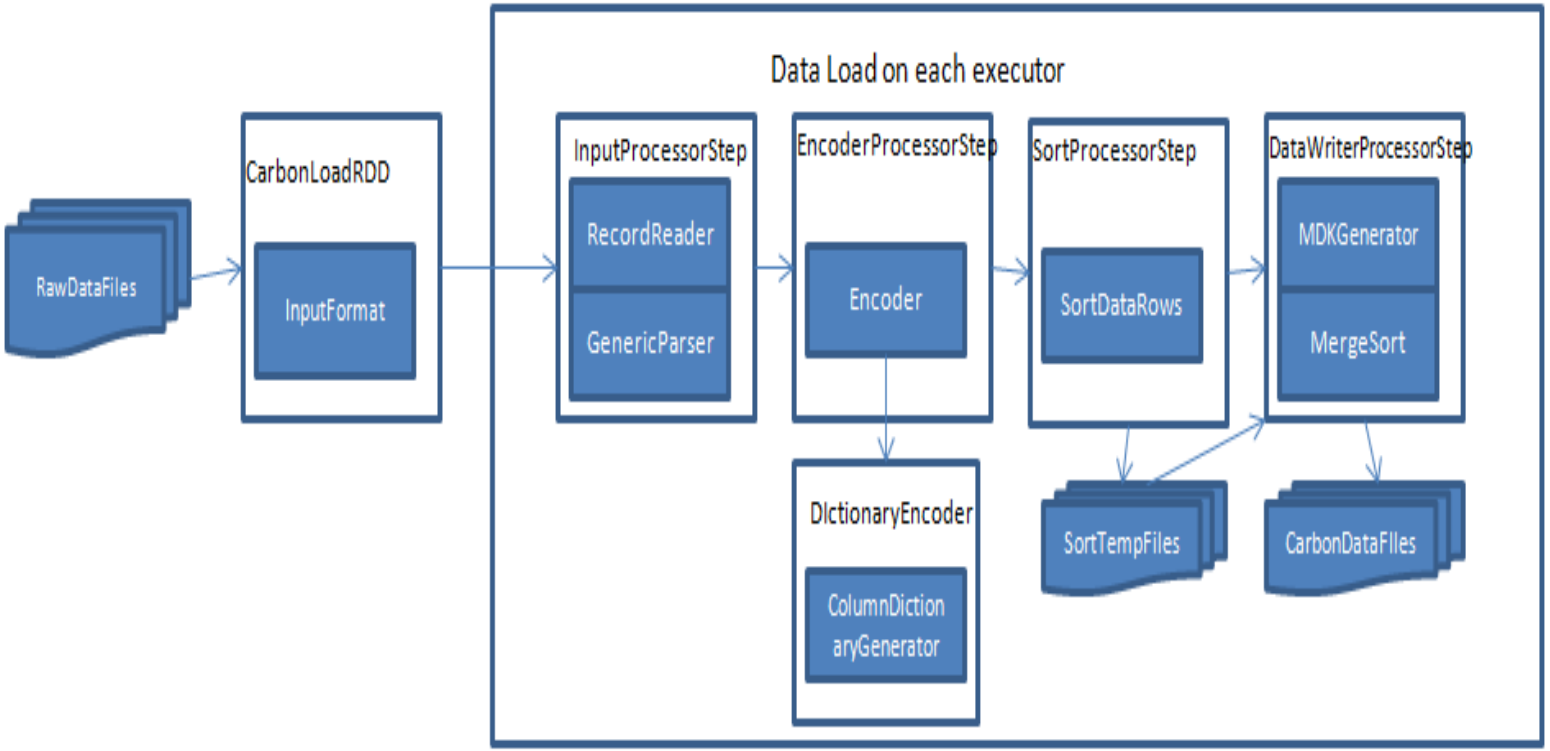
CarbonData Data Load Flow

Data Load Operations



Load Design Using InputFormat



Load Design

- **InputProcessorStep**: It does two jobs, 1. It reads data from RecordReader of InputFormat 2. Parse each field of column as per the data type.

- **EncoderProcessorStep**: It encodes each field with dictionary if requires.And combine all no dictionary columns to single byte array.

- **SortProcessorStep**: It sorts the data on dimension columns and write to intermediate files.

- **DataWriterProcessorStep**: It merge sort the data from intermediate temp files and generate mdk key and writes the data in carbondata format to store.

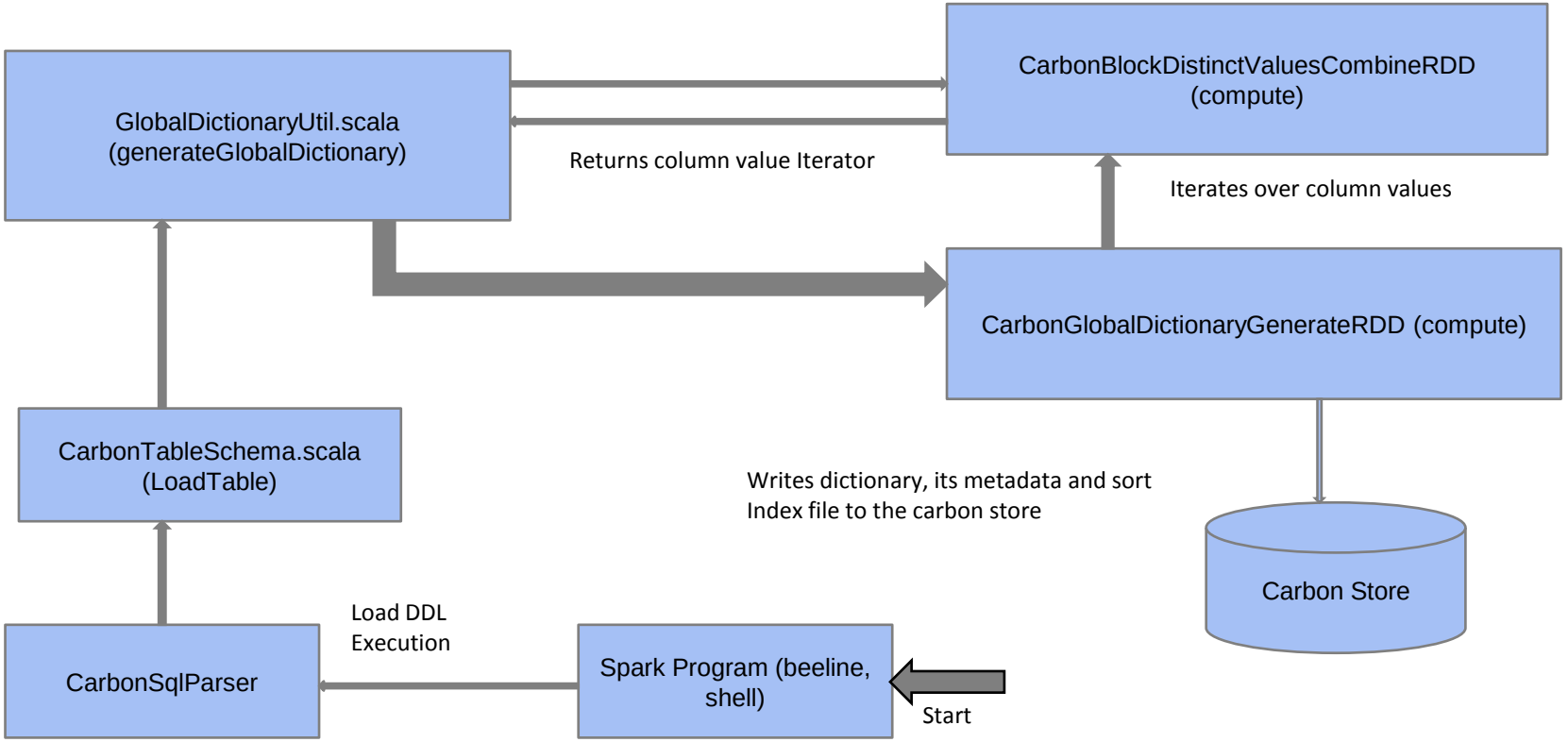
Major Interfaces

```
/**
 * This base interface for data loading. It can do transformation jobs as per the implementation.
 *
 */
public abstract class AbstractDataLoadProcessorStep {

public AbstractDataLoadProcessorStep(CarbonDataLoadConfiguration configuration,
    AbstractDataLoadProcessorStep child) {
}

  /**
   * The output meta for this step. The data returns from this step is as per this meta.
   * @return
   */
 public abstract  DataField[] getOutput();

  /**
   * Intialization process for this step.
   * @param configuration
   * @param child
   * @throws CarbonDataLoadingException
   */
  Public abstract void intialize(CarbonDataLoadConfiguration configuration, DataLoadProcessorStep child) throws
      CarbonDataLoadingException;

  /**
   * Tranform the data as per the implemetation.
   * @return Iterator of data
   * @throws CarbonDataLoadingException
   */
 public  Iterator<Object[]> execute() throws CarbonDataLoadingException;

  /**
   * Any closing of resources after step execution can be done here.
   */
  void close();
}
```

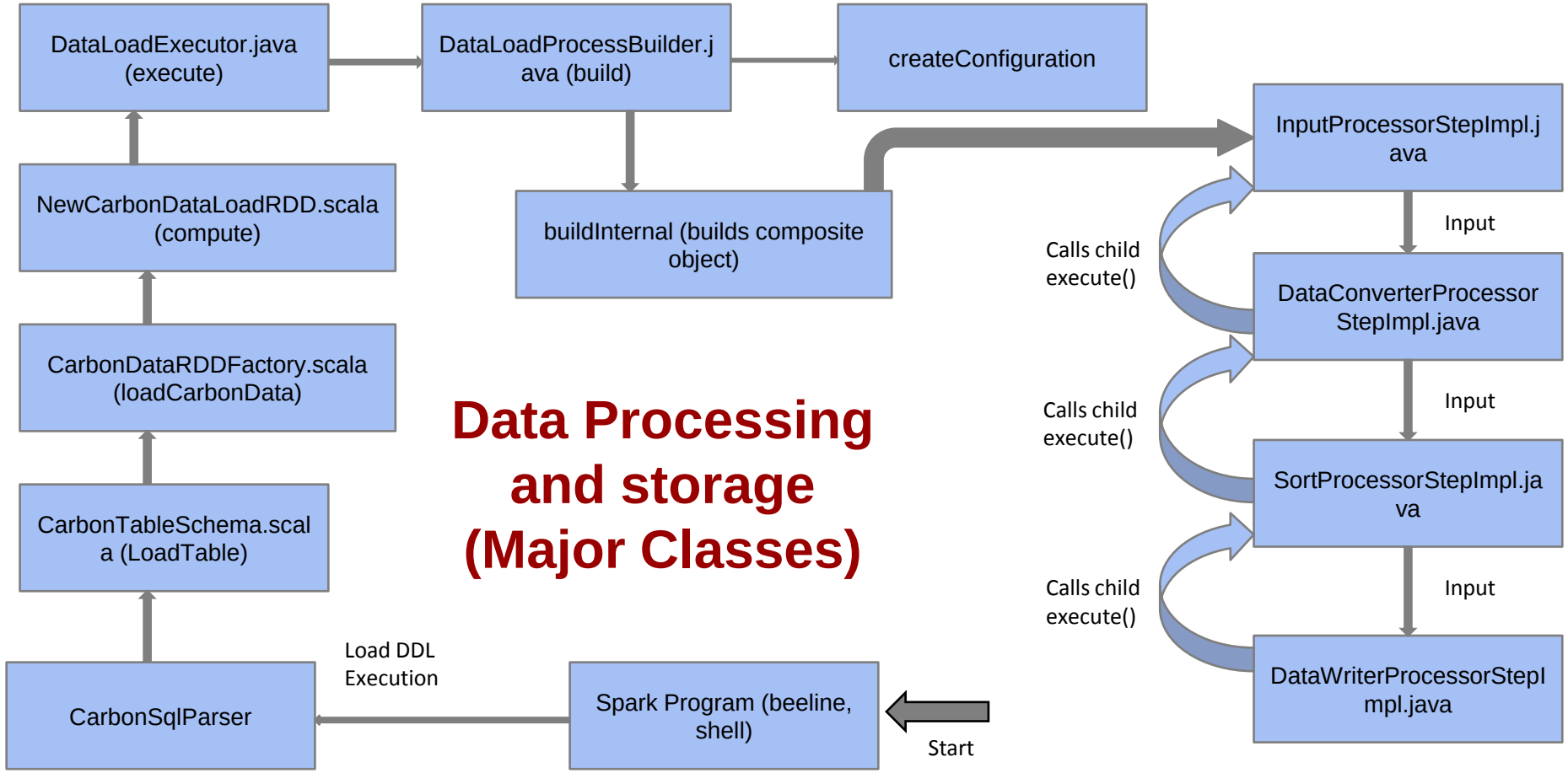
Dictionary Generation

- **Dictionary generation is a process where an integer key is generated for each unique value of a column.**
- **Each column has its won dictionary file.**
- **Output of dictionary generation step includes creation of below files.**
  1. <columnID>.dict -> This file contains the actual data as thrift byte buffer array object.
  2. <columnID>.dictMeta -> This includes the dictionary metadata for each load like startOffset, endOffset, chunkCount, minimum dictionary value and maximum dictionary value for a particular load. This file is also the commit for successful dictionary generation.
  3. <columnID>.sortIndex -> This file stores the sorted and its reverse index for the dictionary files.

Global Dictionary Generation(Major Classes)



Data Processing and storage (Major Classes)



Thank you