RDD : Resilient Distribute Datasets

#Loading External Files into RDD

```
products=sc.textFile("file:///home/hadoop/product.csv")
products.foreach(print)
```

#Loading Constant Data in RDD

```
customers_list=["Ankit Verma","Sunil Sharma","Pankaj Verma"]
customers.parallelize(customers_list)
products.foreach(print)
```

RDD Transform/Action Functions

1. **map(function)**

```
def increment(x):
     x=x+1
     return x


numbers = sc.parallelize(range(1,11))
new_numbers = numbers.map(increment)
new_numbers.collect()
```

#Lambda Function

```
Syntax=> lambda arguments : expression
Example=> lambda x : x+1

new_numbers = numbers.map(lambda x:x*10)
new_numbers.collect()
```

```
products=sc.textFile("file:///home/hadoop/product.csv")
products=products.map(lambda x : x.split(,))
new_products=products.map(lambda x : float(x[4])+((float(x[4])*.05))
new_products.foreach(print)
```

```
def PriceHike(x):
      record=x.split(",")
      record[4]=float(record[4])+(float(record[4])*.05)
      return str(record)

products=sc.textFile("file:///home/hadoop/product.csv")
new_products=products.map(PriceHike)
new_products.foreach(print)
```

2. **filter(function)**

```
numbers=sc.parallelize(range(1,101))
f1 = numbers.filter(lambda x:x>50)
f1.collect()
```

```
products=sc.textFile("file:///home/hadoop/product.csv")
products=products.map(lambda x : x.split(","))
f1=products.filter(lambda x : float(x[4])>50000)
f1.foreach(print)
```

3. **reduce(function)**

```
numbers=sc.parallelize(range(1,11))
result=numbers.reduce(lambda x,y:x+y)
print(result)                                          #55
```

4. **foreach(function)**

```
numbers=sc.parallelize(range(1,11))
numbers.foreach(print)
```

5. **first(function)**

```
numbers=sc.parallelize(range(1,11))
numbers.first()
```

6. **take(n)**

```
numbers=sc.parallelize(range(1,11))
numbers.take(5)
```

7. **collect()**

```
numbers=sc.parallelize(range(1,11))
numbers.collect()
```

8. **count()**

```
numbers=sc.parallelize(range(1,11))
numbers.count()
```