Map Reduce Advanced - Relational Join

Mappers and Reducers

Here's a quick but comprehensive introduction to the idea of splitting tasks into a MapReduce model. The four important functions involved are:

```
Map (the mapper function)
EmitIntermediate(the intermediate key,value pairs emitted by the mapper functions)
Reduce (the reducer function)
Emit (the final output, after summarization from the Reduce functions)
```

We provide you with a single system, single thread version of a basic MapReduce implementation.

Task

A SQL join combines records from two or more tables in a relational database. It creates a set that can be saved as a table or used as it is. A JOIN is a means for combining fields from two tables (or more) by using values common to each.

The input is a number of lines with records from two tables *Employee* and *Department*. A tuple from *Employee* table will look like:

```
Employee [Person_Name] [SSN]
```

A tuple from *Department* table will look like:

```
Department [SSN] [Department_Name]
```

The required output is to print the JOIN of the two tables *Employee* and *Department*, in the format shown. The code for the MapReduce class, parts related to IO etc. has already been provided. However, the mapper and reducer functions are incomplete. Your task is to fill up the mapper and reducer functions appropriately, such that the program works, and outputs the JOIN of the two tables, in lexicographical order.

Also, this program outputs certain information to the error stream. This information has been logged to help beginners gain a better understanding of the the intermediate steps in a map-reduce process.

Languages Supported

Currently, we provide the base code in Python.

Input Format

A list of comma separated records from the tables *Employee* and *Department*. We have already written the input handling code to read in this data.

Output Format

Again, the output handling part has already been provided in the template code. The output contains the

JOINed records in the following format arranged lexicographically -

```
([SSN] [Employee_Name] [Department_Name])
```

Sample Input

```
Department,1234,Sales
Employee,Susan,1234
Department,1233,Marketing
Employee,Joe,1233
Department,1233,Accounts
```

Sample Output

```
('1233', 'Joe', 'Accounts')
('1233', 'Joe', 'Marketing')
('1234', 'Susan', 'Sales')
```

Explanation

We have computed the JOIN of two tables *Employee* and *Department* via the Mapper and Reducer functions.