Loading Data with Static Partitioning

One way to load data into a partitioned table is to use *static partitioning,* in which you manually define the different partitions.(The other way is to have the partitions automatically defined when you load the data. See “Loading Data with Dynamic Partitioning” for this method.)

With static partitioning, you create a partition manually, using an ALTER TABLE … ADD PARTITION statement, and then load the data into the partition.

For example, this ALTER TABLE statement creates the partition for Pakistan (pk):

```
ALTER TABLE customers_by_country

    ADD PARTITION (country='pk');
```

Notice how the partition column name, which is country, and the specific value that defines this partition, which is pk, are both specified in the ADD PARTITION clause. This creates a partition directory named country=pk inside the customers_by_country table directory.

After the partition for Pakistan is created, you can add data into the partition using an INSERT … SELECT statement:

```
INSERT OVERWRITE TABLE customers_by_country

    PARTITION(country='pk')

    SELECT cust_id, name FROM customers WHERE country='pk'
```

Notice how in the PARTITION clause, the partition column name, which is country, and the specific value, which is pk, are both specified, just like in the ADD PARTITION command used to create the partition. Also notice that in the SELECT statement, the

partition column is not included in the SELECT list. Finally, notice that the WHERE clause in the SELECT statement selects only customers from Pakistan.

With static partitioning, you need to repeat these two steps for each partition: first create the partition, then add data. You can actually use any method to load the data; you need not use an INSERT statement. You could instead use hdfs dfs commands or a LOAD DATA INPATH command. But however you load the data, it's your responsibility to ensure that data is stored in the correct partition subdirectories. For example, data for customers in Pakistan must be stored in the Pakistan partition subdirectory, and data for customers in other countries must be stored in those countries' partition subdirectories.

Static partitioning is most useful when the data being loaded into the table is already divided into files based on the partition column, or when the data grows in a manner that coincides with the partition column: For example, suppose your company opens a new store in a different country, like New Zealand ('nz'), and you're given a file of data for new customers, all from that country. You could easily add a new partition and load that file into it.

Try It!

If you did not create the customers_by_country table in the "Creating Partitioned Tables" reading, do so before continuing. If you did the exercises in the "Loading Data with Dynamic Partitioning" reading before doing this one, drop the table and recreate it (without loading the data). (Hint: In Hue, you probably don't have to retype the command, it should be in your Query History. Find it and click on it.)

1. First, look at the contents of the customers_by_country table directory in HDFS. (Where this directory exists depends on which database holds the table.) You can use Hue or an hdfs dfs -ls command to list the contents of the directory. Since you haven't loaded any data, it should be empty.

2. Add the partition for Pakistan (pk) using the ALTER TABLE command:

```
ALTER TABLE customers_by_country
```

ADD PARTITION (country='pk');

3. Check the contents of the customers_by_countrytable directory in HDFS and see that it now has a subdirectory for the partition you just created.

4. Now load *only* the customers from Pakistan into that partition:

```
INSERT OVERWRITE TABLE customers_by_country

PARTITION(country='pk')

SELECT cust_id, name FROM default.customers WHERE country='pk';
```

5. *Optional:* Modify and run both commands to create a partition for one of the other countries (us, ja, or ug) and load the data from the customers table into that partition. You can do it for all three if you like. Check that the customers_by_country directory has one subdirectory for each partition you added.

6. Look at the file in one (or more, if you like) of those directories, using Hue or an hdfs dfs -cat command. Notice that the file contains the row for the customer from that country, and no others; notice also that the country value is not included (because you didn't include it in the SELECT list).

7. Run some SELECT queries on the partitioned table. Try one that does no filtering (like SELECT * FROM customers_by_country;) and one that filters on country. It's a small table so there won't be a significant difference in the time it takes to run; the point is just to notice that you will not query the table any differently than you would query the customers table.