```
slab[][%Start] = 0          // Start at zero for all dimensions
slab[][%Stride] = 1         // Use a stride of 1 for all dimensions
slab[][%Count] = 1          // Use a count of 1 for all dimensions
slab[0][%Block] = 6         // Set block size for the dimension 0 to 6
slab[1][%Block] = 5         // Set block size for the dimension 1 to 5
TestWave = -1               // So we can see the next command change it
HDF5LoadData /N=TestWave /O /SLAB=slab fileID, "TestArray"
```

Here we set the stride, count and block parameters to load every other element from both dimensions.

```
slab[][%Start] = 0          // Start at zero for all dimensions
slab[][%Stride] = 2         // Use a stride of 2 for all dimensions
slab[0][%Count] = 3         // Load three blocks of dimension 0
slab[1][%Count] = 2         // Load two blocks of dimension 1
slab[0][%Block] = 1         // Set block size for dimension 0 to 1
slab[1][%Block] = 1         // Set block size for dimension 1 to 1
TestWave = -1               // So we can see the next command change it
HDF5LoadData /N=TestWave /O /SLAB=slab fileID, "TestArray"
```

Finally we close the file.

```
HDF5CloseFile fileID
```

Each row of the slab wave holds a set of parameters (start, stride, count and block) for the corresponding dimension of the dataset in the file. Row 0 of the slab wave holds the parameters for dimension 0 of the data-set, and so on.

The start values must be greater than or equal to zero. All of the other values must be greater than or equal to 1. All values must be less than 2 billion.

HDF5LoadData clips the values supplied for the block sizes to the corresponding sizes of dataset being loaded.

HDF5LoadData requires that the slab wave have exactly four columns. The HDF5MakeHyperslabWave function, from the automatically-loaded "HDF5 Utiliies.ipf" procedure file, creates a four-column wave with the column dimension labels Start, Stride, Count, and Block. HDF5LoadData does not require the column dimension labels. As of Igor Pro 9.00, HDF5MakeHyperslabWave returns a free wave if you pass "" for the path parameter. It also returns a wave reference as the function result whether the wave is free or not.

The slab wave must have at least as many rows as the dataset has dimensions. Extra rows are ignored.

HDF5LoadData creates a wave just big enough to hold all of the loaded data. So in the first example, it created a 6 x 5 wave whereas in the last example it created a 3 x 2 wave.

Igor Versus HDF5

This section documents differences in how Igor and HDF5 organize data and how Igor reconciles them.

Case Sensitivity

Igor is not case-sensitive but HDF5 is. So, for example, when you specify the name of an HDF5 data set, case matters. "/Dataset1" is not the same as "/dataset1".

If you load /Dataset1 and /dataset1 into Igor using the default wave name, the second load overwrites the wave created by the first.

HDF5 Object Names Versus Igor Object Names

The forward slash character is not allowed in HDF5 object names. If you create an Igor wave or data folder with a name containing a forward slash and attempt to save the object to an HDF5 file, you will get an error. An object name that starts with a dot may also create an error.