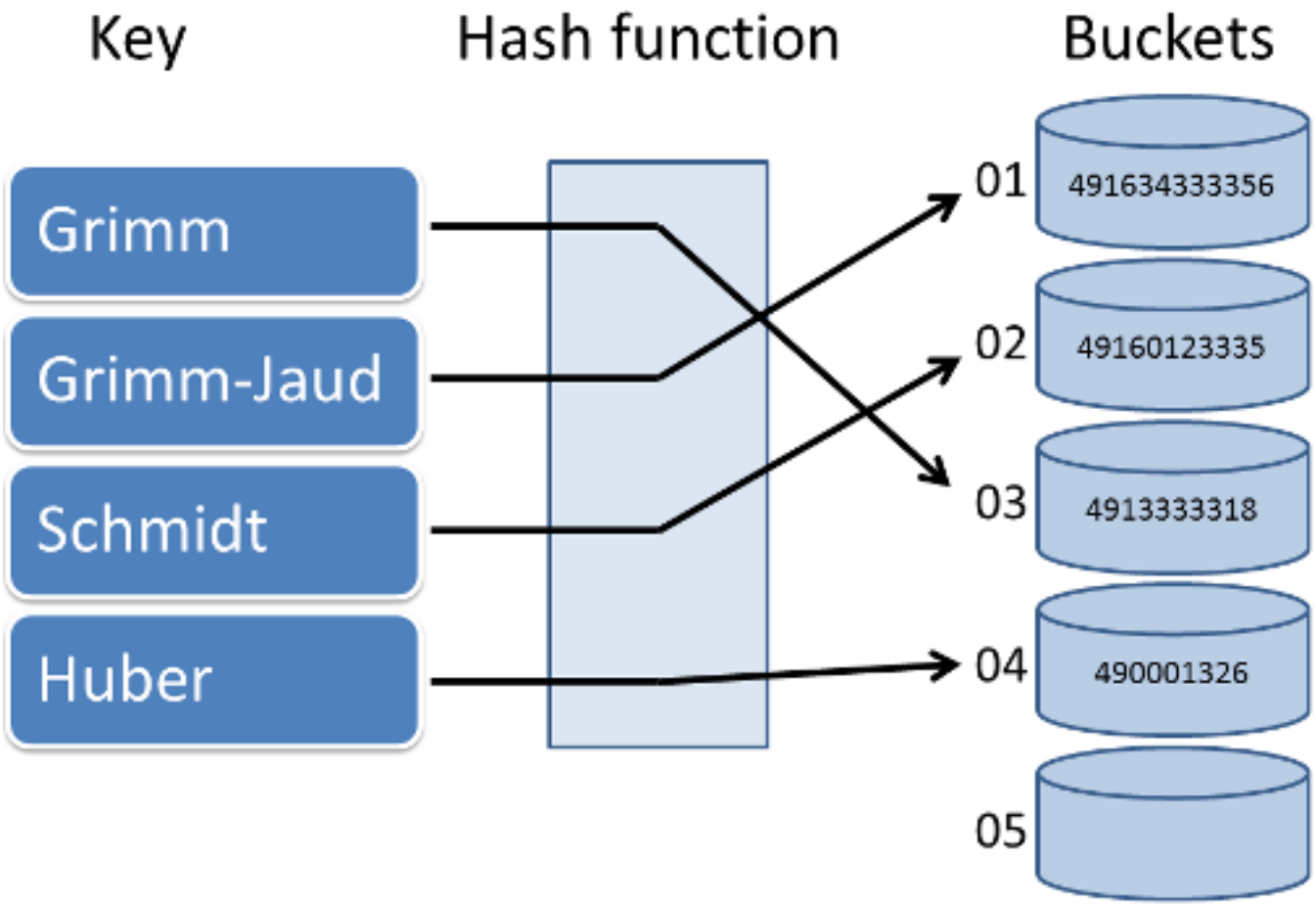
Overview

The main difference between unordered and ordered associative containers is the idea of sorted keys. Let's find out how unordered containers handle keys.



With the new C++11 standard, C++ has four unordered associative containers: `std::unordered_map`, `std::unordered_multimap`, `std::unordered_set`, and `std::unordered_multiset`. They have a lot in common with their namesakes, the ordered associative containers. The difference is that the unordered ones have a richer interface and their keys are not sorted.

This shows the declaration of an `std::unordered_map`.

```
template< class key, class val, class Hash= std::hash<key>,
          class KeyEqual= std::equal_to<key>,
          class Alloc= std::allocator<std::pair<const key, val>>>
class unordered_map;
```

Like `std::map`, `std::unordered_map` has an allocator, but `std::unordered_map` needs no comparison function. Instead `std::unordered_map` needs two additional functions: One to determine the hash value of its key: `std::has<key>` and a second to compare the keys for equality: `std::equal_to<key>`. Because of the three default template parameters, we only have to provide the type of the key and the type of the value. For example, declaration of `std::unordered_map` would be `std::unordered_map<char,int> unordMap`.

In the next lesson, we'll discuss the properties of keys and values.