HASH TABLES: CONCEPTS & PRACTICAL APPLICATIONS.

Swipe to know ⟫

www.alisherabbasi.com

“A HASH TABLE IS A DATA STRUCTURE THAT STORES KEY-VALUE PAIRS. IT OFFERS EFFICIENT INSERTION, DELETION, AND RETRIEVAL OPERATIONS.”

HASH FUNCTIONS

- The core of a hash table is the **hash function**.

- **“Hash Function is simply a function that generates a value of fixed length for each input that it gets.”**

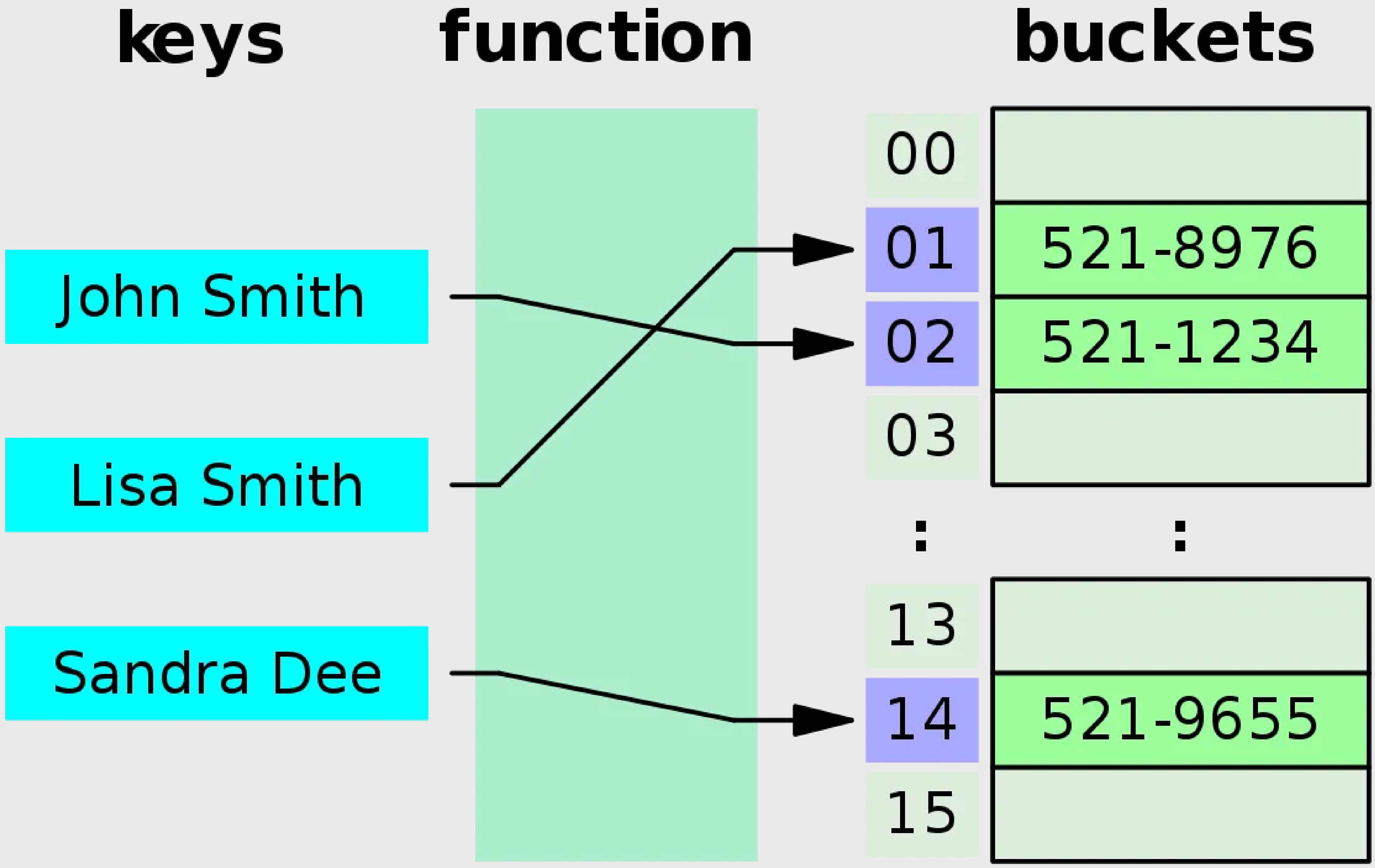
- **Example:**

```
    _hash(input) {
        return key.length % 10;
/*this is an example,  you can use any logic for
                generating hash value. */
     }
```

- The hash function's goal is to evenly distribute keys across the available **buckets** in the table.

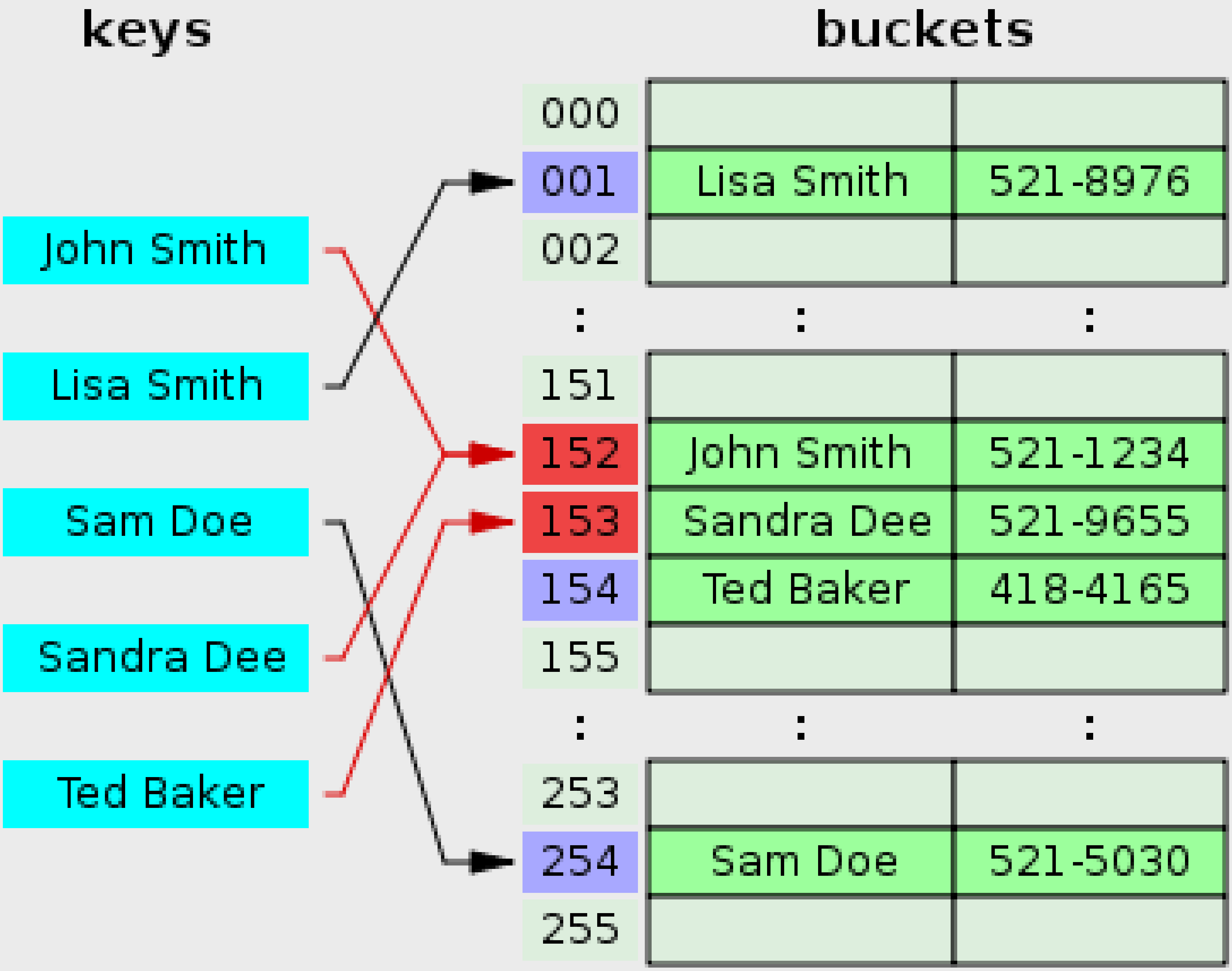
BUCKETS

“Hash tables typically consist of an array of "**buckets**" or "**slots**". Each bucket can hold multiple key-value pairs or sometimes just one.”



COLLISION

"Collisions occur when multiple keys hash to the same bucket."



keys

- John Smith
- Lisa Smith
- Sam Doe
- Sandra Dee
- Ted Baker

buckets

000		
001	Lisa Smith	521-8976
002		
:	:	:
151		
152	John Smith	521-1234
153	Sandra Dee	521-9655
154	Ted Baker	418-4165
155		
:	:	:
253		
254	Sam Doe	521-5030
255		

COLLISION HANDLING

There are various strategies to handle collisions:

- **Chaining:**

Each bucket contains a linked list or array of key-value pairs that hash to the same index.

- **Open Addressing**

When a collision happens, the algorithm probes for alternative slots until an empty one is found.

EFFICIENCY (TIME & SPACE COMPLEXITY)

Hash tables offer O(1) constant-time performance for **insertion**, **deletion**, and **retrieval** operations.

However, their performance can **degrade** under certain circumstances, such as a poor hash function causing many **collisions**.

PRACTICAL APPLICATIONS

1. **Databases.**
2. **Hash-based Encryption.**
3. **Spell Checkers and Dictionaries.**
4. **File Systems.**
5. **Network Routing Tables.**

Thank you!