Notes - Session 11 - Bloom Filters

What is Bloom Filter?

Bloom filters are an efficient data structure that helps you **quickly** determine if an element is in a set or not. They are particularly useful in situations where the set is large, and you want to reduce the use of memory and increase speed. Here's an overview with examples and a visual representation:

Understanding Bloom Filters

Concept:

A Bloom filter is a probabilistic data structure. It tells you if an element is definitely not in the set or might be in the set.

How It Works:

- A Bloom filter uses an array of bits (all initially set to 0) and several hash functions.
- When you add an item to the filter, you pass it through all the hash functions. Each hash function maps the item to a position in the bit array, and the bits at these positions are set to 1.
- To check if an item is in the set, you pass it through the hash functions again. If all the bits at the mapped positions are 1, the item might be in the set. If any bit is 0, the item is definitely not in the set.

False Positives:

Bloom filters can tell you **with certainty if an item is not in the set**, but they can give false positives (saying an item is in the set when it isn't).

No False Negatives:

There are no false negatives; if a Bloom filter says an item is not in the set, it is not.

Examples in System Design

Network Applications:

Used to quickly check if a URL or an IP address is part of a blocklist.

Database Systems:

Before accessing disk storage, databases can use Bloom filters to check if a record exists, reducing unnecessary disk reads.

Cache Filtering:

To check if a particular data is already cached.

Recommendation Systems / Feed Generation System:

To ensure we don’t show the same content to a user twice,

Real-World Bloom Filters:

- Redis provides Bloom Filters as one of it’s features