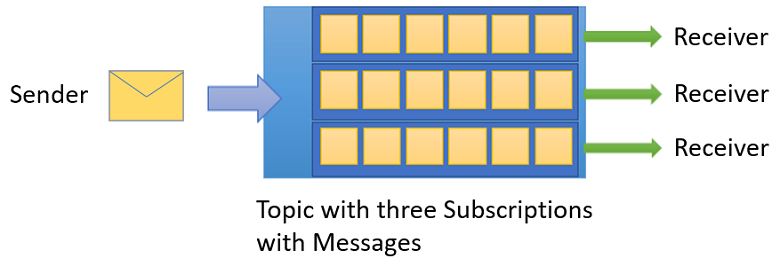
*Topics and subscriptions*

*A queue allows processing of a message by a single consumer. In contrast to queues, topics and subscriptions provide a one-to-many form of communication in a publish and subscribe pattern. It's useful for scaling to large numbers of recipients. Each published message is made available to each subscription registered with the topic. Publisher sends a message to a topic and one or more subscribers receive a copy of the message.*

**

*The subscriptions can use more filters to restrict the messages that they want to receive. Publishers send messages to a topic in the same way that they send messages to a queue. But, consumers don't receive messages directly from the topic. Instead, consumers receive messages from subscriptions of the topic. A topic subscription resembles a virtual queue that receives copies of the messages that are sent to the topic. Consumers receive messages from a subscription identically to the way they receive messages from a queue.*

*The message-sending functionality of a queue maps directly to a topic and its message-receiving functionality maps to a subscription. Among other things, this feature means that subscriptions support the same patterns described earlier in this section regarding queues: competing consumer, temporal decoupling, load leveling, and load balancing.*

***Choosing Between Queues and Topics*** *–*

* *Communication model: If your system requires point-to-point communication or task distribution, queues are the preferred choice. On the other hand, if you need to broadcast messages to multiple consumers or implement an event-driven architecture, topics are more suitable.*
* *Message ordering: If preserving the order of messages is critical, queues provide a natural solution, as they ensure that messages are processed in the order they were added. Topics, while offering filtering and broadcast capabilities, do not guarantee message ordering.*
* *Scalability and distribution: Topics excel in scenarios where you need to scale horizontally and distribute messages across multiple subscribers. Queues, on the other hand, are effective when load balancing tasks or processing messages in a single consumer.*