Interview Questions:

1. **How does Spark structured streaming work?**

   In Spark structured streaming, each incoming row is treated as a small set of rows of a table, and it keeps on appending rows to tables that fall in the same bucket. In this way, it creates a DataFrame on such tables, and Spark structured streaming supports all DataFrame operations, except window and join operations in some use cases.

2. **Why is there a need for Spark structured streaming when Spark already has DStreams?**

   Spark structured streaming works on the concept of micro-batches. Real streaming never existed in the case of DStreams.
   DStreams was built on Resilient Distributed Datasets (RDDs). So, many optimisations that were performed on DataFrames and Datasets were missing, and, consequently, memory optimisations were also missing since RDDs are a non-type data structure.
   The lateness of data was not handled properly and also, fault tolerance was not in place for driver programs.
   All of these disadvantages were eliminated in Spark structured streaming.

3. **What are triggers in Spark structured streaming?**

   Triggers handle the running of stream applications, for instance, running stream applications every 10 seconds or continuously, or only once. This enables Spark to execute jobs in micro-batches or as continuous streams.

4. **What are the different write output modes in Spark structured streaming, and why do we need them?**

   There are three write output modes in Spark spark structured streaming: append, update and complete.
   Append: It is the default write output mode in Spark spark structured streaming, and we use it when we don't want to handle late-arriving data. Each data row is processed individually and once processed, it is appended to the output stream.
   Update: This write output mode is used when we want to handle late-arriving data, and aggregation is changed when we consider late-arriving data. In this case, specific rows of the resulting table will be updated, and this case cannot be run in append mode.
   Complete: This write output mode preserves incoming data until it expires, and it is used in case of aggregations on incoming batches so that we can perform aggregations on an entire set of data.

5. **Why does Spark structured streaming not support all types of joins?**

   In Spark structured streaming, joins can be applied in only certain scenarios. A few such scenarios are given below:
   Stream with Stream:

i. If both the dfs are stream, then all joins, such as inner, left and right, are supported, since the resulting frame will be a stream. The only exception is that a full join will not be supported since both of the dfs are stream, and if one df is late, the other has to wait until its data arrives.

Static and Stream

ii. If one df is static and the other is a stream, then we can perform an inner join. However, left and right joins are supported only when the left table is a stream in case of a left join and the right table is a stream in case of a right join. The reason behind this is simple: if the right table is a stream and you perform a left join, then the result will be a static table only.

iii. A full join between static and stream is not supported for the same reason. The stream df may have late-arriving data and the join may have to be delayed, thus producing inconsistent results.

iv.

6. **What is the problem that is solved by watermarking in Spark structured streaming?**

   Watermarking in Spark structured streaming solves the problem of lateness, that is, data which is coming late as per its event time. Here, we can specify a time window up to which Spark can monitor/consider late-arriving data, so that it can be included along with other relevant data sets while processing using update write mode and a consistent resulting set can be sent.

7. **How is Spark structured streaming managed during streaming?**

   Like a Spark batch, Spark structured streaming also provides fault tolerance. For example, when an executor dies, Spark streaming creates another executor and shifts the tasks to the new executor.

   However, streaming has more complex problems like its not single file where it can read the data back if the partition is lost so, spark structure streaming checks that source stream is replayable that it must allow tracking the data via offsets eg. Kafka It also allows fault tolerance at the driver level, where it checkpoints the stream to hdfs directory and can start the stream again where it failed using the source replayable technique. (**Note:** Socket does not support the source replayable technique.)

8. **Can Spark structured streaming run a window operation if a DataFrame does not have a timestamp column? If yes, then how?**

   Spark window operations are not supported on non-timestamp-based columns and, so, it is not possible to run these operations on such columns.

   Also, if the source is Kafka and you don't have a timestamp column, then it can get an event timestamp when data arrives at the system, and that timestamp can be used for window operations, for instance, all data arriving within a window frame of 10 mins.

9. **Can Kafka read and write data to the same topic? If yes, then what would be the behaviour?**

There is no restriction on writing and reading data to the same Kafka topic, although this will lead to a state of deadlock, wherein it will not come out of processing and will continue to process the same data.

10. **Can we use the show() method on a streaming DataFrame? If yes, then how?**

    The show() method does not work in a DataFrame streaming way as it's not an actual batch DataFrame and streaming works in the background in the thread. However, if you use memory as a write stream, then the result set is kept in memory as a DataFrame and then you can apply the show() method on it.

11. **Can we remove duplicate data in a window of 5 mins?**

    Yes, Spark streaming supports the drop duplicate method and this method can be used to drop duplicate data in case of lateness as well. So, if the same record appears after intervals of, say, 1-2 mins, then we can use the drop duplicate method to drop that data.

12. **Can we count the elements arriving within a window of 5 mins in Spark structure streaming?**

    We cannot simply run the count method on a streaming DataFrame, since it is not supported. However, we can apply this method after a groupBy operation. So, to get the count of the number of rows in a batch, we can apply the group by operation on the window start time and count the number of rows in the result.