Explain issues in Data stream query processing?

1. **Unbounded Memory Requirements:**

   Since data streams are potentially unbounded in size, the amount of storage required to compute an exact answer to a data stream query may also grow without bound. While external memory algorithms for handling data sets larger than main memory have been studied, such algorithms are not well suited to data stream applications since they do not support continuous queries and are typically too slow for real-time response. New data is constantly arriving even as the old data is being processed; the amount of computation time per data element must be low, or else the latency of the computation will be too high and the algorithm will not be able to keep pace with the data stream.

2. **Approximate Query Answering:**

   When we are limited to a bounded amount of memory it is not always possible to produce exact answers for data stream queries; however, high-quality approximate answers are often acceptable in lieu of exact answers. Sliding Window: One technique for producing an approximate answer to a data stream query is to evaluate the query not over the entire past history of the data streams, but rather only over sliding windows of recent data from the streams. For example, only data from the last week could be considered in producing query answers, with data older than one week being discarded.

3. **Blocking Operators:**

   A blocking query operator is a query operator that is unable to produce the first tuple of its output until it has seen its entire input. If one thinks about evaluating continuous stream queries using a traditional tree of query operators, where data streams enter at the leaves and final query answers are produced at the root, then the incorporation of blocking operators into the query tree poses problems. Since continuous data streams may be infinite, a blocking operator that has a data stream as one of its inputs will never see its entire input, and therefore it will never be able to produce any output. Doing away with blocking operators altogether would be problematic, but dealing with them effectively is one of the more challenging aspects of data stream computation.

4. **Queries Referencing Past Data:**

   In the data stream model of computation, once a data element has been streamed by, it cannot be revisited. This limitation means that ad hoc queries that are issued after some data has already been discarded may be impossible to answer accurately. One simple solution to this problem is to stipulate that ad hoc queries are only allowed to reference future data: they are evaluated as though the data streams began at the point when the query was issued, and any past stream elements are ignored (for the purposes of that query). While this solution may not appear very satisfying, it may turn out to be perfectly acceptable for many applications.

5. **Batch Processing sampling and synopses:** We avoid looking at every data element and restrict query evaluation to some sort of sampling or batch processing technique. In batch processing, rather than producing a continually up to date answer, the data elements are buffered as they arrive and the answer to the query is computed periodically. This is an approximate answer at point in the recent past rather than exact answer at the present moment.

   For some classes of data stream queries where no exact data structure with the desired properties exist, one can often design an approximate data structure that maintains a small synopsis or sketch of the data. (computation per data element will be minimum.)

6. **Sliding Windows:** One technique for approximate query answering is to evaluate the query not over the entire past history of the data streams, but rather only over sliding windows of the recent data from the streams. For example,
   Only last week could be considered in producing query answers, with data older than 1 week being discarded. Recent data, in the real world applications is more important and relevant than the old data.

Form of queries

- **One time queries/Ad-Hoc queries:** These are evaluated once over a point of time snapshot of dataset, with the answer returned to the user.
- Ex. Stock price checker may alert the user when a stock price crosses a particular price point.
- **Continuous Queries/Standing queries:** These are evaluated continuously as data streams continue to arrive. The answer to a continuous query is produced over time, always reflecting the stream seen so far. It's answers may be stored and updated as new data arrives, or they may produced as data streams themselves. Typically aggregation queries, such as finding maximum, average, count etc. ex. Maximum price of a particular stock every hour.
- **Predefined queries** are one that are supplied to the DSMS before arriving any relevant data.

  Predefined queries are most commonly continuous queries.
- **Ad-hoc queries:** are basically questions asked once about the current state of streams.

  If we do not store all streams in their entirety, it is difficult to answer arbitrary queries about streams. If we have some idea what kind of queries will be asked then we can prepare for them by storing appropriate parts or summaries of streams. These queries complicate the design of DSMS because they are not known in advance..Ex. What is the maximum value seen so far in a stream?

  Standing Queries: Queries that are in principle, asked about the stream at all times. Ex. Report each new maximum value ever seen in stream.