In this work, we present efficient algorithms for constructing sparse suffix trees, sparse suffix arrays, and sparse position heaps for b arbitrary positions of a text T of length n while using only O(b) words of space during the construction. Attempts at breaking the naïve bound of Ω(nb) time for constructing sparse suffix trees in O(b) space can be traced back to the origins of string indexing in 1968. First results were not obtained until 1996, but only for the case in which the b suffixes were evenly spaced in T. In this article, there is no constraint on the locations of the suffixes. Our main contribution is to show that the sparse suffix tree (and array) can be constructed in O(n log 2b) time. To achieve this, we develop a technique that allows one to efficiently answer b longest common prefix queries on suffixes of T, using only O(b) space. We expect that this technique will prove useful in many other applications in which space usage is a concern. Our first solution is Monte Carlo, and outputs the correct tree with high probability. We then give a Las Vegas algorithm, which also uses O(b) space and runs in the same time bounds with high probability when b = O(√n). Additional trade-offs between space usage and construction time for the Monte Carlo algorithm are given. Finally, we show that, at the expense of slower pattern queries, it is possible to construct sparse position heaps in O(n + blog b) time and O(b) space.