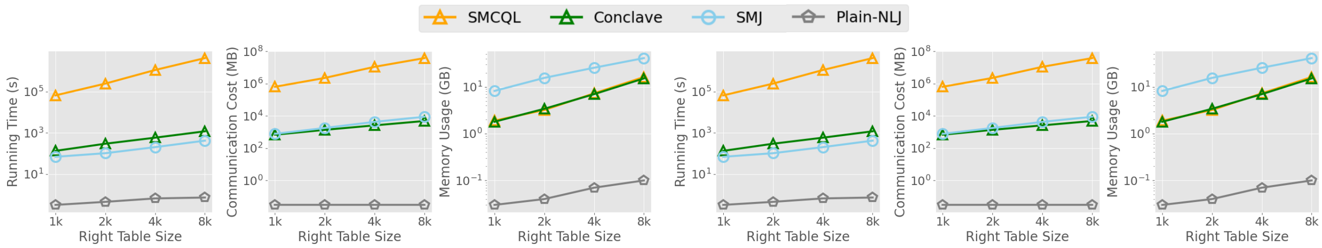
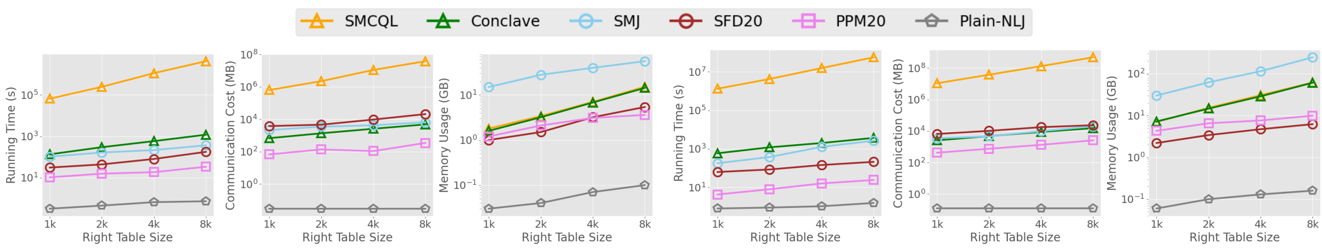


(a) Results when the left table size is 1k

(b) Results when the left table size is 4k

Fig. 12: Evaluation on varying input table sizes of federated theta joins on the TPC-H dataset.



(a) Results when the left table size is 1k

(b) Results when the left table size is 4k

Fig. 13: Evaluation on varying input table sizes of FK-FK federated binary equi-joins on the TPC-H dataset.

Appendix A
Evaluation on Federated Theta Joins on the TPC-H Dataset

We have evaluated the extensions of SMCQL, Conclave, and SMJ to federated theta-joins on the TPC-H dataset. Specifically, we have conducted the experiment of varying the table size when answering a federated theta-join. As shown in Fig. 12, the running time of SMJ is always shorter than SMCQL and Conclave. The communication cost of SMJ is closed to that of Conclave, but lower than that of SMCQL. As for the memory usage, SMCQL and Conclave are more efficient than SMJ. Overall, this experimental pattern is similar to the pattern of federated binary equi-joins when varying the input table sizes.

Appendix B
Evaluation on Varying Input Table Sizes of FK-FK Federated Binary Equi-Joins on the TPC-H Dataset

Fig. 13 presents the experimental results of foreign-key-to-foreign-key federated binary equi-joins on the TPC-H dataset when varying the input table sizes. Fig. 13a illustrates the experimental results when the left table has one thousand tuples and the right table size increases from one thousand to eight thousand. We can observe that the running time, communication cost, and memory usage of all the algorithms increase as the right table size raises. There is a similar pattern in Fig. 13b, where the left table size has four thousand tuples. In terms of the running time and communication cost, PPM20 is always more efficient than the others, and SMCQL is always the least efficient. The sort-merge based method, SFD20, is always the runner-up in terms of the running time. As for the memory usage, either the hash based method PPM20 or the sort-merge based method SFD20 is the most efficient. Moreover, we can also observe the time, communication, and memory cost of all the algorithms get higher when the left table size varies from one thousand to four thousand. This is reasonable since the input size gets larger and all the cost is closely related to the input size. The overall experimental pattern is similar to the pattern in Fig. 6.

Appendix C
Evaluation on Varying Join Selectivity of PK-PK Federated Binary Equi-Joins on the TPC-H Dataset

Fig. 14 presents the experimental result of primary-key-to-primary-key (PK-PK) federated binary equi-joins on the TPC-H dataset when varying the join selectivity. Since this is a PK-PK equi-join, the join selectivity is no higher than $\frac{1}{n}$, where n is the relation size. In other words, the join selectivity is relatively low. During the experiments, the PSI based method VOLE-PSI is always the most efficient, followed by the other PSI based method SecYan. The hash based method PPM20 is more efficient than the sort-merge based methods (SMJ and SFD20) and nested-loop based methods (Conclave and SMCQL). Moreover, based on the detailed results, we can observe that the running time, communication cost, and memory usage of all the algorithms get higher when the join selectivity increases. The result also demonstrates the join selectivity has less notable impact on the PK-PK federated binary equi-joins than PK-FK and FK-FK federated binary equi-joins. Therefore, we did not present this figure in the main manuscript.

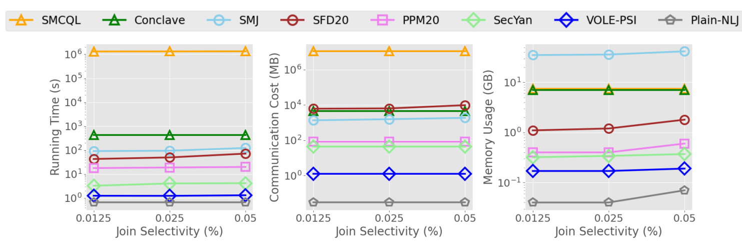


Fig. 14: Evaluation on varying join selectivity of PK-PK federated binary equi-joins on the TPC-H dataset.