1. What is Hashjoin?

Join

First, lets talk about join in SQL. The SQL join statement or command is frequently utilized to retrieve data from multiple tables by combining rows of data based on a shared column (field). Various types of joins, including inner join, outer join, left join, right join, semi join, and anti join, are commonly used in SQL.

Hash Join

The term "Hash join" is derived from the use of a hash function. It is a valuable technique for handling medium to large inputs but may not be efficient for very small datasets. Hash join is particularly effective for equi joins (=), and it can be used for various types of joins such as left, right, semi, and anti joins. Unlike other physical operators, Hash join requires memory for its operation. It has two distinct phases:

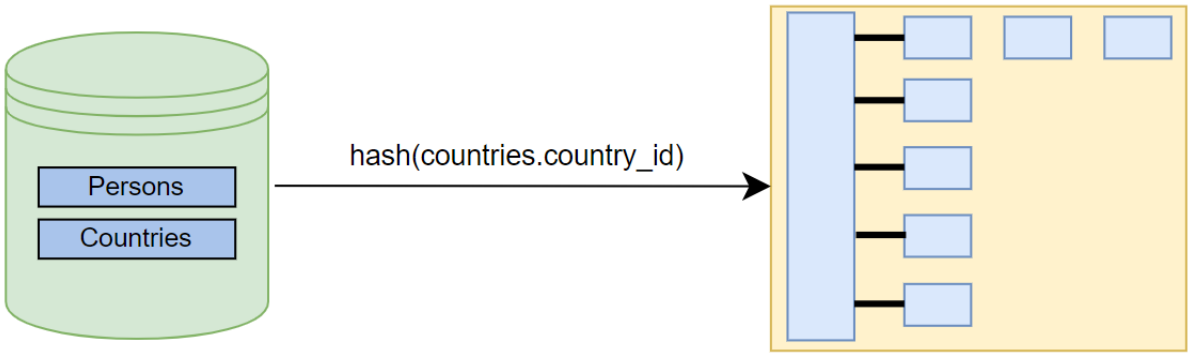
- The building or blocking phase.
- The probe or non-blocking phase.

How it works

Hash join is a popular method for combining two tables based on matching values. It utilizes a hash table to efficiently search for matches, making it faster than other options like nested loop join. While there are multiple types of joins available, hash join stands out for its performance in finding matches between tables.

1. Build Phase

The first step is for the server to create a hash table in its memory. The hash table is used to store rows from the input using a join operation, where the hash join attributes act as the keys. For example, let's say the input is a list of countries, and the hash join condition is based on the "country_id" attribute. This attribute will be used as the key in the hash table. Once all the rows are stored in the hash table, the build phase is finished.



2. Probe Phase

During the probe phase, the server retrieves rows from the probe input (represented as persons in our example). Each row is compared with the hash table using persons.country_id as the lookup key. When a match is found, the corresponding joined row is sent to the client. This process ensures that each input is scanned only once, with fast lookup times for finding matching rows between the two inputs.