The five types of join strategies are Broadcast Hash Join, Shuffle Hash Join, Sort Merge Join, Broadcast Nested Loop Join, and Cartesian Product Join. The summary of key details of each join strategy is as follow:

Broadcast Hash Join

- When to use:
  - Used when one dataset is smaller
  - Smaller dataset must be small enough to fit in memory
  - Very fast for skewed or small/medium joins
  - Avoids shuffling the larger dataset
- How it works:
  - Creates a hash table on the smaller dataset.
  - The hash table is broadcast to the partitions of the larger dataset.
  - The join operation happens locally on each node.
- Miscellaneous:
  - Supports "=" join condition only due to use of hash join
  - Supports all join types except FULL OUTER

Shuffle Hash Join

- When to use:
  - Used when one dataset is small enough to fit in memory
- How it works:
  - Both datasets are shuffled across partitions based on the join key
  - A hash table is built for one dataset. This hash table is probed for matching keys in the other dataset
  - Each partition performs a hash join locally
- Miscellaneous:
  - Supports "=" join condition only due to use of hash join
  - Supports all join types except FULL OUTER

Sort Merge Join

- When to use:
  - Used when datasets are large
  - Costly but efficient
- How it works:
  - Both datasets are partitioned and shuffled such that matching keys are on the same executor.
  - After shuffling, both datasets are sorted by the join key
  - After sorting, rows with matching keys are merged
- Miscellaneous:
  - Supports "=" join condition only

  - Supports all join types
  - Join key must be sortable. Does not build a hash table so there is no risk of running out of memory from building the hash table.

Broadcast Nested Loop Join

- When to use:
  - Used when one dataset is small, when hash or sort-merge joins can't be used (e.g. A non "=" join must be done).
  - Smaller dataset must be small enough to fit in memory
- How it works:
  - Broadcasts the smaller dataset to all partitions of the larger dataset.
  - Every row in the larger dataset is looped against every row in the broadcasted dataset using a nested loop
- Miscellaneous:
  - No join key needed
  - Supports "=" and non "=" join conditions
  - Supports all join types

Cartesian (or Cross) Join

- When to use:
  - Used when all combinations are needed
  - Should not use with large datasets
  - Slow and expensive
- How it works:
  - Every row from one dataset is combined with every row from the other dataset
- Miscellaneous:
  - No join key needed
  - Supports "=" and non "=" join conditions
  - Only supports Inner Join

When considering which strategy would be most suitable for the task, I considered that dataset B is much smaller (10,000 * 2) than dataset A (1,000,000 * 7), and that the join type used is "=", as we are joining on matching "geographical_location_oid". Dataset B is also small enough to fit on memory. With these conditions fulfilled, the most appropriate type of join to use is Broadcast Hash Join, which is the fastest and most efficient of the above methods.