# Join Methods

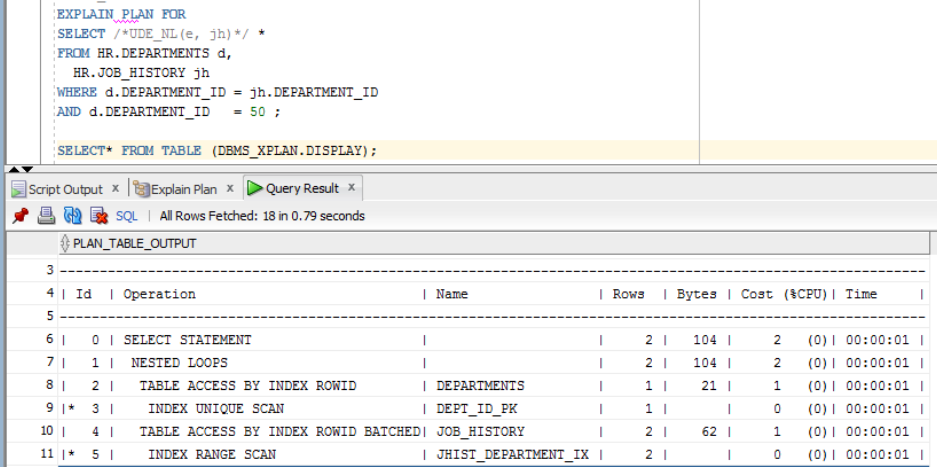
The main task is to try to get different types of different join methods. You can create your own needed objects, or use existing (in sample schemas). If you are creating tables or indexes, please, attach a script.

## Task 2: Nested Loop Join

Example:

The query shows people who worked in the department with id=50 early.

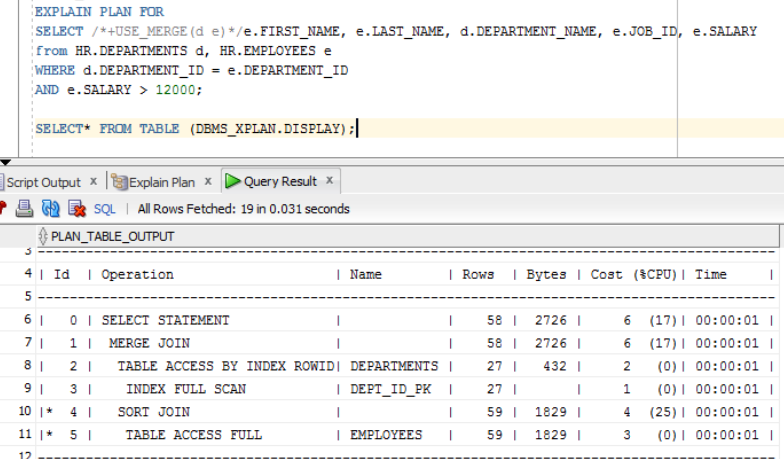
The database joins two small subsets of data that’s why Oracle used nested Loop Join.



## Task 3: Sort-Merge Join

The query shows company’s employees who have the salary more than 12000.

The database joins two tables where the condition is not an equijoin(>) that’s why Oracle used Sort-Merge Join.



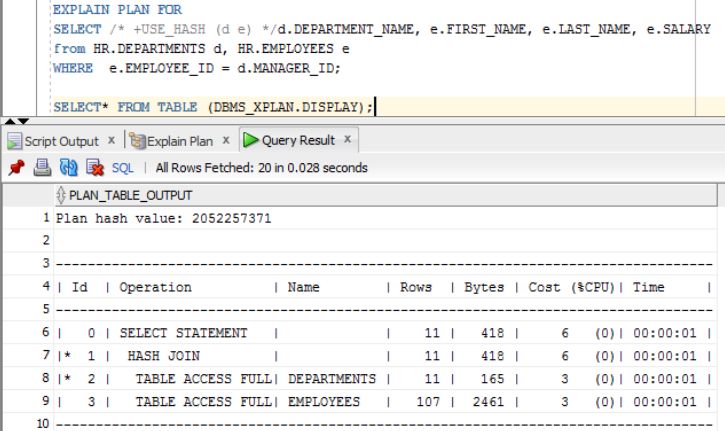
## Task 4: Hash Join

Using the hash join needs the following conditions:

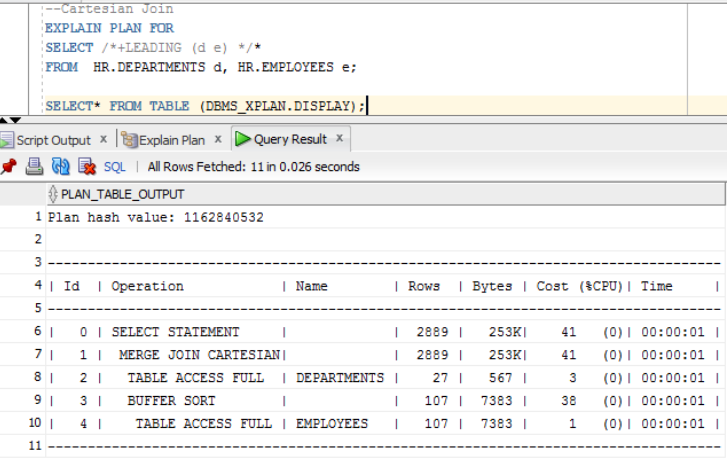
* A relatively **large amount of data** must be joined, or a large fraction of a small table must be joined.
* The join is an **equijoin**.

These two conditions is true, that’s why Oracle used hash join.

The query shows departments and their managers with salary.



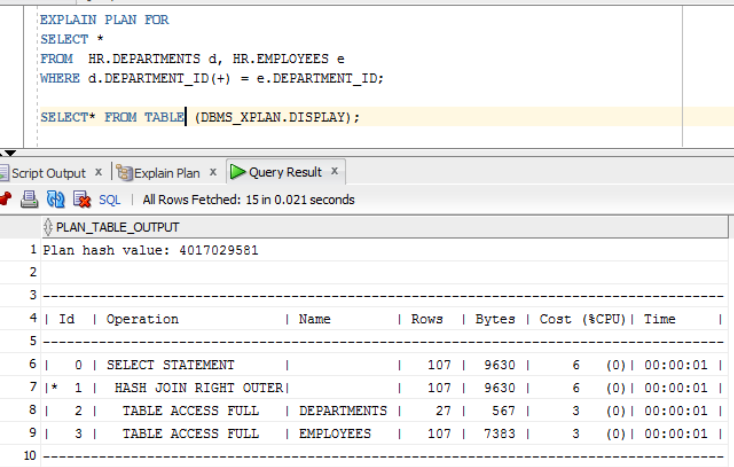
## Task 5: Cartesian Join

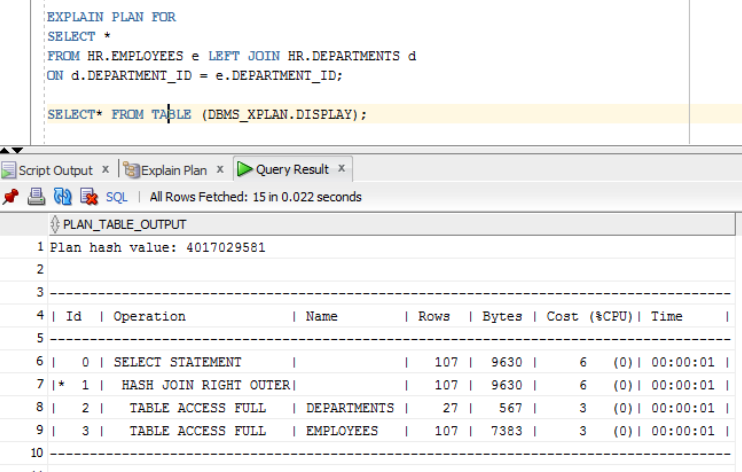


## Task 6: Left/Right Outer Joins

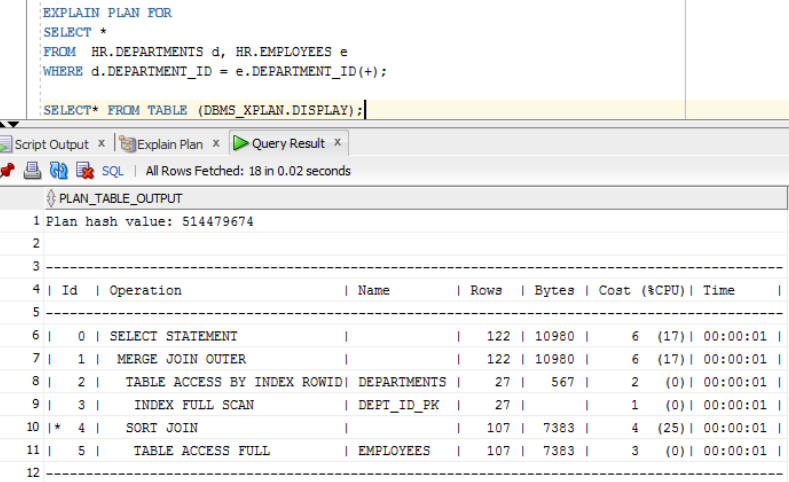
Implement Left/Right outer joins with:

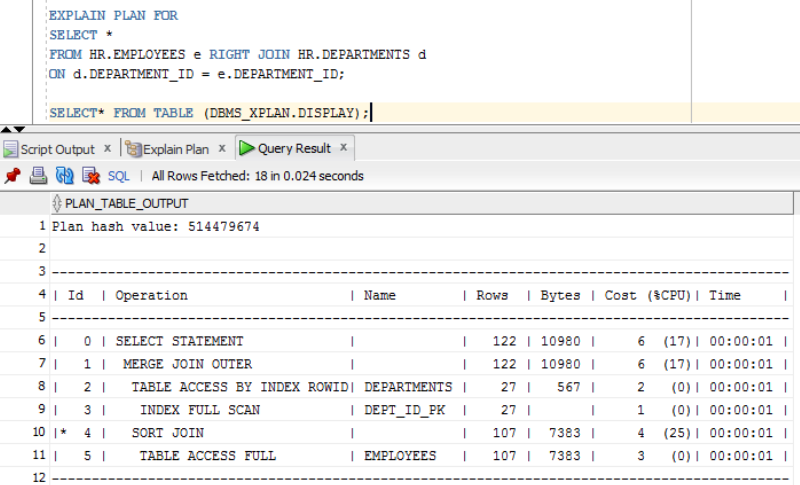
The left join returns all rows from the left table, even if there are no matches in the right table.





The right join returns all rows from the right table, even if there are no matches in the left table.

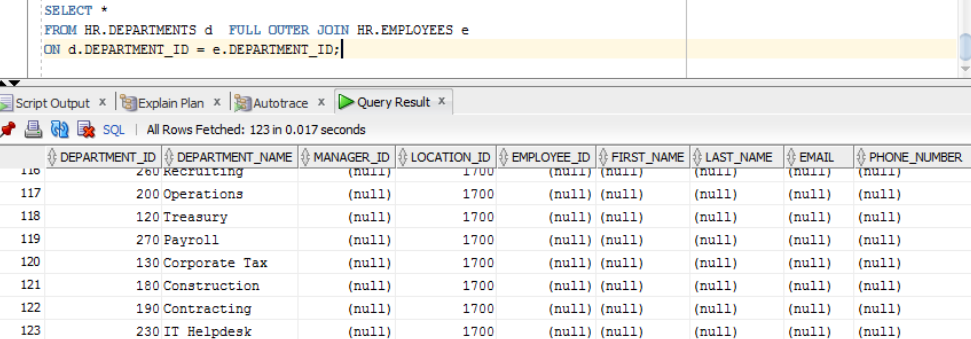


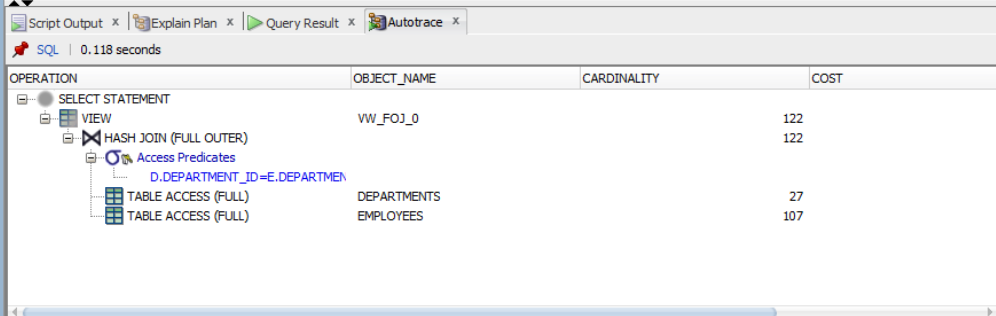


## Task 7: Full Outer Join

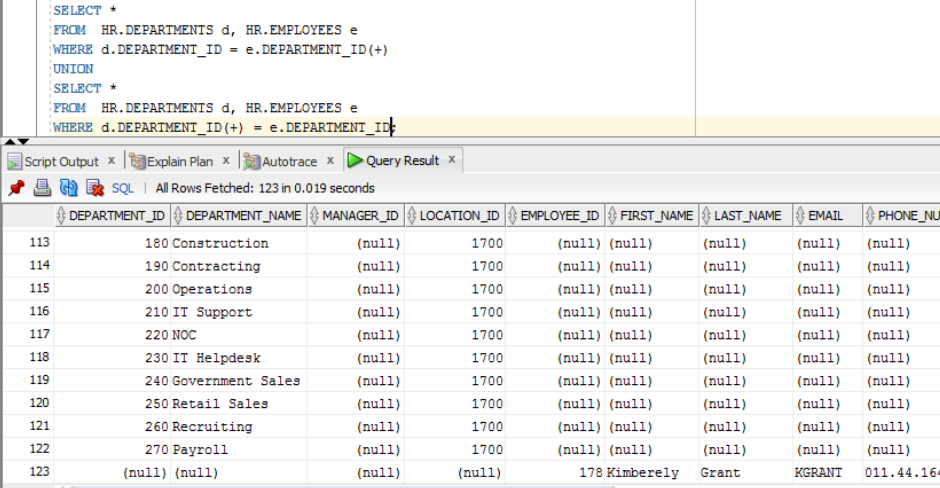
The full outer join will return all the rows from both tables that match plus the rows that are unique to each table.

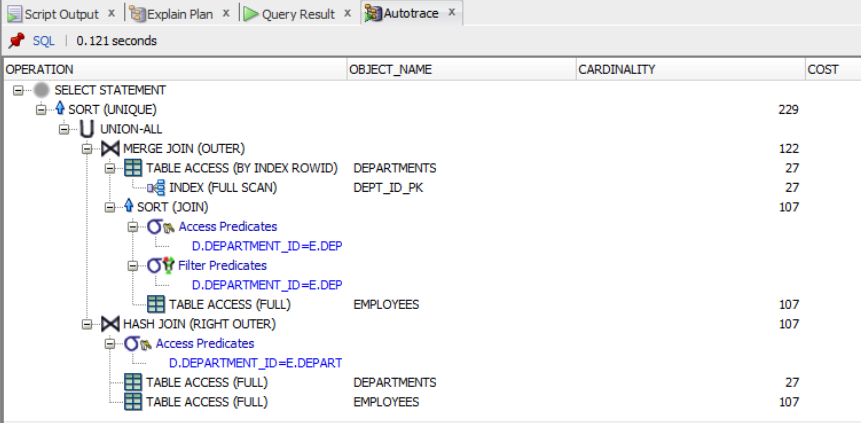
* ANSI syntax (outer join)





* Oracle syntax (+)





## Task 8: Results

|  |  |  |
| --- | --- | --- |
| Table “A” | Table “B” | Join type description |
| Small table with index  on join field | Small table with index on join field | MERGE JOIN was selected by optimizer because both tables have indexes. |
| Small table with index on join field | IOT | MERGE JOIN was selected by optimizer because one of the table index on join field and the second table is IOT table. |
| Small table with index  on join field | Small table w/o index on join field | NESTED LOOP was selected by optimizer because one of the table has index on join field |
| Small table w/o index  on join field | Small table w/o index on join field | MERGE JOIN was selected by optimizer because there was no indexes and no sort on both tables. |
| Medium table with index  on join field | Medium table with index on join field | MERGE JOIN was selected by optimizer because both tables have indexes. |
| Medium table w/o index  on join field | Medium table w/o index on join field | HASH JOIN was selected by optimizer because the cost is too high for merge join. |
| Medium table with index  on join field (w/o join condition) | Small table with index on join field (w/o join condition) | MERGE JOIN CARTESIAN was selected by optimizer because there was no join condition. |
| Big table with index  on join field | Small table w/o index on join field | NESTED LOOP was selected by optimizer because only one of the table has index on join field |
| Big table with index  on join field | Big table with index  on join field | HASH JOIN was selected by optimizer because the cost of sorting is too high and the tables are too big. |
| Big table with index  on join field | Big table w/o index  on join field | HASH JOIN was selected by optimizer because the cost of sorting is too high and the tables are too big. |

**Conclusion**

There are a lot of conditions that influence on the join methods used by Oracle. The join methods depend on the size of the combined data, on the size relative to each other, and on the parameters of the column connections.