Student name

College number

Name of department

Name of professor

Usually when working with databases and tables, there is a need to combine and compare table outputs. SQL joins are an effective method of deriving the implicit view of the data cutting across these tables and be able to draw meaningful information from them.

Natural joins in SQL are used to get the unique rows of records cutting across the tables in reference. The two tables in comparison must have at least one column that acts as a common column between the two tables. This base column is what is used to collect and align data across these tables. Below is the simple syntax and query for a normal NATURAL SQL JOIN

--Natural join syntax;

 SELECT \* FROM TABLE\_NAME\_1 NATURAL JOIN TABLE\_NAME\_2;

Left outer join on the other hand compares two tables. For instance TABLE\_A and another TABLE\_B and then looks for the matching condition, after which returns the result on table A. The first table defined in the left outer join will be the returning table when the query is finally executed. Below is the sample:

--LEFT OUTER JOIN

SELECT \* FROM TABLE\_NAME\_1 a

LEFT OUTER JOIN TABLE\_NAME\_2 b

ON a.TABLE\_NAME\_1.id = b.TABLE\_NAME\_2.id

WHERE --{pass condition here..}

Right outer joins do almost the same thing as the left outer join. The major deference between the two is that right outer join returns the values in the second table defined in the SQL query. Below is the sample syntax for this output:

--RIGHT OUTER JOIN

SELECT \* FROM TABLE\_NAME\_2 a

LEFT OUTER JOIN TABLE\_NAME\_1 b

ON a.TABLE\_NAME\_1.id = b.TABLE\_NAME\_2.id

WHERE --{pass condition here..}

Further, entity relationship diagrams are used to show the existing relationships between database tables. The relationship can also be compared to that of class object relations.

Additionally, the relationship can either be single (one to one), for instance one student belonging to one group and many to many, for instance a case where many classrooms are shared by many students. These are called cardinality relationships.

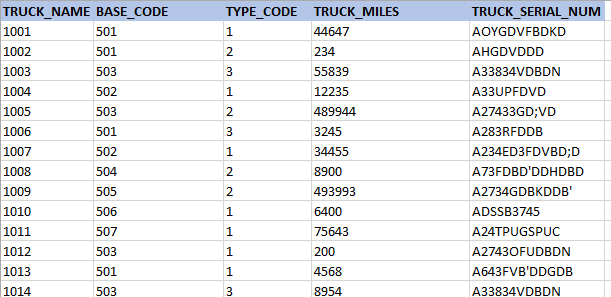
**Dataset descriptions:**

For the purpose of this assignment, there are two datasets that have been used to solve the above case of database analytical scenarios. The first database is a TRUCK\_TABLE that contains a total of 500 records of data. Similarly, the second table BASE\_TABLE is the location data of these trucks. The common column (foreign key) joining these two tables is the “BASE\_CODE”, found on both tables.

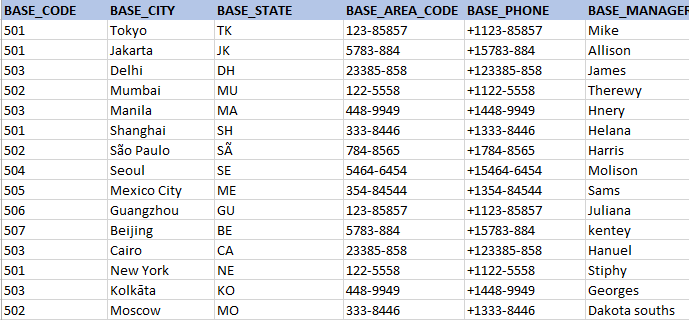
The analysis and queries of these two tables is done on an integrated database interpreter and the resulting outputs shown as below:

Datasets preview:

Truck table:



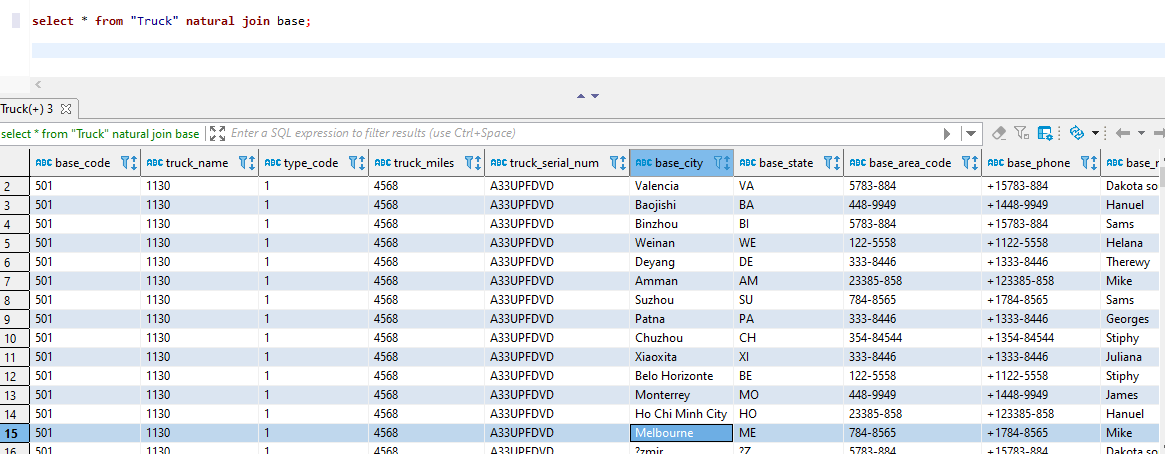
Base table:



**Solution one:** Natural Join

----Solution One

select \* from "Truck" natural join base;

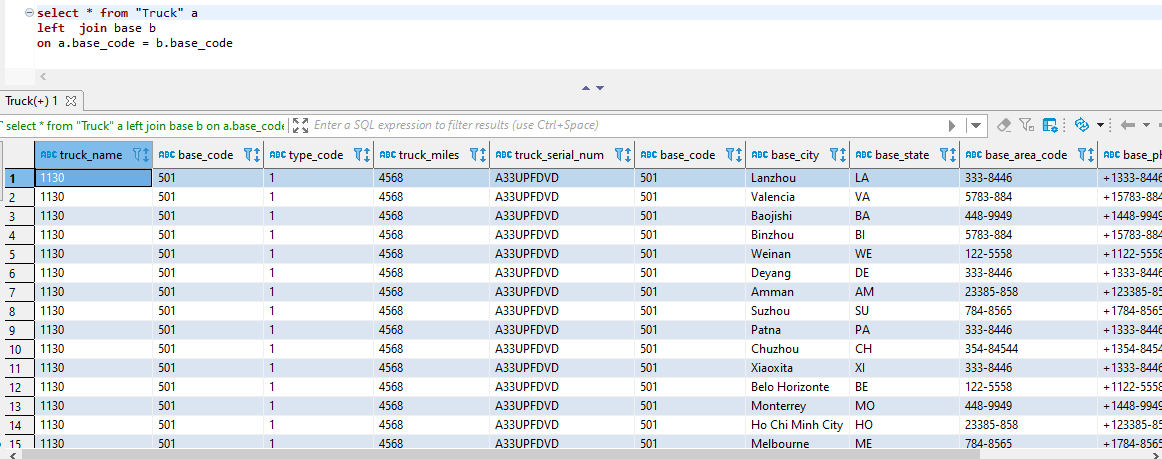


**Solution two:** Left outer Join

select \* from "Truck" a

left  join base b

on a.base\_code = b.base\_code

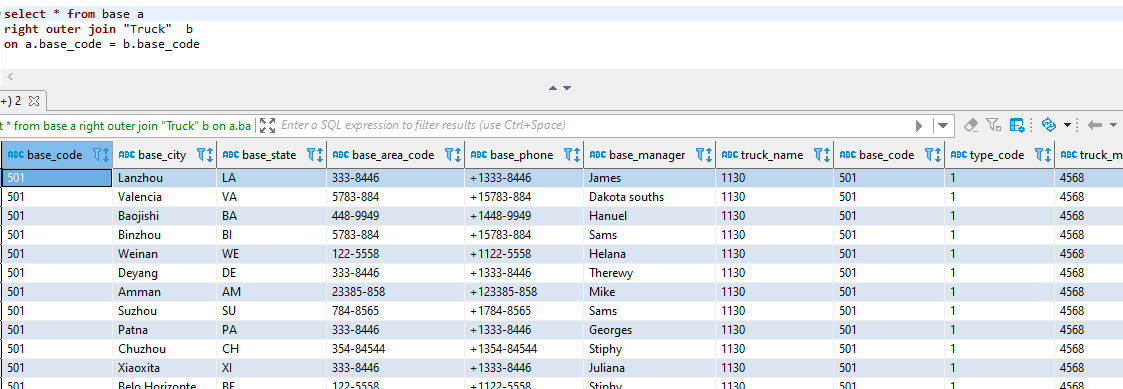


**Solution three:** Right outer join

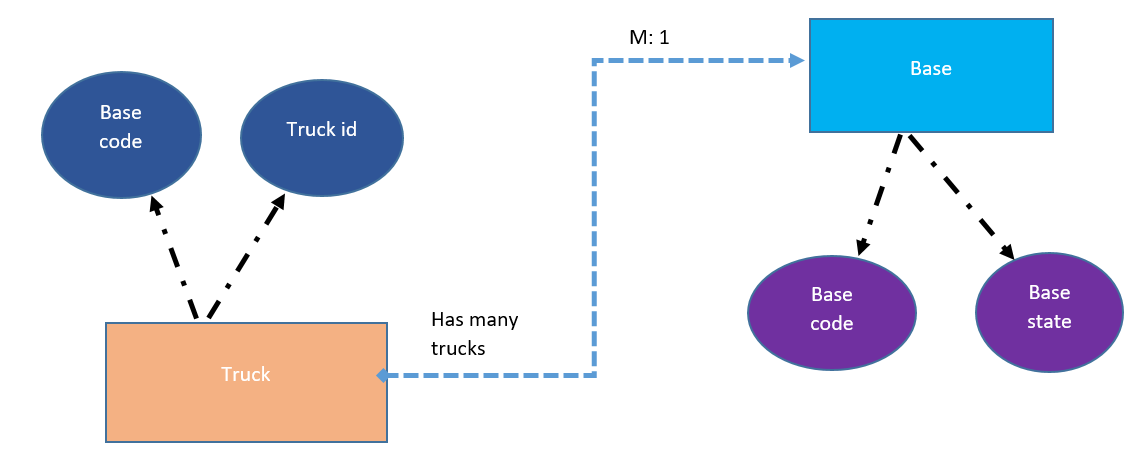
select \* from base a

right  join "Truck"  b

on a.base\_code = b.base\_code



**Chen diagram:**

****