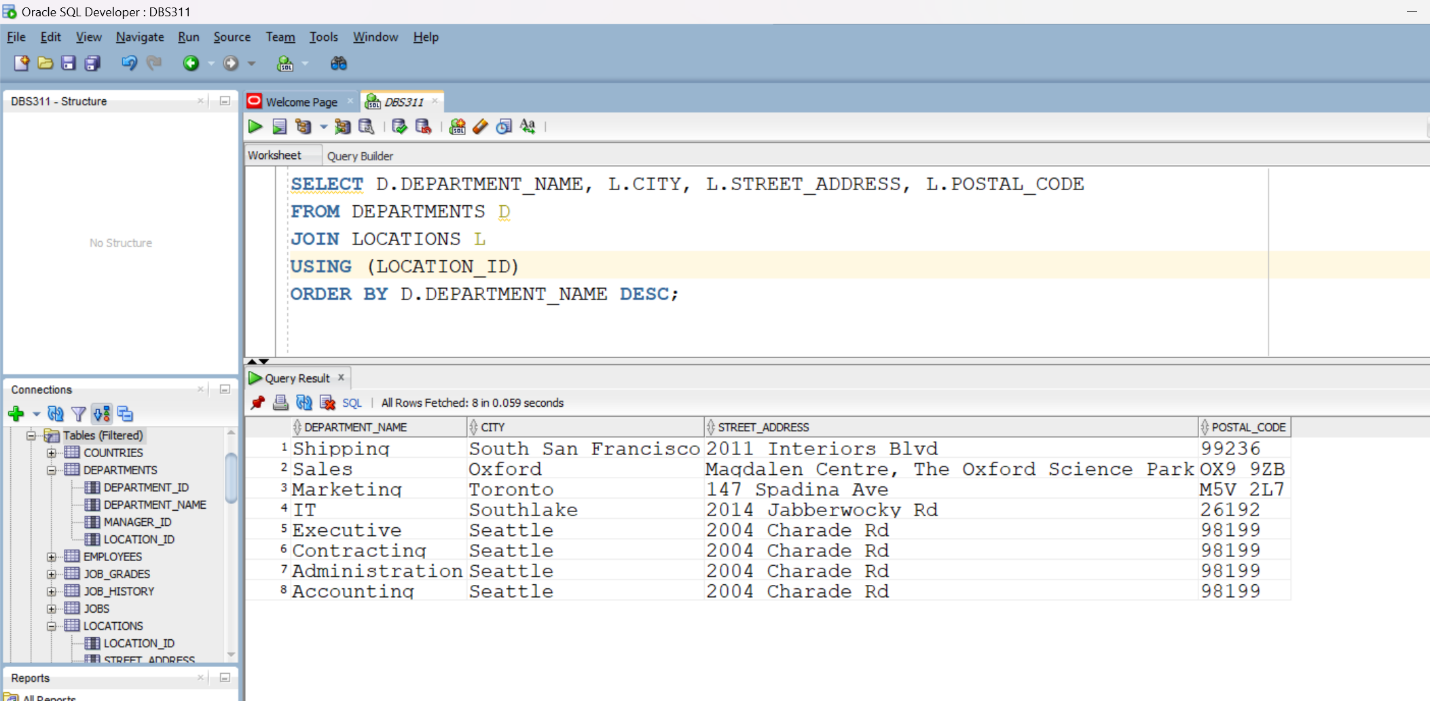
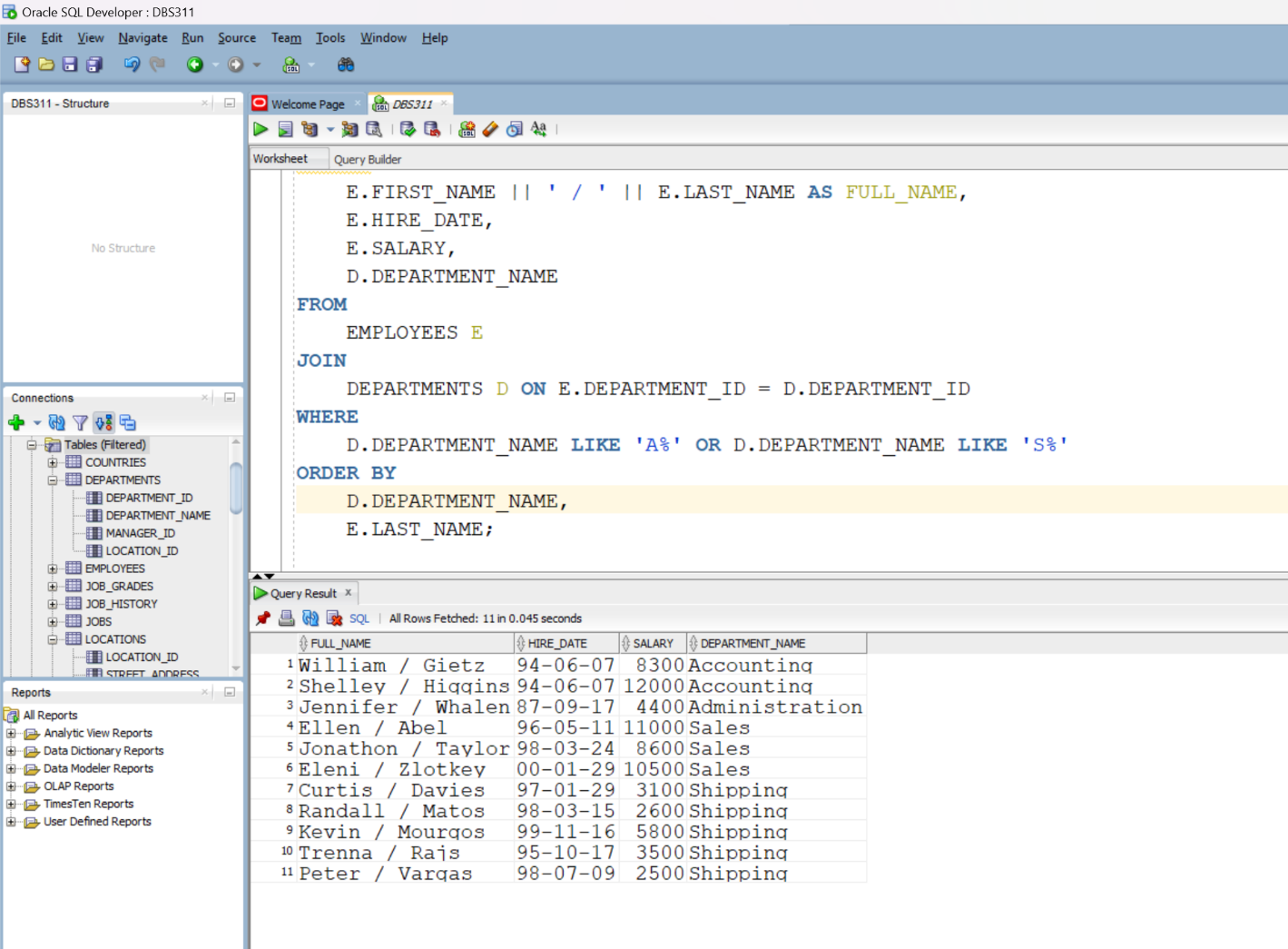
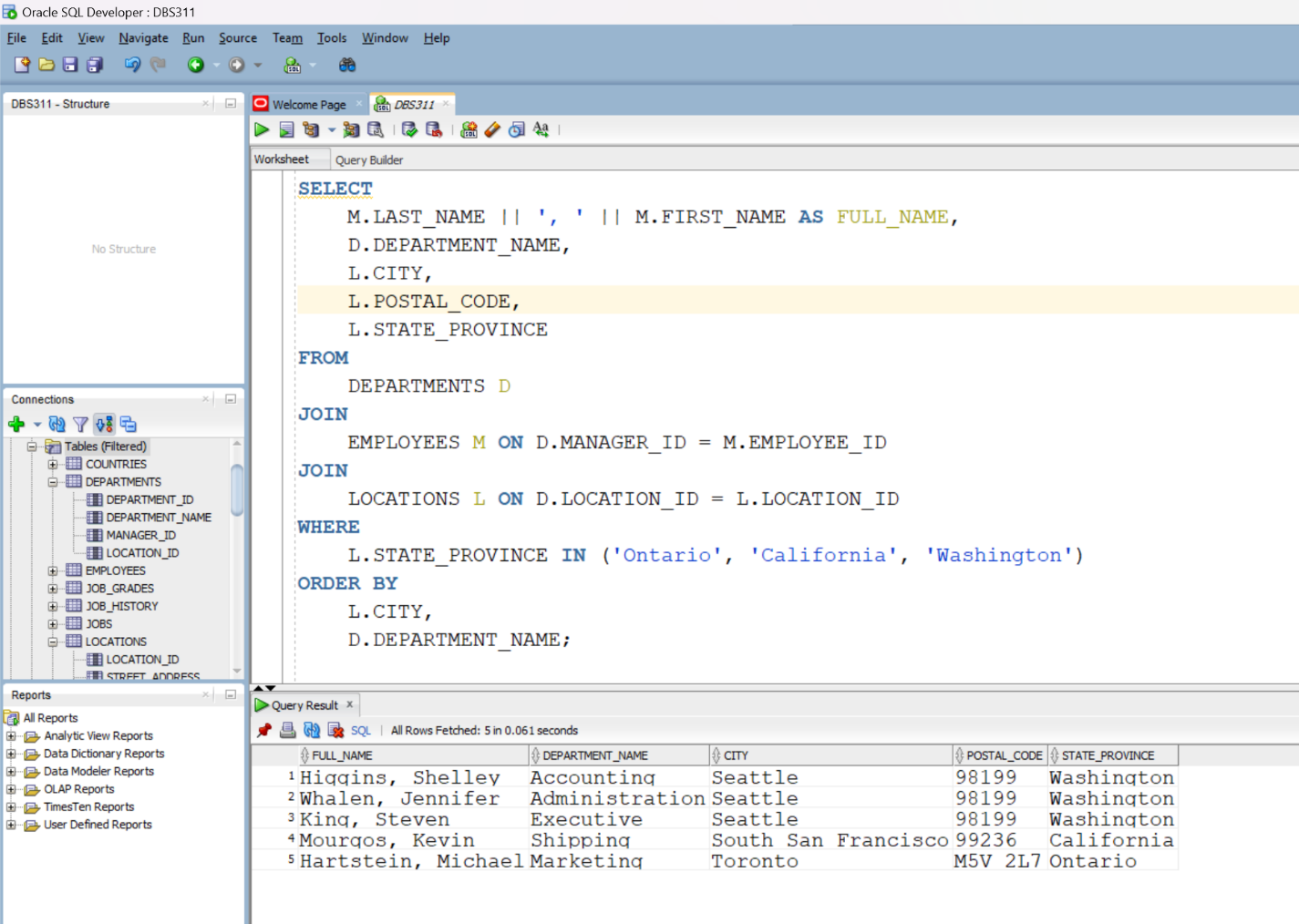
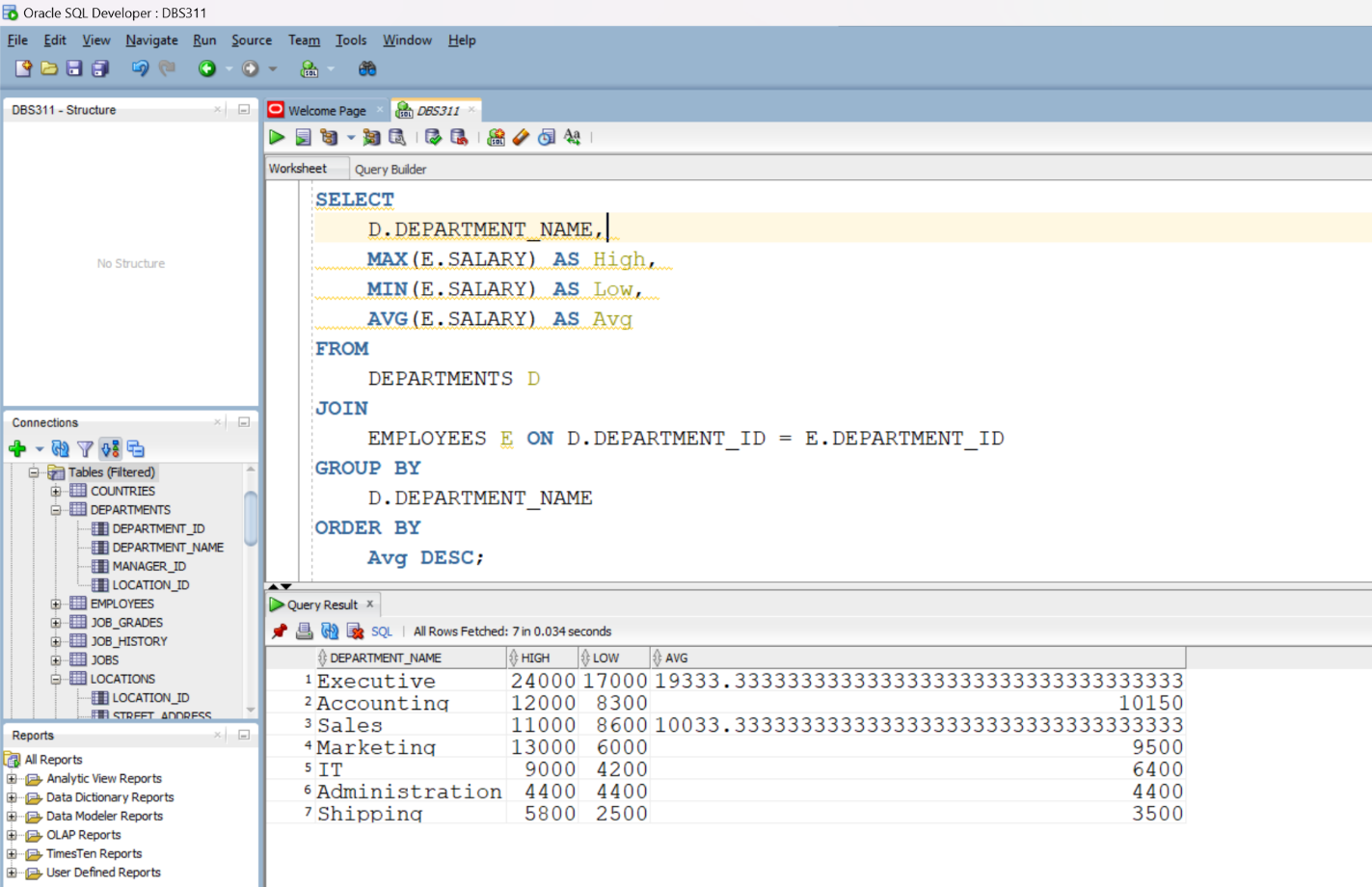
ANS1:

ANS2:

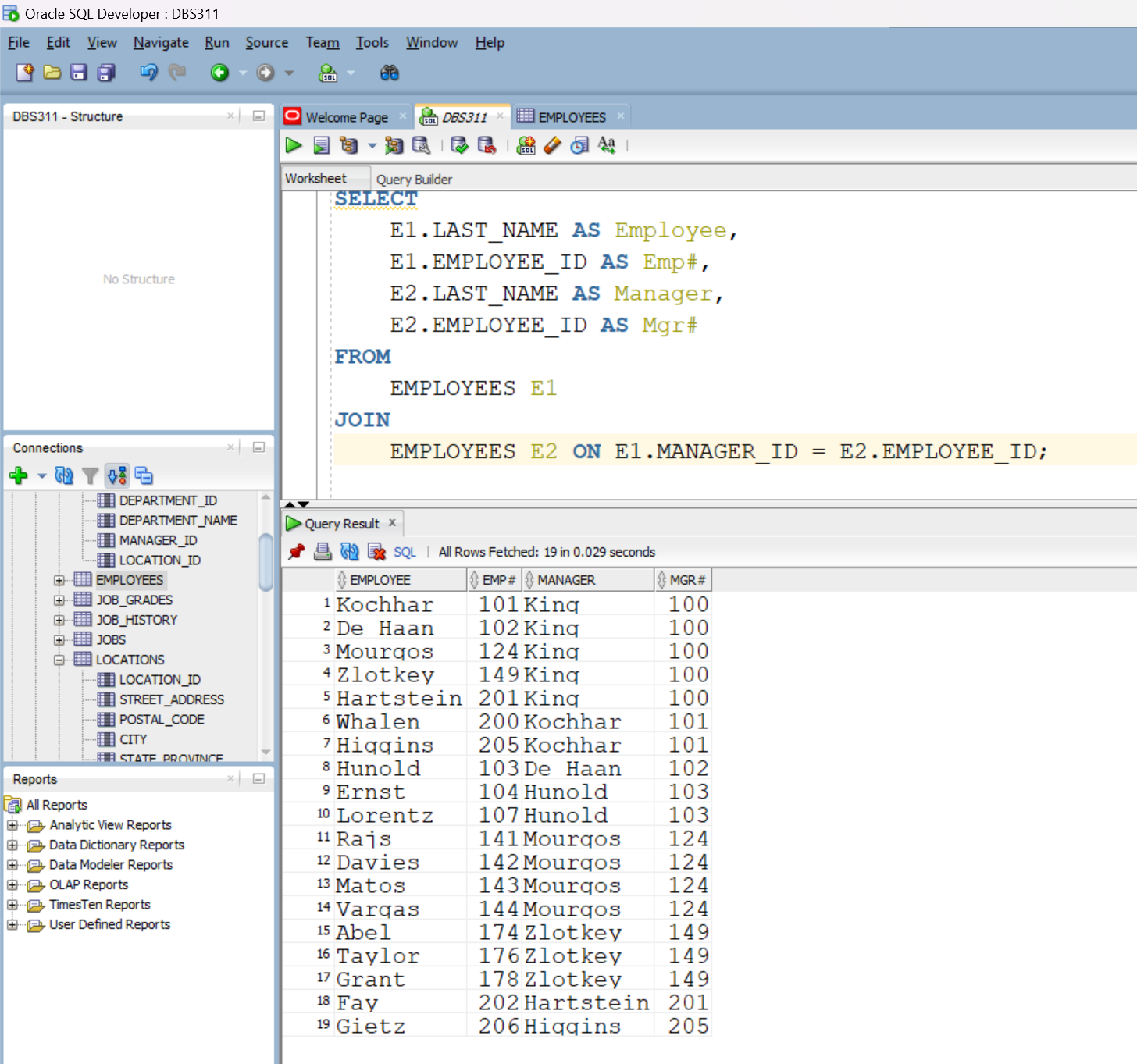


ANS3:

ANS4:



ANS5:



ANS6:  
Joins combine rows from two or more tables based on the same columns, creating a single, more efficient something that is often easier to understand. Subqueries are nested queries that set conditions or generate values consequently. Subqueries give flexibility for complex requirements, but they can be difficult to understand and less efficient. Use joins to combine tables and display related data, and subqueries to filter or compute based on intermediate results.

ANS7:

the issue with the original query lies in the equality operator (=) used in the subquery's WHERE clause. This operator demands an exact match, causing the subquery to return only one salary value per department (the maximum). By replacing "=" with "<" (less than), it transforms the condition to retrieve salaries lower than the maximum, ensuring inclusivity and resolving the problem of filtering out potentially relevant employees.

ANS8:

The \*\*IN\*\* operator checks whether a given value matches any value in a subquery or a list. It is commonly used when comparing a value to multiple possibilities. In contrast, the ANY operator compares a value to a set of values from a subquery. It's more adaptable than IN because it accepts conditions other than equality, like greater than or less than comparisons. It is useful when you need to compare a value to multiple potential values and the specific relationship changes. NOT IN is the inverse of IN, indicating that a value does not match any value in a list or subquery. It is used when you want to exclude certain values from consideration. ALL compares a value to all values returned by a subquery. It is commonly used in conjunction with a comparison operator like greater than or less than. It is useful when you want to ensure that a condition applies to all values returned by a subquery.