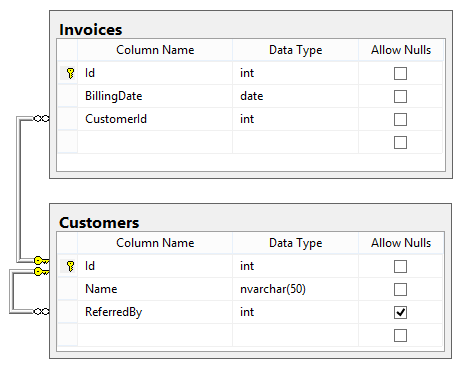
Considering the database schema displayed in the SQL Server - style diagram below, write a SQL query to return a list of all the invoices. **For each invoice, show the Invoice ID, the billing date, the customer’s name, and the name of the customer who referred to that customer (if any)**. The list should be ordered by billing date.



This question simply tests the candidate’s ability take a plain-English requirement and write a corresponding SQL query. There is nothing tricky in this one, it just covers the basics:

* Did the candidate remember to use a LEFT JOIN instead of an inner JOIN when joining the customer table for the referring customer name? If not, any invoices by customers not referred by somebody will be left out altogether.
* Did the candidate alias the tables in the JOIN? Most experienced T-SQL programmers always do this, because repeating the full table name each time it needs to be referenced gets tedious quickly. In this case, the query would actually break if at least the Customer table wasn’t aliased, because it is referenced twice in different contexts (once as the table which contains the name of the invoiced customer, and once as the table which contains the name of the referring customer).
* Did the candidate disambiguate the Id and Name columns in the SELECT? Again, this is something most experienced programmers do automatically, whether or not there would be a conflict. And again, in this case there would be a conflict, so the query would break if the candidate neglected to do so.

Note that this query will not return Invoices that do not have an associated Customer. This may be the correct behavior for most cases (e.g., it is guaranteed that every Invoice is associated with a Customer, or unmatched Invoices are not of interest). However, in order to guarantee that all Invoices are returned no matter what, the Invoices table should be joined with Customers using LEFT JOIN: