Assessment type (☑):

☐ Questioning (Oral/Written)
☐ Practical Demonstration
☐ 3rd Party Report
☒ Other – Portfolio *(please specify)*

Assessment Resources:

- *Exercises:* https://www.w3schools.com/sql/exercise.asp
- *Introductory tutorial:* SQL Tutorial (w3schools.com)
- Documentation tools (office suite, operating system utilities)

Assessment Instructions:

Complete the exercises here https://www.w3schools.com/sql/exercise.asp using the introductory tutorial as reference when needed and then answer the following questions:

1. What keyword allows you to select unique records (that is omit rows with the same result)?

DISTINCT

SELECT DISTINCT EnglishProductName

FROM DimProduct;

2. What do we mean by "same result"? Write an example that selects distinct records, what column(s) has/have to be unique in your example?

"Same result" means rows that have identical values in the selected column(s). Here, a combination of EnglishProductName **and** DaysToManufacture must be unique. If both values match in multiple rows, only one distinct record is returned.

Example:

SELECT DISTINCT EnglishProductName, DaysToManufacture

FROM DimProduct;

3. What keyword(s) allows us to sort in ascending order by a given column?

The keyword used to sort in ascending order by a given column in SQL is:

ORDER BY (ascending is the default): ASC

```
SELECT *

FROM DimProduct

ORDER BY EnglishProductName;
```

4. In the following example, what is the default sort order (which column, ascending or descending)?

```
SELECT * FROM Customers
ORDER BY Country;
```

Default sort order is ascending based on Country

5. What would you use a JOIN for?

In SQL, a JOIN joins rows from two or more tables according to a relevant column, usually a foreign key in one table and a primary key in another. It enables you to use a single query to retrieve related data from several tables.

```
SELECT SalesOrderNumber, UnitPrice

FROM FactInternetSales
```

```
INNER JOIN DimCustomer ON FactInternetSales.CustomerKey =

DimCustomer.CustomerKey;
```

6. In your own words, briefly describe the difference between a LEFT/RIGHT/INNER/FULL OUTER JOINs (see SQL Joins (w3schools.com))

Here's a simple breakdown in my own words:

- **INNER JOIN:** Only the rows with matching values in both tables are returned by an inner join.
- **LEFT JOIN:** All rows from the left table plus any matching rows from the right table (or NULLs if there is no match) are returned by LEFT JOIN.
- **RIGHT JOIN:** In contrast to LEFT JOIN, a RIGHT JOIN yields all rows from the right table together with any matches from the left.
- **FULL OUTER JOIN:** Returns all rows from both tables, with NULLs where no match exists on either side.

7. The following question refer to SQL wildcards (https://www.w3schools.com/sql/exercise.asp?filename=exercise_wildcards2), a query returns the following list of countries [Angola, Australia, United States], which of these are possible wildcards and why:

LIKE '[!abc]%': means first letter **must NOT be a, b, or c**.

Angola & Australia (A) is excluded (A is in abc)

United States (U) is included because U is NOT a, b, or c

Possible wildcard (because it could match United States)

LIKE '[!def]%': means First letter must NOT be **d, e, or f**.

Includes *Angola* (A), *Australia* (A) & *United States* (U)

Possible wildcard because could match all three

LIKE '[bn]%': means First letter must be **b or n**.

Excludes *Angola* (A), *Australia* (A) and *United States* (U)

Not a possible wildcard (none start with b or n)

8. Write two queries one to extract all records from `Customer` where `PostalAddress` is empty and one to extract all records from `Customer` where `PostalAddress` is not empty.

```
SELECT *

FROM Customer

WHERE PostalAddress IS NULL

OR PostalAddress = "";

SELECT *

FROM Customer

WHERE PostalAddress IS NOT NULL
```

```
AND PostalAddress <> “”;
```

9. Write two queries one to count all records from `Customer` where `PostalAddress` is empty and one to count all records from `Customer` where `PostalAddress` is not empty.

```
SELECT COUNT(*) AS EmptyPostalAddressCount

FROM Customer

WHERE PostalAddress = “”;

SELECT COUNT(*) AS NonEmptyPostalAddressCount

FROM Customer

WHERE PostalAddress IS NOT NULL AND PostalAddress <> “”;
```

10. In plain English, explain what the following query does (consider the business significance of the result):

```
SELECT COUNT(CustomerID),
Country
FROM Customers
GROUP BY Country
ORDER BY COUNT(CustomerID) DESC;
```

This query determines the number of clients from each country. After that, it lists the total number of clients in each country after grouping them by country. Lastly, the findings are arranged from the most customer-rich country to the least customer-rich.

Business significance: It assists the company in determining which nations have the

biggest clientele, which is helpful for developing expansion plans, allocating resources, and focussing marketing efforts.