**Table of Contents**

* [Northwind Database Overview](https://learnsql.com/blog/sql-exercises-northwind/#northwind-database-overview)
* [Northwind Database Exercises](https://learnsql.com/blog/sql-exercises-northwind/#northwind-database-exercises)
  + [1. Single Table Queries](https://learnsql.com/blog/sql-exercises-northwind/#1-single-table-queries)
    - [Exercise 1: All Products](https://learnsql.com/blog/sql-exercises-northwind/#exercise-1-all-products)
    - [Exercise 2: Products More Expensive Than 3.5](https://learnsql.com/blog/sql-exercises-northwind/#exercise-2-products-more-expensive-than-3-5)
    - [Exercise 3: Customers’ Names and Emails](https://learnsql.com/blog/sql-exercises-northwind/#exercise-3-customers-rsquo-names-and-emails)
  + [2. Queries from Multiple Tables](https://learnsql.com/blog/sql-exercises-northwind/#2-queries-from-multiple-tables)
    - [Exercise 4: Product and Categories](https://learnsql.com/blog/sql-exercises-northwind/#exercise-4-product-and-categories)
    - [Exercise 5: Employees and Purchases, Part 1](https://learnsql.com/blog/sql-exercises-northwind/#exercise-5-employees-and-purchases-part-1)
    - [Exercise 6: Employees and Purchases, Part 2](https://learnsql.com/blog/sql-exercises-northwind/#exercise-6-employees-and-purchases-part-2)
  + [3. ORDER BY: Sorting Your Results](https://learnsql.com/blog/sql-exercises-northwind/#3-order-by-sorting-your-results)
    - [Exercise 7: Employees by Birth Date](https://learnsql.com/blog/sql-exercises-northwind/#exercise-7-employees-by-birth-date)
    - [Exercise 8: Products’ Names and Prices](https://learnsql.com/blog/sql-exercises-northwind/#exercise-8-products-rsquo-names-and-prices)
    - [Exercise 9: Purchase Data](https://learnsql.com/blog/sql-exercises-northwind/#exercise-9-purchase-data)
  + [4. GROUP BY and Aggregate Functions](https://learnsql.com/blog/sql-exercises-northwind/#4-group-by-and-aggregate-functions)
    - [Exercise 10: Number of Categories](https://learnsql.com/blog/sql-exercises-northwind/#exercise-10-number-of-categories)
    - [Exercise 11: Customer Purchases](https://learnsql.com/blog/sql-exercises-northwind/#exercise-11-customer-purchases)
    - [Exercise 12: Revenue for Customer and Employee Pairs](https://learnsql.com/blog/sql-exercises-northwind/#exercise-12-revenue-for-customer-and-employee-pairs)
  + [5. Other SQL Features](https://learnsql.com/blog/sql-exercises-northwind/#5-other-sql-features)
    - [Exercise 13: Unknown Hire Date](https://learnsql.com/blog/sql-exercises-northwind/#exercise-13-unknown-hire-date)
    - [Exercise 14: Average Price per Category](https://learnsql.com/blog/sql-exercises-northwind/#exercise-14-average-price-per-category)
  + [6. Advanced SQL Features](https://learnsql.com/blog/sql-exercises-northwind/#6-advanced-sql-features)
    - [Exercise 15: Categories with Active Products](https://learnsql.com/blog/sql-exercises-northwind/#exercise-15-categories-with-active-products)
    - [Exercise 16: Wine Statistics](https://learnsql.com/blog/sql-exercises-northwind/#exercise-16-wine-statistics)
* [Want More Northwind Exercises for SQL Learners?](https://learnsql.com/blog/sql-exercises-northwind/#want-more-northwind-exercises-for-sql-learners)

*When it comes to learning SQL, many beginners struggle to gain practical experience that can eventually translate to real-world scenarios. We'll tackle this challenge by presenting Northwind database SQL exercises with solutions and explanations.*

In this article, we’ll cover 16 Northwind exercises taken from our SQL practice course [Basic SQL Practice: A Store](https://learnsql.com/course/basic-sql-practice-store/). As you might have guessed, the course uses Microsoft’s classic [Northwind database](https://github.com/microsoft/sql-server-samples/tree/master/samples/databases/northwind-pubs). It’s a small trading company database that’s designed to give SQL learners some practical experience. It includes realistic data for customers, products, suppliers, and orders, making it an ideal starting point for those looking to build foundational SQL skills.

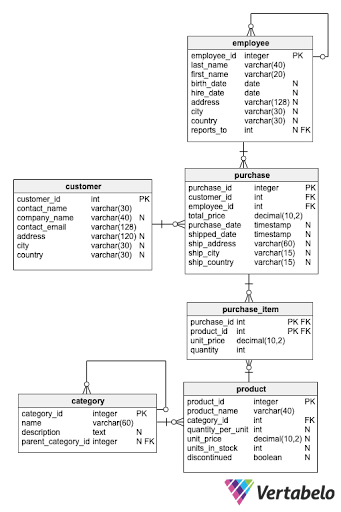
More data is being generated daily, making databases a must for safe and organized information storage. Using SQL, or Structured Query Language, we can draw insights from the data inside these databases. SQL allows us to extract, analyze, and manipulate data for our analysis. While learning SQL may feel challenging at first, proficiency grows with practice – just like learning any language.

[LearnSQL.com](https://learnsql.com/?itm_source=lsqlBlog&itm_campaign=sqlassessment2023&itm_medium=text&itm_content=platform-7) lets you learn SQL by writing SQL code on your own. You build your SQL skills gradually. Each new concept is reinforced by an interactive exercise. By actually writing SQL code, you build your confidence.

The [Basic SQL Practice: A Store](https://learnsql.com/course/basic-sql-practice-store/) course itself offers 169 interactive exercises that cover a wide range of SQL topics. To familiarize ourselves with the type of content in the course, we’ll look at questions from each area. These exercises will help you improve your SQL skills and confidence.

**Northwind Database Overview**

Let's begin by taking a look at the database we’ll be using:



The tables in this database each represent a unique object and are linked to other tables via [primary and foreign keys](https://learnsql.com/blog/why-use-primary-key-foreign-key/). The [primary key](https://learnsql.com/blog/primary-key-vs-unique-key-sql/) uniquely identifies each record in a table (for example, customer\_id in the Customer table), allowing us to connect tables and get relevant data by creating a bridge between them.

Let’s briefly look at each table:

* employee: This table holds data on the company's employees, which can be useful for HR and employee performance analysis.
* purchase: This table holds data for transactions between customers and the company. This information can aid in purchase analysis, stock planning, and location analysis.
* customer: This table contains data on the company's customers. This can be useful for identifying target audiences and personalizing services.
* purchase\_item: This table connects purchases with products and allows us to gain information on the prices and quantities of the different products in each order.
* product: This table displays data on the products in the company’s offer. This can provide insights into popular products and their details.
* category: This table contains information about different product categories, which can help in gaining insights on popular categories.

**Northwind Database Exercises**

Now let’s dive into some practical SQL exercises based on this database. We are going to break down this article into six sections that each introduce different SQL skills. Keep our [SQL Cheat Sheet](https://learnsql.com/blog/sql-basics-cheat-sheet/) nearby for quick reference on the functions and syntax we will be using.

**1. Single Table Queries**

To begin, we will start with the basics: filtering and retrieving data from a single table. In these exercises, you’ll use the SELECT and WHERE clauses to extract specific information from individual tables. Mastering these essential SQL commands will lay a strong foundation for building more complex queries later on.

**Exercise 1: All Products**

Display all the data present in the product table.

**Solution:**

|  |
| --- |
| SELECT \*  FROM product; |

**Explanation:**

In this query:

* SELECT \* instructs SQL to select all columns from the product
* FROM product specifies the table we are using.
* This query returns all rows and columns in the product table, providing a complete view of each product’s details in the database.

**Exercise 2: Products More Expensive Than 3.5**

Display the names of products with a unit price greater than or equal to 3.5.

**Solution:**

|  |
| --- |
| SELECT product\_name  FROM product  WHERE unit\_price >= 3.5; |

**Explanation:**

In this query:

* SELECT product\_name specifies we want just the product names.
* FROM product specifies the table we are using.
* WHERE unit\_price >= 5 limits the output to products priced at 3.5 or above.
* This query is useful for identifying the more expensive items in the store.

**Exercise 3: Customers’ Names and Emails**

Display customers’ names and emails. You'll find the name in the contact\_name column and the email in the contact\_email column of the customer table. Rename the columns to name and email, respectively.

**Solution:**

|  |
| --- |
| SELECT    contact\_name AS name,    contact\_email AS email  FROM customer; |

**Explanation:**

In this query:

* SELECT contact\_name AS name takes the column contact\_name and renames it in the output as name. It does the same thing with contact\_email AS email.
* FROM customer specifies the table we are using.
* Aliasing with AS makes the results more readable and user-friendly. This can be especially useful when sharing results with non-technical stakeholders.

You can find more examples of practical yet simple exercises in our article[SQL Practice for Beginners: AdventureWorks Exercises](https://learnsql.com/blog/sql-practice-for-beginners-adventureworks-exercises/).

**2. Queries from Multiple Tables**

This section focuses on combining data from different tables using JOINs; this enables us to create more complex queries.

**Exercise 4: Product and Categories**

Select product names along with their categories. Display two columns: product\_name and category\_name.

**Solution:**

|  |
| --- |
| SELECT    product\_name,    category.name AS category\_name  FROM product  JOIN category    ON product.category\_id = category.category\_id; |

**Explanation:**

In this query:

* SELECT product\_name, category.name AS category\_name identifies the columns that are requested in our output.
* FROM product specifies the first table in our JOIN.
* JOIN category specifies the second table in our JOIN.
* ON product.category\_id = category.category\_id acts as a bridge between both tables and ensures that only rows with matching category IDs are included in the result set.

If you want to practice SQL using hands-on exercises, try our 100% online [SQL Practice](https://learnsql.com/track/sql-practice?itm_source=lsqlBlog&itm_campaign=sqlassessment2023&itm_medium=text&itm_content=track-sql-practice-7) track.

**Exercise 5: Employees and Purchases, Part 1**

Show the last and first names of the employees handling purchases along with the shipped\_date of the purchase.

**Solution:**

|  |
| --- |
| SELECT    last\_name,    first\_name,    shipped\_date  FROM employee  JOIN purchase    ON employee.employee\_id = purchase.employee\_id; |

**Explanation:**

In this query:

* SELECT last\_name, first\_name, shipped\_date identifies the columns that are shown in our output.
* FROM employee indicates the first table in our JOIN.
* JOIN purchase specifies the second table in our JOIN.
* ON employee.employee\_id = purchase.employee\_id joins both tables on rows where the employee ID in the employeetable matches the employee ID in the purchase Only rows with matching IDs are included in the result set.

**Exercise 6: Employees and Purchases, Part 2**

For each employee, display their last\_name, first\_name, and the ID of the purchase(s) they’ve handled (if such exist). Make sure to display data for all employees, even if they haven't been involved with any purchases.

**Solution:**

|  |
| --- |
| SELECT    last\_name,    first\_name,    purchase\_id  FROM employee  LEFT JOIN purchase    ON employee.employee\_id = purchase.employee\_id; |

**Explanation:**

In this query:

* SELECT last\_name, first\_name, purchase\_id specifies the columns to be included in the output.
* FROM employee designates the first table in our LEFT JOIN, which contains details about employees.
* LEFT JOIN purchase specifies the second table in the join, which holds data about purchases. Using LEFT JOIN ensures that all rows from the employee table are included, even if there is no matching record in the purchase
* ON employee.employee\_id = purchase.employee\_id connects both tables, matching records based on the employee ID. If an employee has no corresponding record in thepurchase table, they will still be included in the result and the purchase fields in the result set will show as NULL.

As you can see, writing queries really builds your SQL skills! Another great way to build your SQL skills is through projects like the ones in [SQL Project for Beginners: AdventureWorks Sales Dashboard](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/).

**3. ORDER BY: Sorting Your Results**

The ORDER BY clause in SQL is particularly useful for organizing results in a more readable and meaningful way.  Whether you want to sort data numerically, alphabetically, or by date, this is very useful in reporting and analytics.

**Exercise 7: Employees by Birth Date**

Show the last names, first names, and birth dates of employees. Sort the employees by their birth date in descending order (i.e. the most recent dates come first).

**Solution:**

|  |
| --- |
| SELECT    last\_name,    first\_name,    birth\_date  FROM employee  ORDER BY birth\_date DESC; |

**Explanation:**

In this query:

* SELECT last\_name, first\_name, birth\_date specifies the columns to include in the output.
* FROM employee indicates the table from which data is retrieved.
* ORDER BY birth\_date DESC sorts the results by the birth\_date column in descending order, so the employees with the most recent birth dates appear first. This overcomes the limitation of not having an age column.

**Exercise 8: Products’ Names and Prices**

Display the names and unit prices of all products. Show the cheapest products first.

**Solution:**

|  |
| --- |
| SELECT    product\_name,    unit\_price  FROM product  ORDER BY unit\_price; |

**Explanation:**

In this query:

* SELECT product\_name, unit\_price specifies the columns to be included in the output.
* FROM product identifies the table from which the data is taken.
* ORDER BY unit\_price sorts the results by the unit\_price column in ascending order, meaning that the products with the lowest prices will appear first.
* As you can see, we don’t need to explicitly specify ASCending; it is the default sort order.

**Exercise 9: Purchase Data**

Show all data for all purchases. Sort the rows by the ship city in ascending order and by the shipped date in descending order.

**Solution:**

|  |
| --- |
| SELECT \*  FROM purchase  ORDER BY    ship\_city ASC,    shipped\_date DESC; |

**Explanation:**

In this query:

* SELECT \* specifies that all columns from the table should be included in the output.
* FROM purchase identifies the table from which data is retrieved.
* ORDER BY ship\_city ASC, shipped\_date DESC sorts the results first by ship\_city in ascending order (so that the cities are sorted alphabetically). Then within each city, the results are further sorted by shipped\_date in descending order (so the most recent shipment dates appear first for each city).

If you want to find more exercises with ORDER BY, check out our article [10 Beginner SQL Practice Exercises with Solutions](https://learnsql.com/blog/sql-practice-exercises/).

**4. GROUP BY and Aggregate Functions**

The next section helps build the skill of summarizing data through grouping and aggregation. Using aggregate functions, we can gain insights by combining rows and conducting calculations within each group.

**Exercise 10: Number of Categories**

Count the number of categories present in the category table. Name the column number\_of\_categories.

**Solution:**

|  |
| --- |
| SELECT    COUNT(category\_id) AS number\_of\_categories  FROM category; |

**Explanation:**

In this query:

* SELECT COUNT(category\_id) AS number\_of\_categories counts the number of rows that do not have NULLs in the category\_id It displays the resulting column as number\_of\_categories.
* FROM category specifies the table from which the data is retrieved.

**Exercise 11: Customer Purchases**

Count the number of purchases made by each customer. Display the customer\_id, contact\_name, and purchases\_number. Ignore any customers that aren't present in the purchase table.

**Solution:**

|  |
| --- |
| SELECT    purchase.customer\_id,    contact\_name,    COUNT(\*) AS purchases\_number  FROM purchase  JOIN customer    ON purchase.customer\_id = customer.customer\_id  GROUP BY    purchase.customer\_id,    contact\_name; |

**Explanation:**

In this query:

* SELECT purchase.customer\_id, contact\_name, COUNT(\*) AS purchases\_number specifies the columns to be included in the output. It retrieves each customer’s ID and contact name, along with the count of their purchases (labeled purchases\_number).
* FROM purchase identifies the first table in the JOIN.
* JOIN customer specifies the second table in the JOIN.
* ON purchase.customer\_id = customer.customer\_id links the two tables, matching rows where the customer\_id is the same in both tables and ensuring only purchases associated with existing customers are included.
* GROUP BY purchase.customer\_id, contact\_name groups the results by each customer’s ID and contact name, allowing the COUNT(\*) function to calculate the number of purchases for each customer.

Tons of practice in an interactive environment? Just click [here](https://learnsql.com/track/sql-practice?itm_source=lsqlBlog&itm_campaign=sqlassessment2023&itm_medium=text&itm_content=track-sql-practice-2) and check out our SQL Practice track!

**Exercise 12: Revenue for Customer and Employee Pairs**

For each customer and employee, find the total price of all purchases they made to which a given employee is assigned. Display three columns: customer\_id, employee\_id, and the total price of purchases. Rename the third column to total\_purchases\_price.

**Solution:**

|  |
| --- |
| SELECT    customer\_id,    employee\_id,    SUM(total\_price) AS total\_purchases\_price  FROM purchase  GROUP BY    customer\_id,    employee\_id; |

**Explanation:**

In this query:

* SELECT customer\_id, employee\_id, SUM(total\_price) AS total\_purchases\_price specifies the columns to include in the output, showing each customer\_id and employee\_id along with the sum of total\_price for purchases associated with each combination. This sum is labeled as total\_purchases\_price in the result.
* FROM purchase identifies the table from which data is taken.
* GROUP BY customer\_id, employee\_id groups the results by both customer\_id and employee\_id, allowing the SUM(total\_price) function to calculate the total purchase price for each unique customer–employee combination.

**5. Other SQL Features**

Next, we'll briefly cover some more SQL features that are included in the course but are also useful in day-to-day analysis. These functions, when combined with others discussed in this article, can help you improve your SQL skills.

**Exercise 13: Unknown Hire Date**

Display the last and first names of employees with an unknown hire date.

**Solution:**

|  |
| --- |
| SELECT    last\_name,    first\_name  FROM employee  WHERE hire\_date IS NULL; |

**Explanation:**

In this query:

* SELECT last\_name, first\_name specifies the columns to include in the output.
* FROM employee indicates the table from which data is taken.
* WHERE hire\_date IS NULL filters the results to only include employees whose hire\_date is NULL, meaning their hire date is unknown.

**Exercise 14: Average Price per Category**

For each category, count the average price of its products. Display only the categories for which the average unit price is greater than the overall average unit price. Name the columns category\_name and average\_price.

**Solution:**

|  |
| --- |
| SELECT    category.name AS category\_name,    AVG(unit\_price) AS average\_price  FROM product  JOIN category    ON product.category\_id = category.category\_id  GROUP BY category.name  HAVING AVG(unit\_price) > (    SELECT AVG(unit\_price)    FROM product  ); |

**Explanation:**

In this query:

* SELECT category.name AS category\_name, AVG(unit\_price) AS average\_price specifies the columns to include in the output.
* FROM product identifies the first table in the JOIN.
* JOIN category specifies the second table in the JOIN.
* ON product.category\_id = category.category\_id links the two tables, ensuring that only products with matching category IDs from both tables are included.
* GROUP BY category.name groups the results by category name, allowing the AVG(unit\_price) function to calculate the average price for each category.
* HAVING AVG(unit\_price) > (SELECT AVG(unit\_price) FROM product) filters the grouped results to include only categories where the average price of products is greater than the overall average price of all products in the product
* The subquery (SELECT AVG(unit\_price) FROM product) calculates the overall average unit price across all products.

**6. Advanced SQL Features**

Finally, we'll look at some more advanced SQL queries. As we do this, we’ll reference functions we’ve looked at earlier.

**Exercise 15: Categories with Active Products**

For each category, display the number of its products that aren't discontinued (they are continued or there is a NULL in the discontinued column). Show the columns named category\_name and products\_number. Show only the rows for which the number of such products is greater than 1. Also, don't show the row for the Other category.

**Solution:**

|  |
| --- |
| SELECT    category.name AS category\_name,    COUNT(product\_id) AS products\_number  FROM product  JOIN category    ON product.category\_id = category.category\_id  WHERE category.name <> 'Other'    AND discontinued IS NOT TRUE  GROUP BY category.name  HAVING COUNT(product\_id) > 1; |

**Explanation:**

In this query:

* SELECT category.name AS category\_name, COUNT(product\_id) AS products\_number specifies the columns to include in the output.
* FROM product identifies the first table in the JOIN.
* JOIN category specifies the second table in the JOIN.
* ON product.category\_id = category.category\_id links the two tables, ensuring that only products with matching category IDs from both tables are included.
* WHERE category.name <> 'Other' AND discontinued IS NOT TRUE filters the results to exclude categories named "Other" and to only include products that are not discontinued.
* GROUP BY category.name groups the results by category name, allowing the COUNT(product\_id) function to count the number of products within each category.
* HAVING COUNT(product\_id) > 1 filters the grouped results to include only categories with more than one product.

**Exercise 16: Wine Statistics**

All wines in the product table have a name starting with Wine. Find the:

* Number of such products in the table (products\_number).
* Total number of units in stock (units\_number).
* Average product price (average\_price).
* Ratio of the maximum price to the minimum price (max\_to\_min\_ratio).
* Difference between the maximum price and the average price (max\_to\_average).
* Difference between the average and minimum price (average\_to\_min).

Round the four last columns to two decimal points.

**Solution:**

|  |
| --- |
| SELECT    COUNT(\*) AS products\_number,    SUM(units\_in\_stock) AS units\_number,    ROUND(AVG(unit\_price), 2) AS average\_price,    ROUND(MAX(unit\_price) / MIN(unit\_price), 2) AS max\_to\_min\_ratio,    ROUND(MAX(unit\_price) - AVG(unit\_price), 2) AS max\_to\_average,    ROUND(AVG(unit\_price) - MIN(unit\_price), 2) AS average\_to\_min  FROM product  WHERE product\_name LIKE 'Wine%'; |

**Explanation:**

In this query:

* First let's break down the SELECT statement:
  + products\_number calculates the total number of products.
  + units\_number sums the total number of units in stock.
  + average\_price gets the average unit price of the products, rounded to 2 decimal places.
  + max\_to\_min\_ratio finds the ratio of the maximum unit price to the minimum unit price, rounded to 2 decimal places.
  + max\_to\_average finds the difference between the maximum unit price and the average unit price, rounded to 2 decimal places.
  + average\_to\_min calculates the difference between the average unit price and the minimum unit price, rounded to 2 decimal places.
* FROM product identifies the table from which the data is taken.
* WHERE product\_name LIKE 'Wine%' filters the results to include only products whose names start with 'Wine'.

**Want More Northwind Exercises for SQL Learners?**

By practicing writing queries, you can strengthen your SQL abilities. Working with real-world data, such as the Northwind Database, gives you experience accessing and evaluating information fast.

[](https://learnsql.com/sql-skill-assessment?itm_source=lsqlBlog&itm_campaign=sqlassessment2023&itm_medium=img&itm_content=platform-12)

If you have found the Northwind exercises in this article interesting, be sure to check out the course [Basic SQL Practice: A Store](https://learnsql.com/course/basic-sql-practice-store/). You will find plenty more interactive exercises. We also have [SQL Project for Portfolio: Northwind Store](https://learnsql.com/blog/sql-project-northwind-store/), an article on building a project based on the Northwind database[.](https://learnsql.com/blog/sql-project-northwind-store/) Happy learning!

Tags:

* [SQL Practice](https://learnsql.com/tags/sql-practice/)

**You may also like**

slide 6 to 8 of 4

**[10 Beginner SQL Practice Exercises With Solutions](https://learnsql.com/blog/sql-practice-exercises/)**

[Here are ten SQL practice exercises for your beginner SQL skills. Then, check how you did by comparing your solution to ours.](https://learnsql.com/blog/sql-practice-exercises/)

**[Read more](https://learnsql.com/blog/sql-practice-exercises/)**

**[Your Guide to SQL Practice at LearnSQL.com](https://learnsql.com/blog/sql-practice-guide/)**

[Boost your SQL skills with LearnSQL.com's practice articles and courses designed for all levels, from beginners to advanced users.](https://learnsql.com/blog/sql-practice-guide/)

**[Read more](https://learnsql.com/blog/sql-practice-guide/)**

**[SQL Project for Portfolio: Northwind Store](https://learnsql.com/blog/sql-project-northwind-store/)**

[Learn how to create an SQL project for a portfolio – from data inspection and basic data analysis to advanced analyses and actionable insights.](https://learnsql.com/blog/sql-project-northwind-store/)

**[Read more](https://learnsql.com/blog/sql-project-northwind-store/)**

**[SQL Project for Beginners: AdventureWorks Sales Dashboard](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

[Learn how to start your first SQL project by building a sales dashboard with the AdventureWorks database. Ideal for people looking to gain hands-on SQL practice.](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)

**[Read more](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

**[10 Beginner SQL Practice Exercises With Solutions](https://learnsql.com/blog/sql-practice-exercises/)**

[Here are ten SQL practice exercises for your beginner SQL skills. Then, check how you did by comparing your solution to ours.](https://learnsql.com/blog/sql-practice-exercises/)

**[Read more](https://learnsql.com/blog/sql-practice-exercises/)**

[[A person and person hiking on a map

AI-generated content may be incorrect.](https://learnsql.com/blog/sql-practice-guide/)](https://learnsql.com/blog/sql-practice-guide/)

**[Your Guide to SQL Practice at LearnSQL.com](https://learnsql.com/blog/sql-practice-guide/)**

[Boost your SQL skills with LearnSQL.com's practice articles and courses designed for all levels, from beginners to advanced users.](https://learnsql.com/blog/sql-practice-guide/)

**[Read more](https://learnsql.com/blog/sql-practice-guide/)**

[[A person standing next to a computer screen

AI-generated content may be incorrect.](https://learnsql.com/blog/sql-project-northwind-store/)](https://learnsql.com/blog/sql-project-northwind-store/)

**[SQL Project for Portfolio: Northwind Store](https://learnsql.com/blog/sql-project-northwind-store/)**

[Learn how to create an SQL project for a portfolio – from data inspection and basic data analysis to advanced analyses and actionable insights.](https://learnsql.com/blog/sql-project-northwind-store/)

**[Read more](https://learnsql.com/blog/sql-project-northwind-store/)**

[[Hands pointing at a computer screen

AI-generated content may be incorrect.](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)

**[SQL Project for Beginners: AdventureWorks Sales Dashboard](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

[Learn how to start your first SQL project by building a sales dashboard with the AdventureWorks database. Ideal for people looking to gain hands-on SQL practice.](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)

**[Read more](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

**[10 Beginner SQL Practice Exercises With Solutions](https://learnsql.com/blog/sql-practice-exercises/)**

[Here are ten SQL practice exercises for your beginner SQL skills. Then, check how you did by comparing your solution to ours.](https://learnsql.com/blog/sql-practice-exercises/)

**[Read more](https://learnsql.com/blog/sql-practice-exercises/)**

**[Your Guide to SQL Practice at LearnSQL.com](https://learnsql.com/blog/sql-practice-guide/)**

[Boost your SQL skills with LearnSQL.com's practice articles and courses designed for all levels, from beginners to advanced users.](https://learnsql.com/blog/sql-practice-guide/)

**[Read more](https://learnsql.com/blog/sql-practice-guide/)**

**[SQL Project for Portfolio: Northwind Store](https://learnsql.com/blog/sql-project-northwind-store/)**

[Learn how to create an SQL project for a portfolio – from data inspection and basic data analysis to advanced analyses and actionable insights.](https://learnsql.com/blog/sql-project-northwind-store/)

**[Read more](https://learnsql.com/blog/sql-project-northwind-store/)**

**[SQL Project for Beginners: AdventureWorks Sales Dashboard](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

[Learn how to start your first SQL project by building a sales dashboard with the AdventureWorks database. Ideal for people looking to gain hands-on SQL practice.](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)

**[Read more](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

**[10 Beginner SQL Practice Exercises With Solutions](https://learnsql.com/blog/sql-practice-exercises/)**

[Here are ten SQL practice exercises for your beginner SQL skills. Then, check how you did by comparing your solution to ours.](https://learnsql.com/blog/sql-practice-exercises/)

**[Read more](https://learnsql.com/blog/sql-practice-exercises/)**

**[Your Guide to SQL Practice at LearnSQL.com](https://learnsql.com/blog/sql-practice-guide/)**

[Boost your SQL skills with LearnSQL.com's practice articles and courses designed for all levels, from beginners to advanced users.](https://learnsql.com/blog/sql-practice-guide/)

**[Read more](https://learnsql.com/blog/sql-practice-guide/)**

**[SQL Project for Portfolio: Northwind Store](https://learnsql.com/blog/sql-project-northwind-store/)**

[Learn how to create an SQL project for a portfolio – from data inspection and basic data analysis to advanced analyses and actionable insights.](https://learnsql.com/blog/sql-project-northwind-store/)

**[Read more](https://learnsql.com/blog/sql-project-northwind-store/)**

**[SQL Project for Beginners: AdventureWorks Sales Dashboard](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

[Learn how to start your first SQL project by building a sales dashboard with the AdventureWorks database. Ideal for people looking to gain hands-on SQL practice.](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)

**[Read more](https://learnsql.com/blog/sql-project-sales-dashboard-adventureworks/)**

Top of Form

**Subscribe to our newsletter**Join our monthly newsletter to be  
notified about the latest posts.

 Email address

A person with curly hair and glasses

AI-generated content may be incorrect.

**How Do You Write a SELECT Statement in SQL?**

A person smiling at a computer

AI-generated content may be incorrect.

**What Is a Foreign Key in SQL?**

A yellow and white circle

AI-generated content may be incorrect.

**Enumerate and Explain All the Basic Elements of an SQL Query**

Bottom of Form

**Quick links**

* [Courses](https://learnsql.com/sql-courses/)
* [Pricing](https://learnsql.com/pricing/)
* [For Students](https://learnsql.com/for-students/)
* [Affiliate Program](https://learnsql.com/affiliate-program/)
* [Blog](https://learnsql.com/blog/)
* [Cookbook](https://learnsql.com/cookbook/)
* [LearnPython.com](https://learnpython.com/)
* [Vertabelo.com](https://vertabelo.com/)

**Assistance**

Need assistance? Drop us a line at **contact@learnsql.com**

[**Write to us**](mailto:contact@learnsql.com)

**Follow us**

Copyright ©2016-2025 **[Vertabelo SA](https://www.vertabelo.com/)** All rights reserved

* [Terms of service](https://learnsql.com/files/terms-of-service/)
* [Privacy policy](https://learnsql.com/files/privacy-policy/)
* [Imprint](https://learnsql.com/imprint/)