## Sort results using the ORDER BY clause

SELECT Name, ListPrice

FROM SalesLT.Product

ORDER BY ListPrice DESC, Name ASC;

Run the query and review the results. Note that they are sorted into descending order of ListPrice, and each set of products with the same price is sorted in ascending order of Name.

**Restrict results using TOP**

SELECT TOP 20 Name, ListPrice

FROM SalesLT.Product

ORDER BY ListPrice DESC;

Run the query and note that the results contain the first twenty products in descending order of **ListPrice**. Typically, you include an **ORDER BY** clause when using the **TOP** parameter; otherwise the query just returns the first specified number of rows in an arbitrary order.

SELECT TOP 20 WITH TIES Name, ListPrice

FROM SalesLT.Product

ORDER BY ListPrice DESC;

This time, there are 21 rows in the results, because there are multiple products that share the same price, one of which wasn't included when ties were ignored by the previous query.

SELECT TOP 20 PERCENT WITH TIES Name, ListPrice

FROM SalesLT.Product

ORDER BY ListPrice DESC;

Note that this time the results contain the 20% most expensive products.

## Retrieve pages of results with OFFSET and FETCH

SELECT Name, ListPrice

FROM SalesLT.Product

ORDER BY Name OFFSET 0 ROWS FETCH NEXT 10 ROWS ONLY;

Run the query and note the effect of the **OFFSET** and **FETCH** parameters of the **ORDER BY** clause. The results start at the 0 position (the beginning of the result set) and include only the next 10 rows, essentially defining the first page of results with 10 rows per page.

## Use the ALL and DISTINCT options

SELECT DISTINCT Color

FROM SalesLT.Product;

Run the modified query and note that the results include one row for each unique **Color** value. This ability to remove duplicates from the results can often be useful - for example to retrieve values in order to populate a drop-down list of color options in a user interface.

SELECT DISTINCT Color, Size

FROM SalesLT.Product;

Run the modified query and note that it returns each unique combination of color and size.

## Filter results with the WHERE clause

SELECT Name, Color, Size

FROM SalesLT.Product

WHERE ProductModelID = 6

ORDER BY Name;

Run the query and review the results, which contain the **Name**, **Color**, and **Size** for each product with a **ProductModelID** value of 6 (this is the ID for the HL Road Frame product model, of which there are multiple variants).

SELECT Name, Color, Size

FROM SalesLT.Product

WHERE ProductModelID <> 6

ORDER BY Name;

Review the results, noting that they contain all products with a **ProductModelID** other than **6**.

SELECT Name, ListPrice

FROM SalesLT.Product

WHERE ListPrice > 1000.00

ORDER BY ListPrice;

Review the results, noting that they contain all products with a **ListPrice** greater than 1000.00.

SELECT Name, ListPrice

FROM SalesLT.Product

WHERE Name LIKE 'HL Road Frame %';

Review the results, noting that the **LIKE** operator enables you to match string patterns. The **%** character in the predicate is a wildcard for one or more characters, so the query returns all rows where the **Name** is HL Road Frame followed by any string.

SELECT Name, ListPrice

FROM SalesLT.Product

WHERE ProductNumber LIKE 'FR-\_[0-9][0-9]\_-[0-9][0-9]';

Review the results. This time the results include products with a **ProductNumber** that matches patterns similar to FR-xNNx-NN (in which x is a letter and N is a numeral).

SELECT Name, ListPrice

FROM SalesLT.Product

WHERE SellEndDate IS NOT NULL;

Note that to filter based on NULL values you must use **IS NULL** (or **IS NOT NULL**) you cannot compare a NULL value using the **=** operator.

SELECT Name

FROM SalesLT.Product

WHERE SellEndDate BETWEEN '2006/1/1' AND '2006/12/31';

Review the results, which contain products that the company stopped selling in 2006.

SELECT ProductCategoryID, Name, ListPrice

FROM SalesLT.Product

WHERE ProductCategoryID IN (5,6,7);

Run above the query, which retrieves products with a **ProductCategoryID** value that is in a specified list.

SELECT ProductCategoryID, Name, ListPrice, SellEndDate

FROM SalesLT.Product

WHERE ProductCategoryID IN (5,6,7) AND SellEndDate IS NULL;

Try above the query, which filters the results to include rows that match one (or both) of two criteria.

### Challenge 1: Retrieve data for transportation reports

The logistics manager at Adventure Works has asked you to generate some reports containing details of the company’s customers to help to reduce transportation costs.

1. Retrieve a list of cities
   * Initially, you need to produce a list of all of you customers' locations. Write a Transact-SQL query that queries the **SalesLT.Address** table and retrieves the values for **City** and **StateProvince**, removing duplicates and sorted in ascending order of city.

SELECT DISTINCT City,StateProvince

FROM SalesLT.Address

ORDER BY City

2. Retrieve the heaviest products

* + Transportation costs are increasing and you need to identify the heaviest products. Retrieve the names of the top ten percent of products by weight.

SELECT TOP 10 PERCENT WITH TIES Name

FROM SalesLT.Product

ORDER BY Weight DESC

### Challenge 2: Retrieve product data

The Production Manager at Adventure Works would like you to create some reports listing details of the products that you sell.

1. Retrieve product details for product model 1
   * Initially, you need to find the names, colors, and sizes of the products with a product model ID 1.

SELECT Name,Color,Size

FROM SalesLT.Product

WHERE ProductModelID=1

2. Filter products by color and size

* + Retrieve the product number and name of the products that have a color of *black*, *red*, or *white* and a size of *S* or *M*.

SELECT ProductNumber,Name

FROM SalesLT.Product

WHERE Color IN('Black','Red','White') AND [Size] IN('S','M');

3. Filter products by product number

* + Retrieve the product number, name, and list price of products whose product number begins *BK-*

SELECT ProductNumber,Name,ListPrice

FROM SalesLT.Product

WHERE ProductNumber LIKE 'BK-%';

4. Retrieve specific products by product number

* + Modify your previous query to retrieve the product number, name, and list price of products whose product number begins *BK-* followed by any character other than *R*, and ends with a *-* followed by any two numerals.