Using W3Schools.com and SQLZoo.Net as resources, complete the following lab.

Ray Henry is the owner of a bookstore chain named Henry Books. Ray has decided to store his data in a SQL 2005 database. He wants to ensure his data is safe, current, and accurate. He would also like to be able to ask questions concerning his data and have you obtain the answers for him quickly and concisely. A new database has been completed and all of his data has been imported into this new database. You; as the companies super database guru/king; are tasked with attaching this new database to your system and answering the questions Mr. Henry has asked.

Attach the HENRY database. Write the SQL query that answers the following scenarios.

* List the book code and book title of each book. (33 Rows)

**SELECT book\_code, book\_title  
FROM books;**

* List the complete PUBLISHER table. (28 rows)

**SELECT \* FROM publishers;**

* List the name of each publisher located in Boston. (2 rows)

**SELECT publisher\_name  
FROM publishers**

**WHERE publisher\_city = 'Boston';**

* List the name of each publisher not located in Boston. (26 rows)

**SELECT publisher\_name  
FROM publishers**

**WHERE publisher\_city NOT IN ('Boston');**

* List the name of each branch that has at least nine employees. (3 rows)

**SELECT branch\_name**

**FROM branches**

**WHERE branch\_emps >= 9;**

* List the book code and book title of each book that has the type SFI. (3 rows)

**SELECT book\_code, book\_title**

**FROM books**

**WHERE book\_type = 'sfi';**

* List the book code and book title of each book that has the type SFI and is in paperback. (1 row)

**SELECT book\_code, book\_title  
FROM books  
WHERE book\_type = 'sfi' and book\_paperback = 'Y';**

* List the book code and book title of each book that has the type SFI or is published by the publisher with code SC. (5 rows)

**SELECT book\_code, book\_title  
FROM books  
WHERE book\_type = 'sfi' Or publisher\_code = 'sc';**

* List the book code, book title, and price of each book with a price between $20 and $30. (4 rows)

**SELECT book\_code, book\_title, book\_price  
FROM books  
WHERE book\_price >= 20 And book\_price <= 30;**

* List the book code and book title of each book that has the type MYS and a price of less than $20. (2 rows)

**SELECT book\_code, book\_title, book\_price**

**FROM books  
WHERE book\_price < 20 And book\_type = 'MYS';**

* Customers who are part of a special program get a 10% discount off regular book prices. List the book code, book title, and discounted price for each book. Use discounted\_price as the name for the computed column, which should calculate 90% of the current price; that is, 100% less a 10% discount. (33 rows)

**SELECT book\_code, book\_title, book\_price, ROUND(book\_price \* 0.9, 2) AS Discounted\_price   
FROM books;**

* Find the name of each publisher containing the word “and”. (4 rows)

**SELECT publisher\_name**

**FROM publishers  
WHERE publisher\_name LIKE '%and%';**

* List the book code and book title of each book that has the type SFI, MYS, or ART. (9 rows)

**SELECT book\_code, book\_title  
FROM books  
WHERE book\_type IN ('SFI', 'MYS', 'ART');**

* Same as #13 only list the books in alphabetical order by title. (9 rows)

**SELECT book\_code, book\_title  
FROM books  
WHERE book\_type IN ('SFI', 'MYS', 'ART';**

**ORDER BY book\_title ASC;**

* Same as #13 but also include the price and list the books in descending order by price. Within a group of books having the same price, further order by book title. (9 rows)

**SELECT book\_code, book\_title, book\_price  
FROM books  
WHERE book\_type IN ('SFI', 'MYS', 'ART')**

**ORDER BY book\_price DESC, book\_title ASC;**

* Display the list of book types in the database. List each book type only once. (12 rows)

**SELECT DISTINCT book\_type  
FROM books;**

* How many books have the type SFI? (write sql to produce the answer of 3)

**SELECT COUNT(\*)  
FROM books  
WHERE book\_type = 'SFI';**

* For each type of book, list the type and the average price. (Research: GROUP BY - 12 rows)

**SELECT book\_type, ROUND(AVG(book\_price), 2) AS Average\_Price  
FROM books  
GROUP By book\_type;**

* Same as #18, but consider only paperback books. (10 rows)

**SELECT book\_type, Count(\*)  
FROM books  
WHERE book\_paperback = 'Y'  
GROUP By book\_type;**

* Same as #18, but consider only paperback books for those types which the average price is more than $10. (Research: HAVING. 3 rows)

**SELECT book\_type, AVG(book\_price) AS Average\_Price  
FROM books  
WHERE book\_paperback = 'Y'**

**GROUP by book\_type  
HAVING AVG(book\_price) > 10;**

* What is the most expensive book in the database? (Nested Select - 1 row)

**SELECT book\_code, book\_title, book\_price  
FROM books  
WHERE book\_price = (SELECT MAX(book\_price) FROM books);**

* What are the title(s)/price(s) of the least expensive book(s) in the DB? (3 rows)
* How many employees does Henry Books have?

**SELECT SUM(branch\_emps) AS Total\_Employees  
FROM branches;**

**Multiple Table**

* For each book, list the book code, book title, publisher code, and the publisher name. Order the results by publisher name. (33 rows)

**SELECT b.book\_code, b.book\_title, b.publisher\_code, p.publisher\_name  
FROM books b  
INNER JOIN publishers p**

**ON b.publisher\_code = p.publisher\_code;**

* For each book published by Plume, list the book code, book title, and price. (3 rows)

**SELECT book\_code, book\_title, book\_price  
FROM books  
WHERE publisher\_code = 'PL'**

* List the book title, book code, and price of each book published by Plume that has a book price of at least $14. (1 row)

**SELECT book\_code, book\_title, book\_price  
FROM books  
WHERE publisher\_code = 'PL' AND book\_price >= 14;**

* List the book code, book title, and units on hand for each book in branch number 4. (9 rows)

**SELECT b.book\_code, b.book\_title, i.on\_hand  
FROM books b  
INNER JOIN inventory i  
ON b.book\_code = i.book\_code  
WHERE branch\_number = 4;**

* List the book title for each book that has the type PSY and that is published by Jove Publications. (1 row)

**SELECT book\_title  
FROM book  
WHERE book\_type = 'PSY'  
AND publisher code =  
(SELECT publisher\_code FROM publishers WHERE publisher\_name = 'Jove Publications');**

* Find the book title for each book written by author number 18. Use the IN operator in your formulation. (2 rows)

**SELECT b.book\_title  
FROM books b  
INNER JOIN wrote w  
ON b.book\_code = w.book\_code  
WHERE w.author\_number IN (18);**

* Repeat exercise 6 but use the EXISTS operator. (2 rows)

**SELECT b.book\_title  
FROM books b  
WHERE EXISTS  
(Select 1 FROM wrote w  
WHERE w.book\_code = b.book\_code  
AND w.author\_number = 18);**

* Find the book codes and book title for each book located in branch number 2 and written by author 20. (1 row)

**SELECT b.book\_code, b.book\_title  
FROM books b  
INNER JOIN inventory i  
ON b.book\_code = i.book\_code  
WHERE i.branch\_number = 2  
AND b.book\_code IN  
(SELECT w.book\_code  
FROM wrote w  
WHERE w.author\_number = 20);**

* List book codes for each pair of books that have the same price. (on such pair would be book 0200 and book 7559 because the price of both books is $8.00) Sort the results by first book code and then by second book code. (11 rows)

**SELECT b1. book\_code AS book\_code1,  
 b2.book\_code AS book\_code2  
FROM book b1 , book b2  
WHERE b1.book\_code < b2.book\_code  
AND b1.book\_price = b2.book\_price**

**ORDER BY b1.book\_code, b2.book\_code**

* Find the book title, author last name, and units on hand for each book in branch number 4. (10 rows)

**SELECT b.book\_title, a.author\_last, i.on\_hand  
FROM books b  
INNER JOIN wrote w ON b.book\_code = w.book\_code  
INNER JOIN authors a ON w.author\_number = a.author\_number  
INNER JOIN inventory i ON b.book\_code = i.book\_code  
WHERE i.branch\_number = 4;**

* Repeat exercise 10 but list only paperback books. (4 rows)

**SELECT b.book\_title, a.author\_last, i.on\_hand  
FROM books b  
INNER JOIN wrote w ON b.book\_code = w.book\_code  
INNER JOIN authors a ON w.author\_number = a.author\_number  
INNER JOIN inventory i ON b.book\_code = i.book\_code  
WHERE i.branch\_number = 4;**

**AND b.book\_paperback = 'Y';**

* Find the book code and book title for each book whose price is more than $10 and that was published in Boston. (Research Union keyword. 20 rows)

**SELECT b.book\_code, b.book\_title  
FROM books b  
INNER JOIN publishers p on b.publisher\_code = p.publisher\_code  
WHERE book\_price > 10  
AND publisher\_city != 'Boston';**

* Find the book code and book title for each book whose price is more than $10 and that was not published in Boston. (18 rows)

**SELECT b.book\_code, b.book\_title  
FROM books b  
INNER JOIN publishers p on b.publisher\_code = p.publisher\_code  
WHERE book\_price > 10  
AND publisher\_city NOT IN 'Boston';**

* Find the book code and book title for each book whose price is greater than the book price for every book that has the type HOR. (5 rows)

**SELECT book\_code, book\_title  
FROM books  
WHERE book\_price > ALL  
(SELECT book\_price FROM books WHERE book\_type = 'HOR')**

* Find the book code and book title for each book whose price is greater than the price of at least one book that has the type HOR. (23 rows)

**SELECT book\_code, book\_title  
FROM books  
WHERE book\_price > ANY  
(SELECT book\_price  
FROM books  
WHERE book\_type = 'HOR')  
ORDER BY book\_price;**

* List the book code, book title, and units on hand for each book in branch number 2. Be sure each book is included, regardless of whether there are copies of the book currently on hand in branch 2. Order the output by book code. (Research Left Join: 33 rows)

**SELECT b.book\_code, b.book\_title, COALESCE(i.on\_Hand, 0) AS ONHAND  
FROM books b  
LEFT JOIN inventory i ON b.book\_code = i.book\_code  
AND i.branch\_number = 2   
ORDER BY b.book\_code;**

**LINQ EXPRESSION SYNTAX BELOW**

Book Code / Title

**var query1 = from b in books**

**select new { b.BookCode, b.BookTitle };**

Publisher  
**var query1 = from p in publishers**

**select p;**

Publisher BOSTON  
**var query1 = from p in publishers**

**where p.PublisherCity == "Boston"**

**select p.PublisherName;**

Publisher NOT BOSTON

**var query1 = from p in publishers**

**where p.PublisherCity != "Boston"**

**select p.PublisherName;**

Branches by 9  
**var query1 = from b in branches**

**where b.BranchEmps >= 9**

**select b.BranchName;**

Select book code AND title with book type SFI  
**var query1 = from b in books**

**where b.BookType == "sfi"**

**select new { b.BookCode, b.BookTitle };"**Same as previous with paperback check

**var query1 = from b in books**

**where b.BookType == "sfi" && b.BookPaperback == "Y"**

**select new { b.BookCode, b.BookTitle };**

Selecting book code and title where book type is 'sfi' or publisher code is 'sc':

**var query1 = from b in books**

**where b.BookType == "sfi" || b.PublisherCode == "sc"**

**select new { b.BookCode, b.BookTitle };**

Selecting book code, title, and price where price is between $20 and $30:

**var query1 = from b in books**

**where b.BookPrice >= 20 && b.BookPrice <= 30**

**select new { b.BookCode, b.BookTitle, b.BookPrice };**

Selecting book code, title, and price where price < $20 and type is 'MYS':

**var query1 = from b in books**

**where b.BookPrice < 20 && b.BookType == "MYS"**

**select new { b.BookCode, b.BookTitle, b.BookPrice };**

Selecting book code, title, price, and discounted price:

**var query1 = from b in books**

**select new { b.BookCode, b.BookTitle, b.BookPrice, DiscountedPrice = Math.Round(b.BookPrice \* 0.9, 2) };**

Finding the name of each publisher containing the word "and":

**var query1 = from p in publishers**

**where p.PublisherName.Contains("and")**

**select p.PublisherName;**

Selecting book code and title where book type is 'SFI', 'MYS', or 'ART':

**var query1 = from b in books**

**where new[] { "SFI", "MYS", "ART" }.Contains(b.BookType)**

**select new { b.BookCode, b.BookTitle };**

Selecting book code and title where book type is 'SFI', 'MYS', or 'ART' and ordering by title ascending:

**var query1 = from b in books**

**where new[] { "SFI", "MYS", "ART" }.Contains(b.BookType)**

**orderby b.BookTitle ascending**

**select new { b.BookCode, b.BookTitle };**

Selecting book code, title, and price where book type is 'SFI', 'MYS', or 'ART' and ordering by price descending and title ascending:

**var query1 = from b in books**

**where new[] { "SFI", "MYS", "ART" }.Contains(b.BookType)**

**orderby b.BookPrice descending, b.BookTitle ascending**

**select new { b.BookCode, b.BookTitle, b.BookPrice };**

Selecting distinct book types:

**var query1 = (from b in books**

**select b.BookType).Distinct();**

Counting books with the type 'SFI':

**var query1 = (from b in books**

**where b.BookType == "SFI"**

**select b).Count();**

Finding average price per book type:

**var query1 = from b in books**

**group b by b.BookType into g**

**select new { BookType = g.Key, AveragePrice = Math.Round(g.Average(x => x.BookPrice), 2) };**

Counting books per book type where paperback is 'Y':

**var query1 = from b in books**

**where b.BookPaperback == "Y"**

**group b by b.BookType into g**

**select new { BookType = g.Key, Count = g.Count() };**

Finding average price per book type where paperback is 'Y' and average price > 10:

**var query1 = from b in books**

**where b.BookPaperback == "Y"**

**group b by b.BookType into g**

**where g.Average(x => x.BookPrice) > 10**

**select new { BookType = g.Key, AveragePrice = g.Average(x => x.BookPrice) };**

Selecting book code, title, and price of the book(s) with the maximum price:

**var query1 = from b in books**

**where b.BookPrice == books.Max(x => x.BookPrice)**

**select new { b.BookCode, b.BookTitle, b.BookPrice };**

Selecting book title and price of the book(s) with the minimum price:

**var query12 = from b in books**

**where b.BookPrice == books.Min(x => x.BookPrice)**

**select new { b.BookTitle, b.BookPrice };**

Finding the total number of employees in all branches:

**var query13 = branches.Sum(b => b.BranchEmps);**

Joining books with publishers and selecting book code, title, publisher code, and publisher name:

**var query1 = from b in books**

**join p in publishers on b.PublisherCode equals p.PublisherCode**

**select new { b.BookCode, b.BookTitle, b.PublisherCode, p.PublisherName };**

Selecting book code, title, and price of books published by Plume with price at least $14:

**var query2 = from b in books**

**where b.PublisherCode == "PL" && b.BookPrice >= 14**

**select new { b.BookCode, b.BookTitle, b.BookPrice };**

Joining books with inventory and selecting book code, title, and on-hand quantity for branch number 4:

**var query3 = from b in books**

**join i in inventory on b.BookCode equals i.BookCode**

**where i.BranchNumber == 4**

**select new { b.BookCode, b.BookTitle, i.OnHand };**

Selecting book titles where book type is 'PSY' and publisher is 'Jove Publications':

**var query4 = from b in books**

**where b.BookType == "PSY" &&**

**(from p in publishers where p.PublisherName == "Jove Publications" select p.PublisherCode).Contains(b.PublisherCode)**

**select b.BookTitle;**

Selecting book titles where author number is 18 using inner join:

**var query5 = from b in books**

**join w in wrote on b.BookCode equals w.BookCode**

**where w.AuthorNumber == 18**

**select b.BookTitle;**

Selecting book titles where author number is 18 using EXISTS subquery:

**var query6 = from b in books**

**where wrote.Any(w => w.BookCode == b.BookCode && w.AuthorNumber == 18)**

**select b.BookTitle;**

Selecting book code and title where author number 20 and branch number 2:

**var query7 = from b in books**

**join w in wrote on b.BookCode equals w.BookCode**

**join i in inventory on b.BookCode equals i.BookCode**

**where i.BranchNumber == 2 && w.AuthorNumber == 20**

**select new { b.BookCode, b.BookTitle };**

Selecting pairs of book codes where the first book price is equal to the second book price:

**var query8 = from b1 in books**

**from b2 in books**

**where b1.BookCode < b2.BookCode && b1.BookPrice == b2.BookPrice**

**orderby b1.BookCode, b2.BookCode**

**select new { BookCode1 = b1.BookCode, BookCode2 = b2.BookCode };**

Selecting book title, author last name, and on-hand quantity where branch number is 4 using joins:

**var query9 = from b in books**

**join w in wrote on b.BookCode equals w.BookCode**

**join a in authors on w.AuthorNumber equals a.AuthorNumber**

**join i in inventory on b.BookCode equals i.BookCode**

**where i.BranchNumber == 4**

**select new { b.BookTitle, a.AuthorLast, i.OnHand };**

Selecting book title, author last name, and on-hand quantity for branch number 4 where paperback is 'Y':

**var query1 = from b in books**

**join w in wrote on b.BookCode equals w.BookCode**

**join a in authors on w.AuthorNumber equals a.AuthorNumber**

**join i in inventory on b.BookCode equals i.BookCode**

**where i.BranchNumber == 4 && b.BookPaperback == "Y"**

**select new { b.BookTitle, a.AuthorLast, i.OnHand };**

Selecting book code and title where book price is greater than $10 and publisher city is not 'Boston':

**var query2 = from b in books**

**join p in publishers on b.PublisherCode equals p.PublisherCode**

**where b.BookPrice > 10 && p.PublisherCity != "Boston"**

**select new { b.BookCode, b.BookTitle };**

Selecting book code and title where book price is greater than $10 and publisher city is not 'Boston' (alternative):

**var query3 = from b in books**

**join p in publishers on b.PublisherCode equals p.PublisherCode**

**where b.BookPrice > 10 && !p.PublisherCity.Contains("Boston")**

**select new { b.BookCode, b.BookTitle };**

Selecting book code and title where book price is greater than all prices of books with type 'HOR':

**var maxPriceHOR = (from b in books where b.BookType == "HOR" select b.BookPrice).Max();**

**var query4 = from b in books**

**where b.BookPrice > maxPriceHOR**

**select new { b.BookCode, b.BookTitle };**

Selecting book code and title where book price is greater than any price of books with type 'HOR', ordered by book price:

**var minPriceHOR = (from b in books where b.BookType == "HOR" select b.BookPrice).Min();**

**var query5 = from b in books**

**where b.BookPrice > minPriceHOR**

**orderby b.BookPrice**

**select new { b.BookCode, b.BookTitle };**

Selecting book code, title, and on-hand quantity with a left join to inventory for branch number 2:

**var query6 = from b in books**

**join i in inventory on b.BookCode equals i.BookCode into iGroup**

**from i in iGroup.DefaultIfEmpty()**

**where (i == null || i.BranchNumber == 2)**

**orderby b.BookCode**

**select new { b.BookCode, b.BookTitle, OnHand = (i != null ? i.OnHand : 0)**

**LINQ METHOD SYNTAX BELOW**

Selecting book code and title from books:

**var query1 = books.Select(b => new { b.BookCode, b.BookTitle });**

Selecting all publishers:

**var query2 = publishers.ToList();**

Selecting publisher names from Boston:

**var query3 = publishers.Where(p => p.PublisherCity == "Boston")**

**.Select(p => p.PublisherName);**

Selecting publisher names not from Boston:

**var query4 = publishers.Where(p => p.PublisherCity != "Boston")**

**.Select(p => p.PublisherName);**

Selecting branch names with employees >= 9:

**var query5 = branches.Where(b => b.BranchEmps >= 9)**

**.Select(b => b.BranchName);**

Selecting book code and title where book type is 'sfi':

**var query6 = books.Where(b => b.BookType == "sfi")**

**.Select(b => new { b.BookCode, b.BookTitle });**

Selecting book code and title where book type is 'sfi' and paperback is 'Y':

**var query7 = books.Where(b => b.BookType == "sfi" && b.BookPaperback == "Y")**

**.Select(b => new { b.BookCode, b.BookTitle });**

Selecting book code and title where book type is 'sfi' or publisher code is 'sc':

**var query8 = books.Where(b => b.BookType == "sfi" || b.PublisherCode == "sc")**

**.Select(b => new { b.BookCode, b.BookTitle });**

Selecting book code, title, and price where price is between $20 and $30:

**var query9 = books.Where(b => b.BookPrice >= 20 && b.BookPrice <= 30)**

**.Select(b => new { b.BookCode, b.BookTitle, b.BookPrice });**

Selecting book code, title, and price where price is less than $20 and type is 'MYS':

**var query10 = books.Where(b => b.BookPrice < 20 && b.BookType == "MYS")**

**.Select(b => new { b.BookCode, b.BookTitle, b.BookPrice });**

Selecting book code, title, price, and discounted price  
**var query = from b in books**

**select new {**

**b.BookCode,**

**b.BookTitle,**

**b.BookPrice,**

**DiscountedPrice = Math.Round(b.BookPrice \* 0.9, 2)**

**};**

**query = query.Union(**

**from p in publishers**

**where p.PublisherName.Contains("and")**

**select new {**

**PublisherName = p.PublisherName**

**}**

**);**

Selecting book code and title where book type is 'SFI', 'MYS', or 'ART':

**var query3 = books.Where(b => new[] { "SFI", "MYS", "ART" }.Contains(b.BookType))**

**.Select(b => new { b.BookCode, b.BookTitle });**

Selecting book code and title where book type is 'SFI', 'MYS', or 'ART' and ordering by title ascending:

**var query4 = books.Where(b => new[] { "SFI", "MYS", "ART" }.Contains(b.BookType))**

**.OrderBy(b => b.BookTitle)**

**.Select(b => new { b.BookCode, b.BookTitle });**

Selecting book code, title, and price where book type is 'SFI', 'MYS', or 'ART' and ordering by price descending then title ascending:

**var query5 = books.Where(b => new[] { "SFI", "MYS", "ART" }.Contains(b.BookType))**

**.OrderByDescending(b => b.BookPrice)**

**.ThenBy(b => b.BookTitle)**

**.Select(b => new { b.BookCode, b.BookTitle, b.BookPrice });**

Selecting distinct book types:

**var distinctBookTypes = books.Select(b => b.BookType).Distinct();**

Counting the number of books with the type 'SFI':

**var countSFIBooks = books.Count(b => b.BookType == "SFI");**

Finding the average price per book type:

**var averagePricePerType = books.GroupBy(b => b.BookType)**

**.Select(g => new {**

**BookType = g.Key,**

**AveragePrice = Math.Round(g.Average(b => b.BookPrice), 2)**

**});**

Counting books per book type where paperback is 'Y':

**var countPaperbackBooksPerType = books.Where(b => b.BookPaperback == "Y")**

**.GroupBy(b => b.BookType)**

**.Select(g => new {**

**BookType = g.Key,**

**Count = g.Count()**

**});**

Finding the average price per book type where paperback is 'Y' and average price > 10:

**var query1 = books.Where(b => b.BookPaperback == "Y")**

**.GroupBy(b => b.BookType)**

**.Where(g => g.Average(b => b.BookPrice) > 10)**

**.Select(g => new {**

**BookType = g.Key,**

**AveragePrice = g.Average(b => b.BookPrice)**

**});**

Selecting book code, title, and price of the book(s) with the maximum price:

**var maxPrice = books.Max(b => b.BookPrice);**

**var query2 = books.Where(b => b.BookPrice == maxPrice)**

**.Select(b => new {**

**b.BookCode,**

**b.BookTitle,**

**b.BookPrice**

**});**

Selecting book title and price of the book(s) with the minimum price:

**var minPrice = books.Min(b => b.BookPrice);**

**var query3 = books.Where(b => b.BookPrice == minPrice)**

**.Select(b => new {**

**b.BookTitle,**

**b.BookPrice**

**});**

Finding the total number of employees across all branches:

**var totalEmployees = branches.Sum(b => b.BranchEmps);**

Joining books with publishers and selecting book code, title, publisher code, and publisher name:

**var query5 = books.Join(publishers,**

**b => b.PublisherCode,**

**p => p.PublisherCode,**

**(b, p) => new {**

**b.BookCode,**

**b.BookTitle,**

**b.PublisherCode,**

**p.PublisherName**

**});**

Selecting book code, title, and price of books published by 'PL':

**var query1 = books.Where(b => b.PublisherCode == "PL")**

**.Select(b => new {**

**b.BookCode,**

**b.BookTitle,**

**b.BookPrice**

**});**

Selecting book code, title, and price of books published by 'PL' with price at least 14:

**var query2 = books.Where(b => b.PublisherCode == "PL" && b.BookPrice >= 14)**

**.Select(b => new {**

**b.BookCode,**

**b.BookTitle,**

**b.BookPrice**

**});**

Selecting book code, title, and on-hand quantity from inventory for branch number 4:

**var query3 = books.Join(inventory,**

**b => b.BookCode,**

**i => i.BookCode,**

**(b, i) => new {**

**b.BookCode,**

**b.BookTitle,**

**i.OnHand**

**})**

**.Where(j => j.OnHand == 4);**

Selecting book titles where book type is 'PSY' and publisher name is 'Jove Publications':

**var query4 = books.Where(b => b.BookType == "PSY" && b.Publisher.PublisherName == "Jove Publications")**

**.Select(b => b.BookTitle);**

Selecting book titles where author number is 18 using an inner join:

**var query5 = books.Join(wrote,**

**b => b.BookCode,**

**w => w.BookCode,**

**(b, w) => new {**

**b.BookCode,**

**b.BookTitle,**

**w.AuthorNumber**

**})**

**.Where(j => j.AuthorNumber == 18)**

**.Select(j => j.BookTitle);**

Selecting book titles where author number is 18 using EXISTS:

**var query6 = books.Where(b => wrote.Any(w => w.BookCode == b.BookCode && w.AuthorNumber == 18))**

**.Select(b => b.BookTitle);**

Selecting book code and title from books where branch number is 2 and the book is written by author number 20:

**var query1 = books.Join(inventory,**

**b => b.BookCode,**

**i => i.BookCode,**

**(b, i) => new {**

**b.BookCode,**

**b.BookTitle,**

**i.BranchNumber**

**})**

**.Where(j => j.BranchNumber == 2 && wrote.Any(w => w.BookCode == j.BookCode && w.AuthorNumber == 20))**

**.Select(j => new {**

**j.BookCode,**

**j.BookTitle**

**});**

Selecting pairs of book codes where the first book's price is less than the second book's price:

**var query2 = from b1 in books**

**from b2 in books**

**where b1.BookCode < b2.BookCode && b1.BookPrice == b2.BookPrice**

**orderby b1.BookCode, b2.BookCode**

**select new {**

**BookCode1 = b1.BookCode,**

**BookCode2 = b2.BookCode**

**};**

Selecting book title, author last name, and on-hand quantity from inventory for branch number 4:

**var query3 = books.Join(wrote,**

**b => b.BookCode,**

**w => w.BookCode,**

**(b, w) => new {**

**b.BookTitle,**

**w.Author.AuthorLast,**

**b.Inventory.OnHand**

**})**

**.Where(j => j.Inventory.BranchNumber == 4)**

**.Select(j => new {**

**j.BookTitle,**

**j.AuthorLast,**

**j.OnHand**

**});**

Selecting book title, author last name, and on-hand quantity from inventory for branch number 4 and where book is paperback:

**var query4 = books.Join(wrote,**

**b => b.BookCode,**

**w => w.BookCode,**

**(b, w) => new {**

**b.BookTitle,**

**w.Author.AuthorLast,**

**b.Inventory.OnHand**

**})**

**.Where(j => j.Inventory.BranchNumber == 4 && j.Book.BookPaperback == "Y")**

**.Select(j => new {**

**j.BookTitle,**

**j.AuthorLast,**

**j.OnHand**

**});**

Selecting book code and title from books where the price is greater than 10 and publisher city is not 'Boston':

**var query5 = books.Join(publishers,**

**b => b.PublisherCode,**

**p => p.PublisherCode,**

**(b, p) => new {**

**b.BookCode,**

**b.BookTitle,**

**b.BookPrice,**

**p.PublisherCity**

**})**

**.Where(j => j.BookPrice > 10 && j.PublisherCity != "Boston")**

**.Select(j => new {**

**j.BookCode,**

**j.BookTitle**

**});**

Selecting book code and title from books where the price is greater than 10 and publisher city is not 'Boston':

**var query6 = books.Join(publishers,**

**b => b.PublisherCode,**

**p => p.PublisherCode,**

**(b, p) => new {**

**b.BookCode,**

**b.BookTitle,**

**b.BookPrice,**

**p.PublisherCity**

**})**

**.Where(j => j.BookPrice > 10 && !j.PublisherCity.Contains("Boston"))**

**.Select(j => new {**

**j.BookCode,**

**j.BookTitle**

**});**

Selecting book code and title from books where the price is greater than all prices of books with type 'HOR':

**var query1 = books.Where(b => b.BookPrice > books.Where(b => b.BookType == "HOR").Max(b => b.BookPrice))**

**.Select(b => new {**

**b.BookCode,**

**b.BookTitle**

**});**

Selecting book code and title from books where the price is greater than any price of books with type 'HOR', ordered by price:

**var query2 = books.Where(b => b.BookPrice > books.Where(b => b.BookType == "HOR").Any() ? books.Where(b => b.BookType == "HOR").Min(b => b.BookPrice) : 0)**

**.OrderBy(b => b.BookPrice)**

**.Select(b => new {**

**b.BookCode,**

**b.BookTitle**

**});**

Selecting book code, title, and on-hand quantity from inventory for branch number 2, with a left join:

**var query3 = books.GroupJoin(inventory.Where(i => i.BranchNumber == 2),**

**b => b.BookCode,**

**i => i.BookCode,**

**(b, i) => new {**

**b.BookCode,**

**b.BookTitle,**

**OnHand = i.Any() ? i.Sum(i => i.OnHand) : 0**

**})**

**.OrderBy(b => b.BookCode)**

**.Select(b => new {**

**b.BookCode,**

**b.BookTitle,**

**b.OnHand**

**});**