SQL

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

SELECT TOP (33) BOOK\_CODE, BOOK\_TITLE

FROM Books;

2. List the complete PUBLISHER table. (28 rows)

SELECT TOP (28) \*

FROM Publishers;

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

SELECT TOP (2) PUBLISHER\_NAME

FROM Publishers

WHERE PUBLISHER\_CITY LIKE ‘%Boston%’;

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

SELECT TOP (26) PUBLISHER\_NAME

FROM Publishers

WHERE PUBLISHER\_CITY NOT LIKE ‘%Boston%’;

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

SELECT TOP (3) BRANCH\_NAME

FROM Branches

WHERE BRANCH\_EMPS >= 9;

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

SELECT TOP (3) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_TYPE = ‘SFI’;

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

SELECT TOP (1) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_TYPE = ‘SFI’ AND BOOK\_PAPERBACK = ‘Y’;

8. 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 TOP (5) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_TYPE = ‘SFI’ OR PUBLISHER\_CODE = ‘SC’;

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

SELECT TOP (4) BOOK\_CODE, BOOK\_TITLE, BOOK\_PRICE

FROM Books

WHERE BOOK\_PRICE BETWEEN 20 AND 30;

10. 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 TOP (2) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_TYPE = ‘MYS’ AND BOOK\_PRICE < 20;

11. 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 TOP (33) BOOK\_CODE, BOOK\_TITLE, BOOK\_PRICE \* 0.90 AS discounted\_price

FROM Books;

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

SELECT TOP (4) PUBLISHER\_NAME

FROM Publishers

WHERE PUBLISHER\_NAME LIKE ‘%and%’;

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

SELECT TOP (9) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_TYPE IN (‘SFI’, ‘MYS’, ‘ART’);

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

SELECT TOP (9) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_TYPE IN (‘SFI’, ‘MYS’, ‘ART’)

ORDER BY BOOK\_TITLE;

15. 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 TOP (9) BOOK\_CODE, BOOK\_TITLE, BOOK\_PRICE

FROM Books

WHERE BOOK\_TYPE IN (‘SFI’, ‘MYS’, ‘ART’)

ORDER BY BOOK\_PRICE DESC, BOOK\_TITLE ASC;

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

SELECT DISTINCT TOP (12) BOOK\_TYPE

FROM Books;

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

SELECT COUNT(\*) AS NumberOfBooks

FROM Books

WHERE BOOK\_TYPE = ‘SFI’

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

SELECT TOP (12) BOOK\_TYPE, AVG(BOOK\_PRICE) AS AveragePrice

FROM Books

GROUP BY BOOK\_TYPE;

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

SELECT TOP (10) BOOK\_TYPE, AVG(BOOK\_PRICE) As AveragePrice

FROM Books

WHERE BOOK\_PAPERBACK = ‘Y’

GROUP BY BOOK\_TYPE;

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

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

SELECT TOP (1) BOOK\_CODE, BOOK\_TITLE, BOOK\_PRICE

FROM Books

WHERE BOOK\_PRICE = (SELECT MAX(BOOK\_PRICE) FROM Books);

22. What are the title(s)/price(s) of the least expensive book(s) in the DB? (3 rows)

SELECT TOP (3) BOOK\_TITLE, BOOK\_PRICE

FROM Books

WHERE BOOK\_PRICE = (SELECT MIN(BOOK\_PRICE) FROM Books)

ORDER BY BOOK\_TITLE;

23. How many employees does Henry Books have?

SELECT SUM(BRANCH\_EMPS) AS TotalEmployees

FROM Branches;

Multiple Table

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

SELECT TOP (33) b.BOOK\_CODE, b.BOOK\_TITLE, b.PUBLISHER\_CODE, p.PUBLISHER\_NAME

FROM Books b

JOIN Publishers p ON b.PUBLISHER\_CODE = p.PUBLISHER\_CODE

ORDER BY p.PUBLISHER\_NAME;

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

SELECT TOP (3) BOOK\_CODE, BOOK\_TITLE, BOOK\_PRICE

FROM Books

WHERE PUBLISHER\_CODE = ‘PL’;

3. 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 TOP (1) BOOK\_TITLE, BOOK\_CODE, BOOK\_PRICE

FROM Books

WHERE PUBLISHER\_CODE = ‘PL’ AND BOOK\_PRICE >= 14;

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

SELECT TOP (9) b.BOOK\_CODE, b.BOOK\_TITLE, i.ON\_HAND

FROM Books b

JOIN Inventory i ON b.BOOK\_CODE = i.BOOK\_CODE

WHERE i.BRANCH\_NUMBER = 4;

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

SELECT TOP (1) BOOK\_TITLE

FROM Books

WHERE BOOK\_TYPE = ‘PSY’ AND PUBLISHER\_CODE = ‘JP’;

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

SELECT TOP (2) BOOK\_TITLE

FROM Books

WHERE BOOK\_CODE IN (SELECT BOOK\_CODE, FROM Wrote WHERE AUTHOR\_NUMBER = 18);

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

SELECT TOP (2) BOOK\_TITLE

FROM Books b

WHERE EXISTS (

SELECT 1

FROM Wrote w

WHERE w.BOOK\_CODE = b.BOOK\_CODE AND w.AUTHOR\_NUMBER = 18

);

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

SELECT TOP (1) b.BOOK\_CODE, b.BOOK\_TITLE

FROM Books b

JOIN Inventory i ON b.BOOK\_CODE = i.BOOK\_CODE

JOIN Wrote w ON b.BOOK\_CODE = w.BOOK\_CODE

WHERE i.BRANCH\_NUMBER = 2 AND w.AUTHOR\_NUMBER = 20;

9. 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 TOP (11) a.BOOK\_CODE AS BookCode1, b.BOOK\_CODE AS BookCode2

FROM Books a, Books b

WHERE a.BOOK\_PRICE = b.BOOK\_PRICE AND a.BOOK\_CODE < b.BOOK\_CODE

ORDER BY a.BOOK\_CODE, b.BOOK\_CODE;

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

SELECT TOP (10) b.BOOK\_TITLE, a.AUTHOR\_LAST, i.ON\_HAND

FROM Books b

JOIN Inventory i ON b.BOOK\_CODE = i.BOOK\_CODE

JOIN Wrote w ON b.BOOK\_CODE = w.BOOK\_CODE

JOIN Authors a ON w.AUTHOR\_NUMBER = a.AUTHOR\_NUMBER

WHERE i.BRANCH\_NUMBER = 4;

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

SELECT TOP (4) b.BOOK\_TITLE, a.AUTHOR\_LAST, i.ON\_HAND

FROM Books b

JOIN Inventory i ON b.BOOK\_CODE = w.BOOK\_CODE

JOIN Wrote w ON b.BOOK\_CODE = w.BOOK\_CODE

JOIN Authors a ON w.AUTHOR\_NUMBER = a.AUTHOR\_NUMBER

WHERE i.BRANCH\_NUMBER = 4 AND b.BOOK\_PAPERBACK = ‘Y’;

12. 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 TOP (20) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_PRICE > 10 AND PUBLISHER\_CODE IN (SELECT PUBLISHER\_CODE FROM Publishers WHERE PUBLISHER\_CITY LIKE ‘%Boston%’);

13. 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 TOP (18) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_PRICE > AND PUBLISHER\_CODE NOT IN (SELECT PUBLISHER\_CODE FROM Publishers WHERE PUBLISHER\_CITY LIKE ‘%Boston%’);

14. 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 TOP (5) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_PRICE > ALL (SELECT BOOK\_PRICE FROM Books WHERE BOOK\_TYPE = ‘HOR’);

15. 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 TOP (23) BOOK\_CODE, BOOK\_TITLE

FROM Books

WHERE BOOK\_PRICE > ANY (SELECT BOOK\_PRICE FROM Books WHERE BOOK\_TYPE = ‘HOR’);

16. 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 TOP (33) b.BOOK\_CODE, b.BOOK\_TITLE, COALESCE(i.ON\_HAND, 0) AS ON\_HAND

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

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

var query1 = (from book in context.Books

select new { book.BookCode, book.BookTitle }).Take(33);

2. List the complete PUBLISHER table. (28 rows)

var query2 = (from publisher in context.Publishers

select publisher).Take(28);

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

var query3 = (from publisher in context.Publishers

where publisher.PublisherCity.Contains("Boston")

select publisher.PublisherName).Take(2);

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

var query4 = (from publisher in context.Publishers

where !publisher.PublisherCity.Contains("Boston")

select publisher.PublisherName).Take(26);

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

var query5 = (from branch in context.Branches

where branch.BranchEmps >= 9

select branch.BranchName).Take(3);

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

var query6 = (from book in context.Books

where book.BookType == "SFI"

select new { book.BookCode, book.BookTitle }).Take(3);

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

var query7 = (from book in context.Books

where book.BookType == "SFI" && book.BookPaperback == "Y"

select new { book.BookCode, book.BookTitle }).Take(1);

8. 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)

var query8 = (from book in context.Books

where book.BookType == "SFI" || book.PublisherCode == "SC"

select new { book.BookCode, book.BookTitle }).Take(5);

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

var query9 = (from book in context.Books

where book.BookPrice >= 20 && book.BookPrice <= 30

select new { book.BookCode, book.BookTitle, book.BookPrice }).Take(4);

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

var query10 = (from book in context.Books

where book.BookType == "MYS" && book.BookPrice < 20

select new { book.BookCode, book.BookTitle }).Take(2);

11. 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)

var query11 = (from book in context.Books

select new {

book.BookCode,

book.BookTitle,

DiscountedPrice = book.BookPrice \* 0.90M

}).Take(33);

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

var query12 = (from publisher in context.Publishers

where publisher.PublisherName.Contains("and")

select publisher.PublisherName).Take(4);

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

var query13 = (from book in context.Books

where book.BookType == "SFI" || book.BookType == "MYS" || book.BookType == "ART"

select new { book.BookCode, book.BookTitle }).Take(9);

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

var query14 = (from book in context.Books

where book.BookType == "SFI" || book.BookType == "MYS" || book.BookType == "ART"

orderby book.BookTitle

select new { book.BookCode, book.BookTitle }).Take(9);

15. 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)

var query15 = (from book in context.Books

where book.BookType == "SFI" || book.BookType == "MYS" || book.BookType == "ART"

orderby book.BookPrice descending, book.BookTitle

select new { book.BookCode, book.BookTitle, book.BookPrice }).Take(9);

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

var query16 = (from book in context.Books

select book.BookType).Distinct().Take(12);

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

var query17 = (from book in context.Books

where book.BookType == "SFI"

select book).Count();

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

var query18 = (from book in context.Books

group book by book.BookType into bookGroup

select new {

BookType = bookGroup.Key,

AveragePrice = bookGroup.Average(b => b.BookPrice)

}).Take(12);

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

var query19 = (from book in context.Books

where book.BookPaperback == "Y"

group book by book.BookType into bookGroup

select new {

BookType = bookGroup.Key,

AveragePrice = bookGroup.Average(b => b.BookPrice)

}).Take(10);

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

var query20 = (from book in context.Books

where book.BookPaperback == "Y"

group book by book.BookType into bookGroup

let avgPrice = bookGroup.Average(b => b.BookPrice)

where avgPrice > 10

select new {

BookType = bookGroup.Key,

AveragePrice = avgPrice

}).Take(3);

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

var maxPrice = context.Books.Max(b => b.BookPrice);

var query21 = (from book in context.Books

where book.BookPrice == maxPrice

select new { book.BookCode, book.BookTitle, book.BookPrice }).Take(1);

22. What are the title(s)/price(s) of the least expensive book(s) in the DB? (3 rows)

var minPrice = context.Books.Min(b => b.BookPrice);

var query22 = (from book in context.Books

where book.BookPrice == minPrice

select new { book.BookTitle, book.BookPrice }).Take(3);

23. How many employees does Henry Books have?

var query23 = (from branch in context.Branches

select branch.BranchEmps).Sum();

Multiple Table

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

var query1 = (from book in context.Books

join publisher in context.Publishers

on book.PublisherCode equals publisher.PublisherCode

orderby publisher.PublisherName

select new {

book.BookCode,

book.BookTitle,

book.PublisherCode,

PublisherName = publisher.PublisherName

}).Take(33);

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

var query2 = (from book in context.Books

where book.PublisherCode == "PL"

select new {

book.BookCode,

book.BookTitle,

book.BookPrice

}).Take(3);

3. 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)

var query3 = (from book in context.Books

where book.PublisherCode == "PL" && book.BookPrice >= 14

select new {

book.BookTitle,

book.BookCode,

book.BookPrice

}).Take(1);

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

var query4 = (from book in context.Books

join inventory in context.Inventory

on book.BookCode equals inventory.BookCode

where inventory.BranchNumber == 4

select new {

book.BookCode,

book.BookTitle,

inventory.OnHand

}).Take(9);

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

var query5 = (from book in context.Books

where book.BookType == "PSY" && book.PublisherCode == "JP"

select book.BookTitle).Take(1);

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

var query6 = (from book in context.Books

where (from wrote in context.Wrote

where wrote.AuthorNumber == 18

select wrote.BookCode).Contains(book.BookCode)

select book.BookTitle).Take(2);

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

var query7 = (from book in context.Books

where context.Wrote.Any(wrote => wrote.BookCode == book.BookCode && wrote.AuthorNumber == 18)

select book.BookTitle).Take(2);

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

var query8 = (from book in context.Books

join inventory in context.Inventory

on book.BookCode equals inventory.BookCode

where inventory.BranchNumber == 2

join wrote in context.Wrote

on book.BookCode equals wrote.BookCode

where wrote.AuthorNumber == 20

select new {

book.BookCode,

book.BookTitle

}).Take(1);

9. 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)

var query9 = (from book1 in context.Books

from book2 in context.Books

where book1.BookPrice == book2.BookPrice && book1.BookCode < book2.BookCode

select new {

BookCode1 = book1.BookCode,

BookCode2 = book2.BookCode

}).Take(11);

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

var query10 = (from book in context.Books

join inventory in context.Inventory

on book.BookCode equals inventory.BookCode

where inventory.BranchNumber == 4

join wrote in context.Wrote

on book.BookCode equals wrote.BookCode

join author in context.Authors

on wrote.AuthorNumber equals author.AuthorNumber

select new {

book.BookTitle,

AuthorLastName = author.AuthorLast,

inventory.OnHand

}).Take(10);

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

var query11 = (from book in context.Books

where book.BookPaperback == 'Y'

join inventory in context.Inventory

on book.BookCode equals inventory.BookCode

where inventory.BranchNumber == 4

join wrote in context.Wrote

on book.BookCode equals wrote.BookCode

join author in context.Authors

on wrote.AuthorNumber equals author.AuthorNumber

select new {

book.BookTitle,

AuthorLastName = author.AuthorLast,

inventory.OnHand

}).Take(4);

12. 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)

var query12 = (from book in context.Books

join publisher in context.Publishers

on book.PublisherCode equals publisher.PublisherCode

where book.BookPrice > 10 && publisher.PublisherCity.Contains("Boston")

select new {

book.BookCode,

book.BookTitle

}).Take(20);

13. 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)

var query13 = (from book in context.Books

join publisher in context.Publishers

on book.PublisherCode equals publisher.PublisherCode

where book.BookPrice > 10 && !publisher.PublisherCity.Contains("Boston")

select new {

book.BookCode,

book.BookTitle

}).Take(18);

14. 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)

var highestHorPrice = context.Books.Where(b => b.BookType == "HOR").Max(b => b.BookPrice);

var query14 = (from book in context.Books

where book.BookPrice > highestHorPrice

select new {

book.BookCode,

book.BookTitle

}).Take(5);

15. 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)

var horPrices = context.Books.Where(b => b.BookType == "HOR").Select(b => b.BookPrice);

var query15 = (from book in context.Books

where horPrices.Any(price => book.BookPrice > price)

select new {

book.BookCode,

book.BookTitle

}).Take(23);

16. 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)

var query16 = (from book in context.Books

join inventory in context.Inventory

on book.BookCode equals inventory.BookCode into inventoryGroup

from subInventory in inventoryGroup.DefaultIfEmpty()

where subInventory.BranchNumber == 2 || subInventory == null

orderby book.BookCode

select new {

book.BookCode,

book.BookTitle,

OnHand = (int?)subInventory.OnHand ?? 0

}).Take(33);

LINQ Method Syntax

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

var query1 = context.Books

.Select(book => new { book.BookCode, book.BookTitle })

.Take(33);

2. List the complete PUBLISHER table. (28 rows)

var query2 = context.Publishers

.Take(28);

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

var query3 = context.Publishers

.Where(p => p.PublisherCity.Contains("Boston"))

.Select(p => p.PublisherName)

.Take(2);

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

var query4 = context.Publishers

.Where(p => !p.PublisherCity.Contains("Boston"))

.Select(p => p.PublisherName)

.Take(26);

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

var query5 = context.Branches

.Where(branch => branch.BranchEmps >= 9)

.Select(branch => branch.BranchName)

.Take(3);

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

var query6 = context.Books

.Where(book => book.BookType == "SFI")

.Select(book => new { book.BookCode, book.BookTitle })

.Take(3);

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

var query7 = context.Books

.Where(book => book.BookType == "SFI" && book.BookPaperback == "Y")

.Select(book => new { book.BookCode, book.BookTitle })

.Take(1);

8. 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)

var query8 = context.Books

.Where(book => book.BookType == "SFI" || book.PublisherCode == "SC")

.Select(book => new { book.BookCode, book.BookTitle })

.Take(5);

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

var query9 = context.Books

.Where(book => book.BookPrice >= 20 && book.BookPrice <= 30)

.Select(book => new { book.BookCode, book.BookTitle, book.BookPrice })

.Take(4);

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

var query10 = context.Books

.Where(book => book.BookType == "MYS" && book.BookPrice < 20)

.Select(book => new { book.BookCode, book.BookTitle })

.Take(2);

11. 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)

var query11 = context.Books

.Select(book => new {

book.BookCode,

book.BookTitle,

DiscountedPrice = book.BookPrice \* 0.90M

})

.Take(33);

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

var query12 = context.Publishers

.Where(p => p.PublisherName.Contains("and"))

.Select(p => p.PublisherName)

.Take(4);

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

var query13 = context.Books

.Where(book => book.BookType == "SFI" || book.BookType == "MYS" || book.BookType == "ART")

.Select(book => new { book.BookCode, book.BookTitle })

.Take(9);

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

var query14 = context.Books

.Where(book => book.BookType == "SFI" || book.BookType == "MYS" || book.BookType == "ART")

.OrderBy(book => book.BookTitle)

.Select(book => new { book.BookCode, book.BookTitle })

.Take(9);

15. 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)

var query15 = context.Books

.Where(book => book.BookType == "SFI" || book.BookType == "MYS" || book.BookType == "ART")

.OrderByDescending(book => book.BookPrice).ThenBy(book => book.BookTitle)

.Select(book => new { book.BookCode, book.BookTitle, book.BookPrice })

.Take(9);

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

var query16 = context.Books

.Select(book => book.BookType)

.Distinct()

.Take(12);

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

var query17 = context.Books

.Count(book => book.BookType == "SFI");

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

var query18 = context.Books

.GroupBy(book => book.BookType)

.Select(group => new {

BookType = group.Key,

AveragePrice = group.Average(book => book.BookPrice)

})

.Take(12);

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

var query19 = context.Books

.Where(book => book.BookPaperback == "Y")

.GroupBy(book => book.BookType)

.Select(group => new {

BookType = group.Key,

AveragePrice = group.Average(book => book.BookPrice)

})

.Take(10);

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

var query20 = context.Books

.Where(book => book.BookPaperback == "Y")

.GroupBy(book => book.BookType)

.Select(group => new {

BookType = group.Key,

AveragePrice = group.Average(book => book.BookPrice)

})

.Where(group => group.AveragePrice > 10)

.Take(3);

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

var maxPrice = context.Books.Max(book => book.BookPrice);

var query21 = context.Books

.Where(book => book.BookPrice == maxPrice)

.Select(book => new {

book.BookCode,

book.BookTitle,

book.BookPrice

})

.Take(1);

22. What are the title(s)/price(s) of the least expensive book(s) in the DB? (3 rows)

var minPrice = context.Books.Min(book => book.BookPrice);

var query22 = context.Books

.Where(book => book.BookPrice == minPrice)

.Select(book => new {

book.BookTitle,

book.BookPrice

})

.Take(3);

23. How many employees does Henry Books have?

var query23 = context.Branches.Sum(branch => branch.BranchEmps);

Multiple Table

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

var query1 = context.Books

.Join(context.Publishers,

book => book.PublisherCode,

publisher => publisher.PublisherCode,

(book, publisher) => new { book.BookCode, book.BookTitle, book.PublisherCode, PublisherName = publisher.PublisherName })

.OrderBy(x => x.PublisherName)

.Take(33);

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

var query2 = context.Books

.Where(book => book.PublisherCode == "PL")

.Select(book => new { book.BookCode, book.BookTitle, book.BookPrice })

.Take(3);

3. 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)

var query3 = context.Books

.Where(book => book.PublisherCode == "PL" && book.BookPrice >= 14)

.Select(book => new { book.BookTitle, book.BookCode, book.BookPrice })

.Take(1);

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

var query4 = context.Books

.Join(context.Inventory,

book => book.BookCode,

inventory => inventory.BookCode,

(book, inventory) => new { book.BookCode, book.BookTitle, inventory.OnHand, inventory.BranchNumber })

.Where(x => x.BranchNumber == 4)

.Take(9);

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

var query5 = context.Books

.Where(book => book.BookType == "PSY" && book.PublisherCode == "JP")

.Select(book => book.BookTitle)

.Take(1);

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

var query6 = context.Books

.Where(book => context.Wrote

.Where(wrote => wrote.AuthorNumber == 18)

.Select(wrote => wrote.BookCode)

.Contains(book.BookCode))

.Select(book => book.BookTitle)

.Take(2);

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

var query7 = context.Books

.Where(book => context.Wrote.Any(wrote => wrote.BookCode == book.BookCode && wrote.AuthorNumber == 18))

.Select(book => book.BookTitle)

.Take(2);

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

var query8 = context.Books

.Join(context.Inventory,

book => book.BookCode,

inventory => inventory.BookCode,

(book, inventory) => new { book, inventory })

.Where(x => x.inventory.BranchNumber == 2)

.Join(context.Wrote,

x => x.book.BookCode,

wrote => wrote.BookCode,

(x, wrote) => new { x.book.BookCode, x.book.BookTitle, wrote.AuthorNumber })

.Where(x => x.AuthorNumber == 20)

.Take(1);

9. 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)

var query9 = context.Books

.SelectMany(book1 => context.Books,

(book1, book2) => new { book1, book2 })

.Where(x => x.book1.BookPrice == x.book2.BookPrice && x.book1.BookCode < x.book2.BookCode)

.Select(x => new { BookCode1 = x.book1.BookCode, BookCode2 = x.book2.BookCode })

.OrderBy(x => x.BookCode1).ThenBy(x => x.BookCode2)

.Take(11);

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

var query10 = context.Books

.Join(context.Inventory,

book => book.BookCode,

inventory => inventory.BookCode,

(book, inventory) => new { book, inventory })

.Where(x => x.inventory.BranchNumber == 4)

.SelectMany(

x => x.book.Wrotes,

(x, wrote) => new {

x.book.BookTitle,

AuthorLastName = wrote.Author.AuthorLast,

x.inventory.OnHand

})

.Take(10);

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

var query11 = context.Books

.Where(book => book.BookPaperback == "Y")

.Join(context.Inventory,

book => book.BookCode,

inventory => inventory.BookCode,

(book, inventory) => new { book, inventory })

.Where(x => x.inventory.BranchNumber == 4)

.SelectMany(

x => x.book.Wrotes,

(x, wrote) => new {

x.book.BookTitle,

AuthorLastName = wrote.Author.AuthorLast,

x.inventory.OnHand

})

.Take(4);

12. 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)

var query12 = context.Books

.Where(book => book.BookPrice > 10 && book.Publisher.PublisherCity.Contains("Boston"))

.Select(book => new { book.BookCode, book.BookTitle })

.Take(20);

13. 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)

var query13 = context.Books

.Where(book => book.BookPrice > 10 && !book.Publisher.PublisherCity.Contains("Boston"))

.Select(book => new { book.BookCode, book.BookTitle })

.Take(18);

14. 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)

var highestHorPrice = context.Books.Where(b => b.BookType == "HOR").Max(b => b.BookPrice);

var query14 = context.Books

.Where(book => book.BookPrice > highestHorPrice)

.Select(book => new { book.BookCode, book.BookTitle })

.Take(5);

15. 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)

var horPrices = context.Books.Where(b => b.BookType == "HOR").Select(b => b.BookPrice).ToList();

var query15 = context.Books

.Where(book => horPrices.Any(price => book.BookPrice > price))

.Select(book => new { book.BookCode, book.BookTitle })

.Take(23);

16. 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)

var query16 = context.Inventory

.Where(i => i.BranchNumber == 2)

.GroupJoin(context.Books,

inventory => inventory.BookCode,

book => book.BookCode,

(inventory, books) => new { Inventory = inventory, Books = books })

.SelectMany(

x => x.Books.DefaultIfEmpty(),

(x, book) => new {

BookCode = x.Inventory.BookCode,

BookTitle = book != null ? book.BookTitle : "No title available",

OnHand = x.Inventory.OnHand

})

.OrderBy(x => x.BookCode)

.Take(33);