**Assignment 4 SQL**

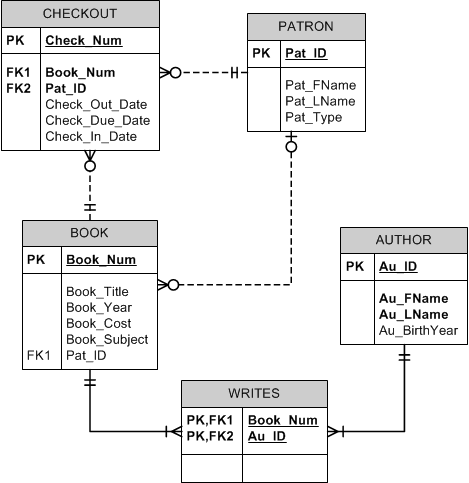
Total points: 50

*All problems worth 6 pts unless otherwise specified.*

**This assignment should be completed individually. For each problem, submit your SQL statements and a screen shot of the SQL results in a single Word document or pdf file. Submit the file via eLearning.**

I recommend creating a new user and workspace named after your netid, log in as that user and load the database script facts.sql (provided in this week's assignment folder).  Before you attempt to write any SQL queries, familiarize yourself with the database structure and data. I have provided a relational diagram and sample data for this database.

Write queries to address each of the problems below. **Submit both the SQL statements and the screen prints of the outputs from Oracle.** (2 points for the screen shots).  **Be sure the workspace name (your id) is included in your screen shots!!!**



FACT Description and ERD

The CIS Department at Tiny College maintains the Free Access to Current Technology (FACT) library of eBooks. FACT is a collection of current technology eBooks for use by faculty and students. Agreements with the publishers allow patrons to electronically check out a book, which gives them exclusive access to the book online through the FACT website, but only one patron at a time can have access to a book. A book must have at least one author but can have many. An author must have written at least one book to be included in the system, but may have written many. A book may have never been checked out, but can be checked out many times by the same patron or different patrons over time. Because all faculty and staff in the department are given accounts at the online library, a patron may have never checked out a book or they may have checked out many books over time. To simplify determining which patron currently has a given book checked out, a redundant relationship between BOOK and PATRON is maintained.

1. **Create a sequence that can be used to manage the check\_num in the CHECKOUT table. After you create the sequence, insert a new record into the Checkout table using the sequence to assign the check\_num.** *(7 pts)*

**select max(check\_num) from checkout**

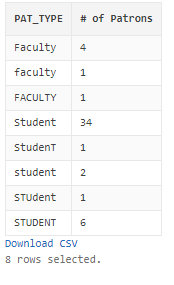
**create sequence check\_num\_sequence**

**start with 91068 nocache**

**insert into checkout**

**values(check\_num\_sequence.nextval, 5235, 1165, '24-OCT-19', '01-NOV-19', '28-OCT-19')**

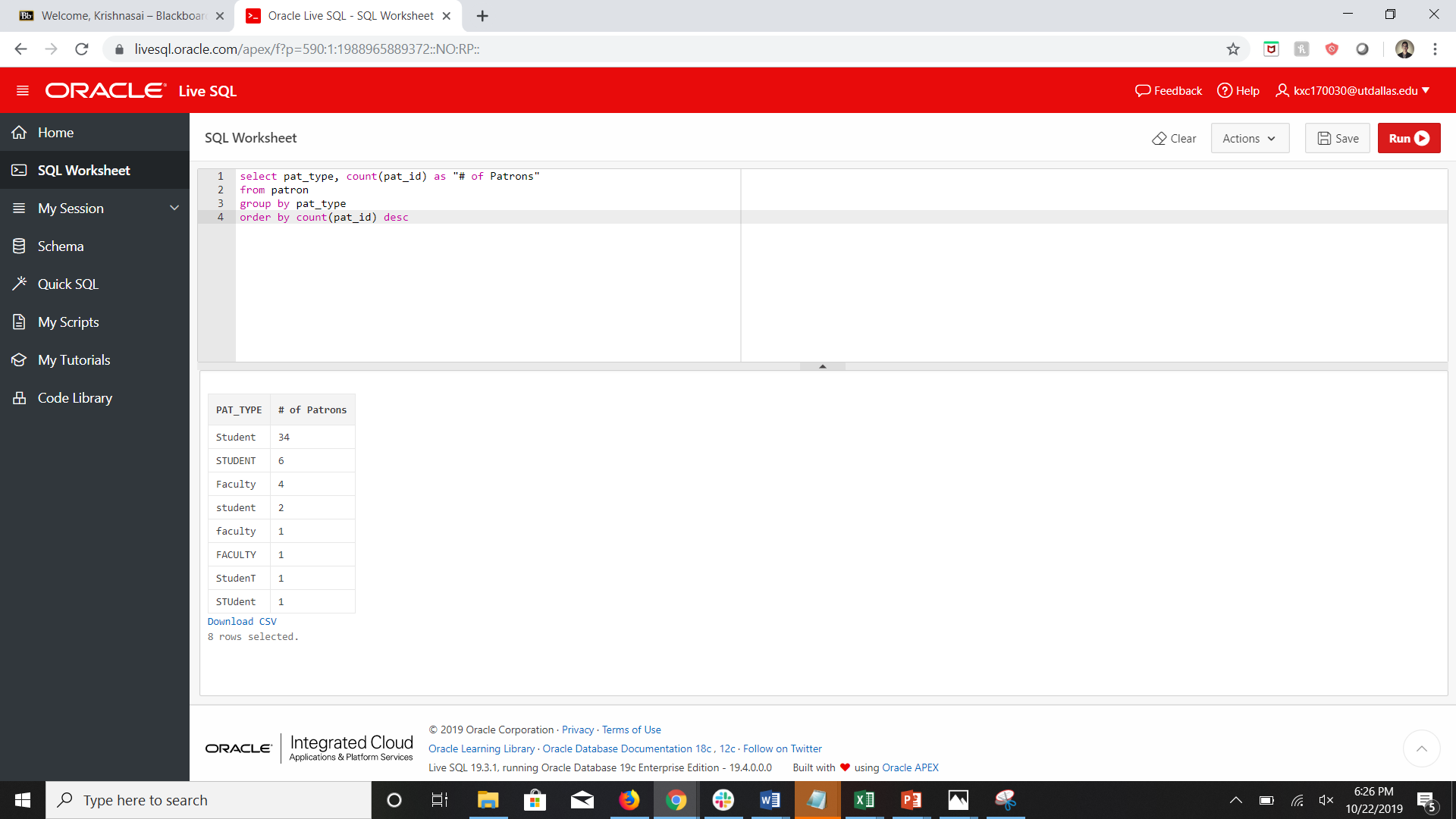
1. **Write a query to display the patron type and the number of patrons in each type, order by the number of patrons in descending order.**

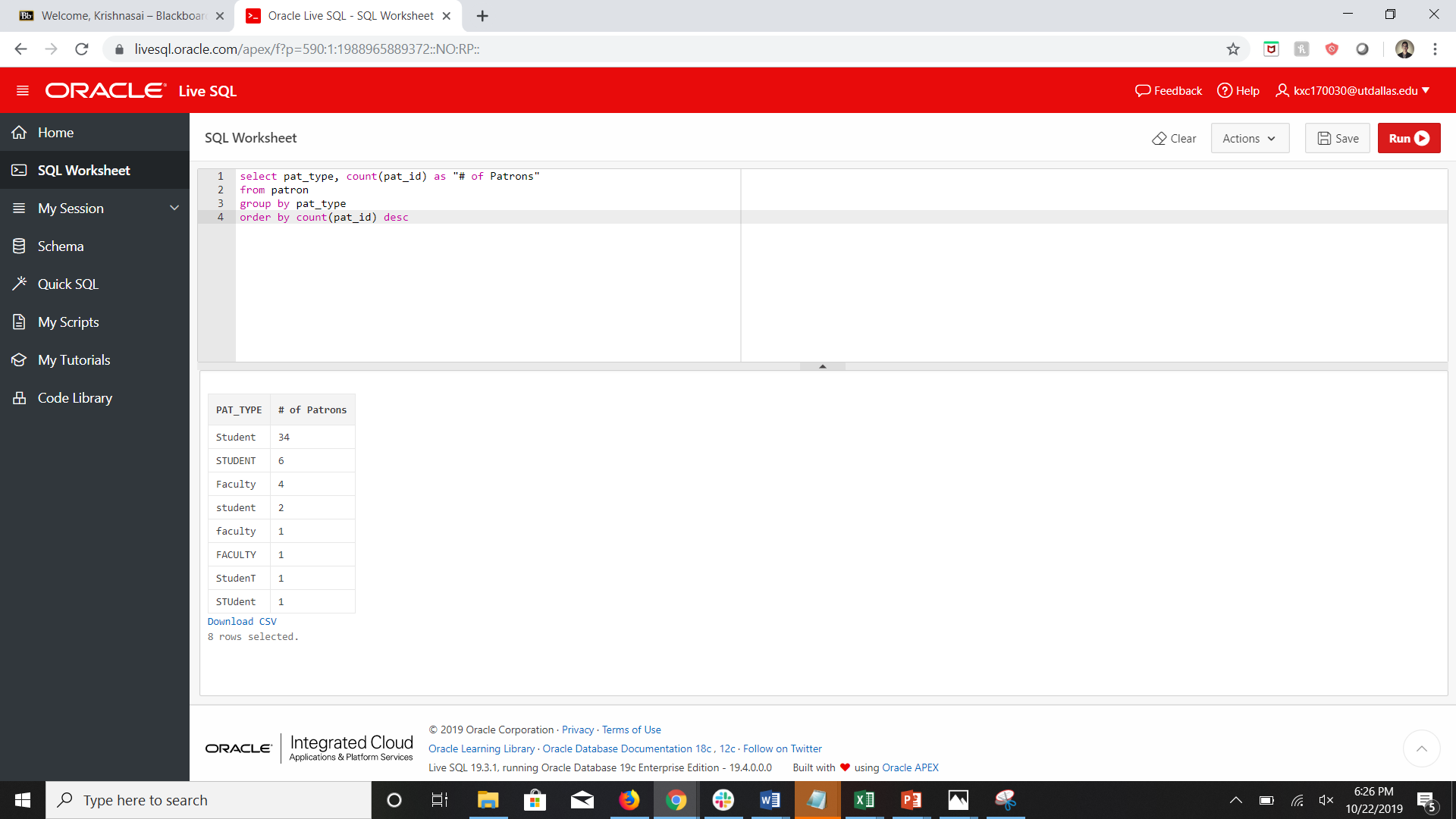


**select pat\_type, count(pat\_id) as "# of Patrons"**

**from patron**

**group by pat\_type**

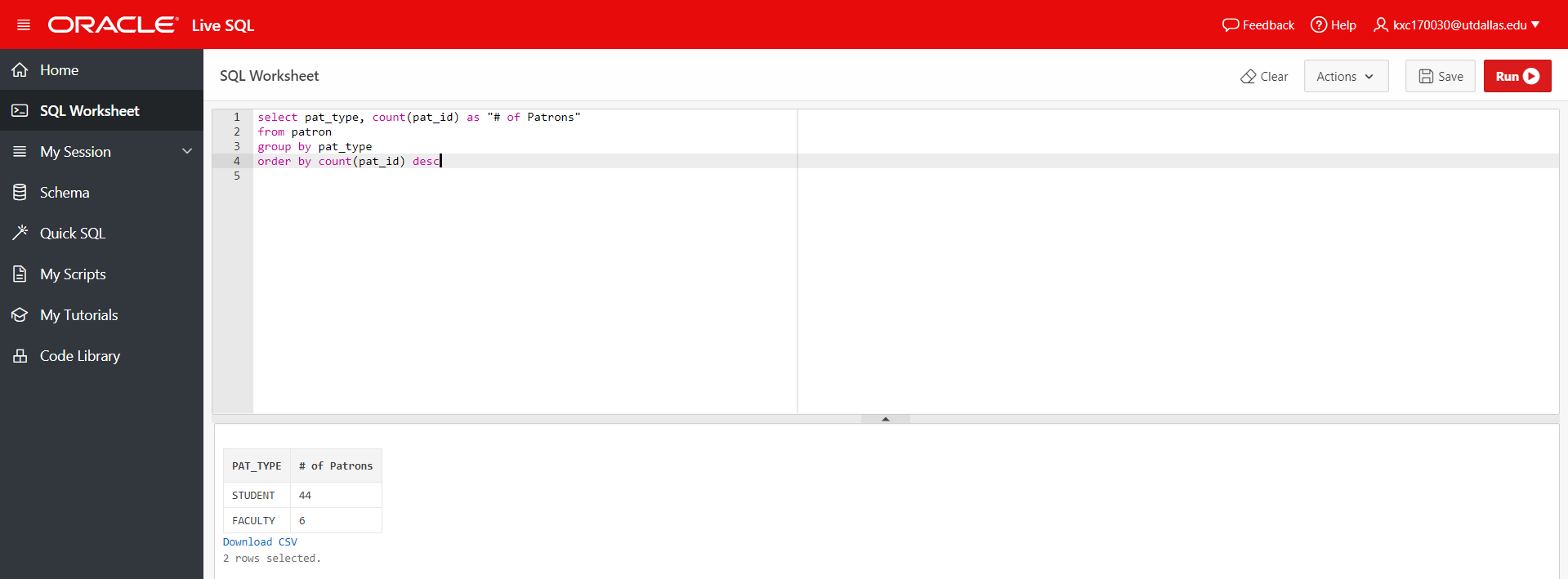
**order by count(pat\_id) desc**

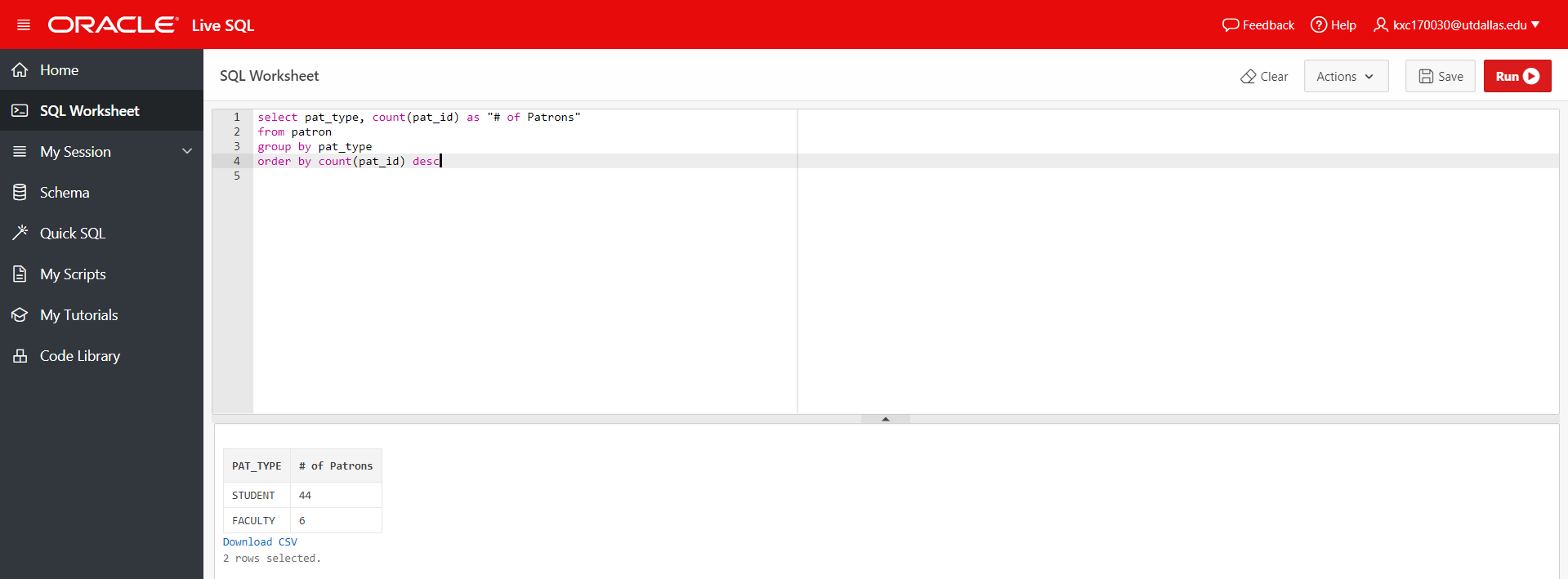
****

1. **Write a query that will update the values of PAT\_TYPE in the PATRON table to have consistent formatting – all upper case.** *Show the SQL statement to perform the update here and then rerun the query in #2 and show the results in the screen shot. (7 pts)*

**update patron**

**set pat\_type = upper(pat\_type)**

****

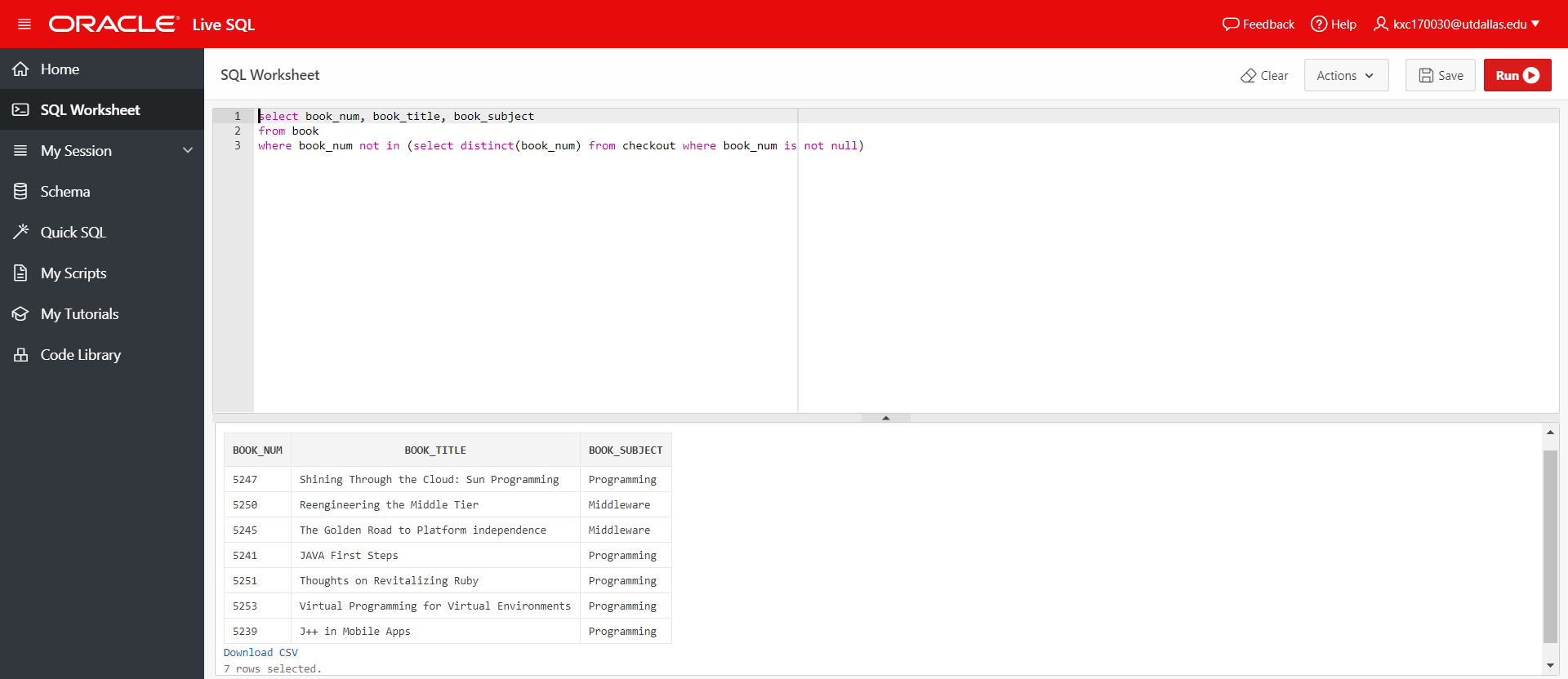
****

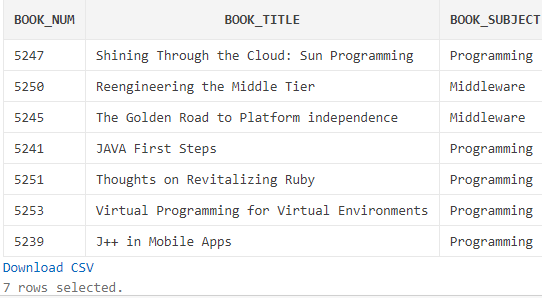
1. **Write a query that will list all the books that have never been checked out. List the book number, book title and subject. Use a subquery.** *Show the query and provide a screen shot.*

**select book\_num, book\_title, book\_subject**

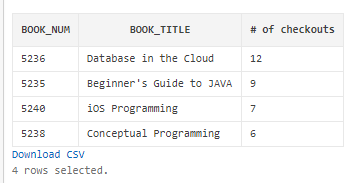
**from book**

**where book\_num not in (select distinct(book\_num) from checkout where book\_num is not null)**

****

****

1. **Write a query that will list all the number of times each book has been checked out. List the book number, book title and number of times each book has been checked out. Only list those books that have been checked out more than 5 times. Order the result set in descending order by number of checkouts.** *Show the query and provide a screen shot.*



**select b.book\_num, b.book\_title, count(check\_num) as "# of checkouts"**

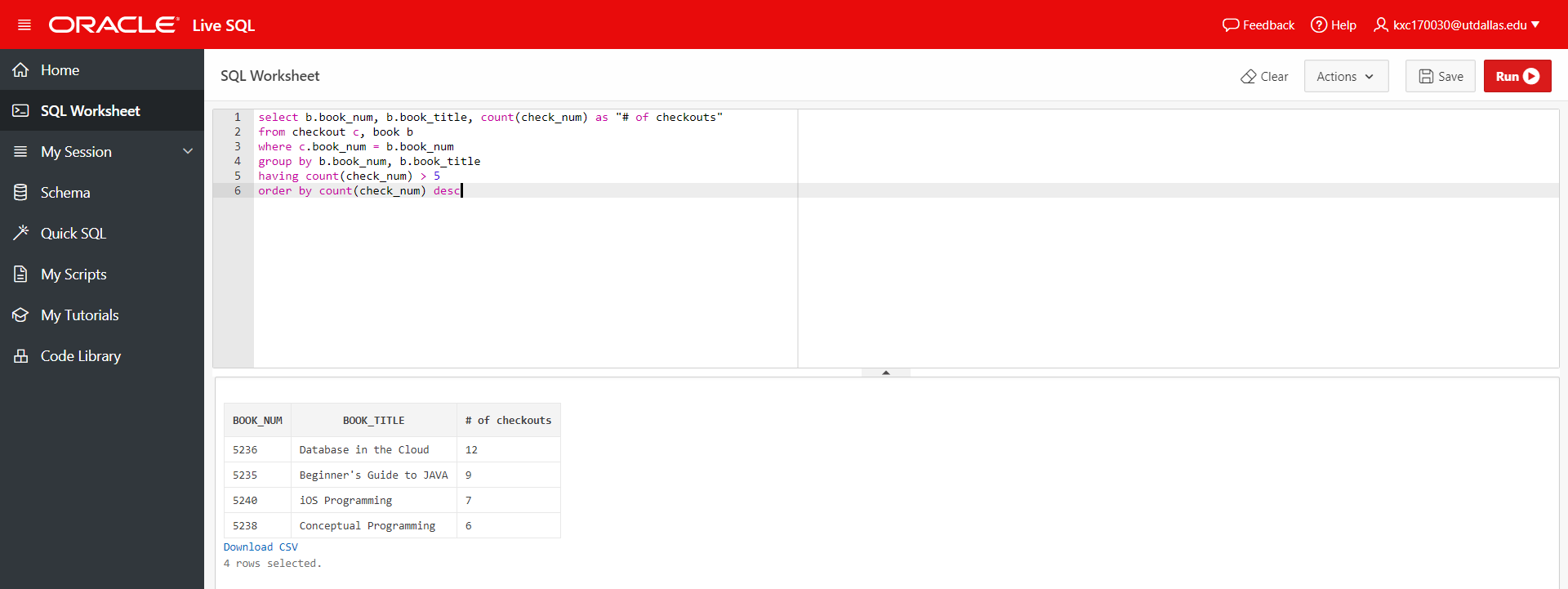
**from checkout c, book b**

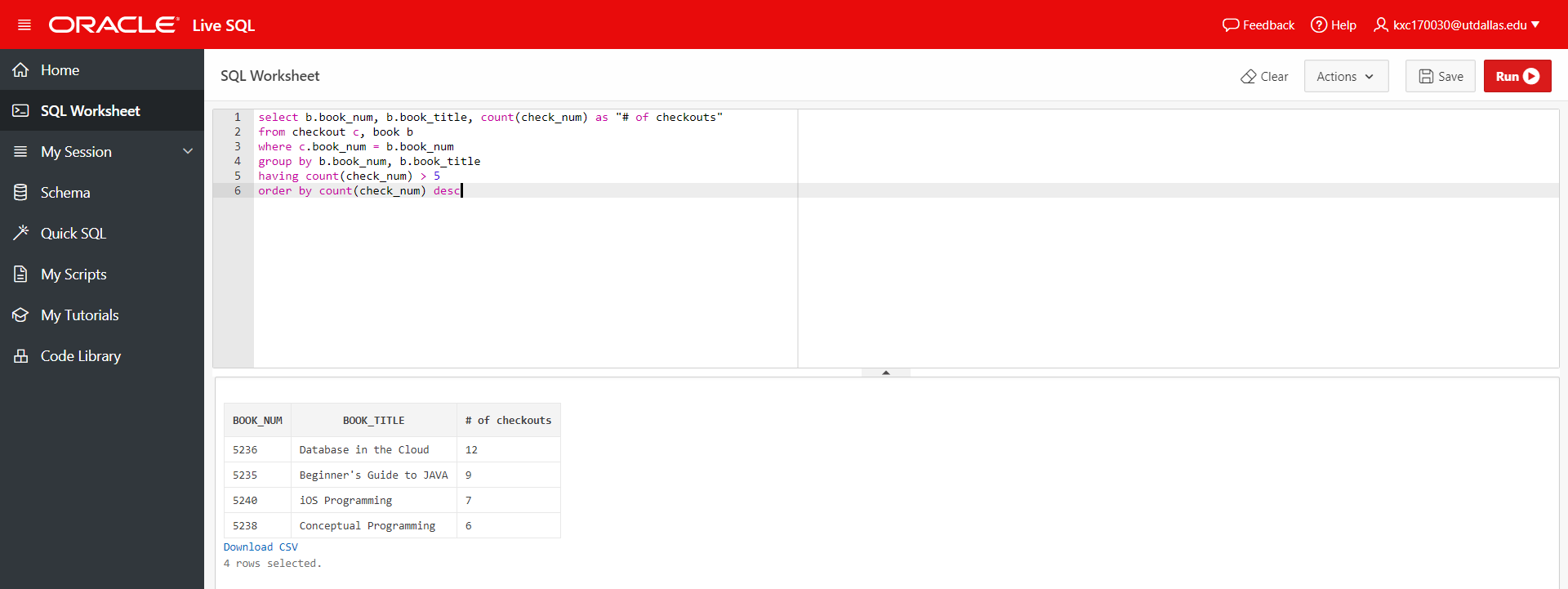
**where c.book\_num = b.book\_num**

**group by b.book\_num, b.book\_title**

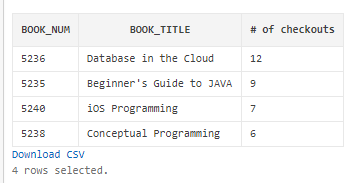
**having count(check\_num) > 5**

**order by count(check\_num) desc**

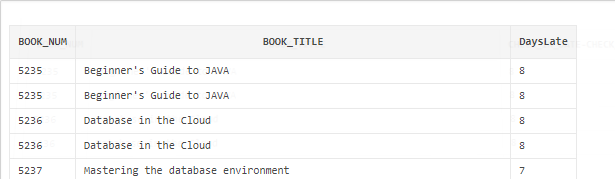
****

****

1. **Write a query that will list all the number of times each book has been checked out. List the book number, book title and number of times each book has been checked out. Only list those books that have been checked out more than 5 times. Order the result set in descending order by number of checkouts.** *Show the query and provide a screen shot.*



1. **Write a query that will display those books that were returned after the due date. Include the number of days the book was returned after the due date. Display the book\_title, book number, and number of days over the check out date.** *Show the query and provide a screen shot.*

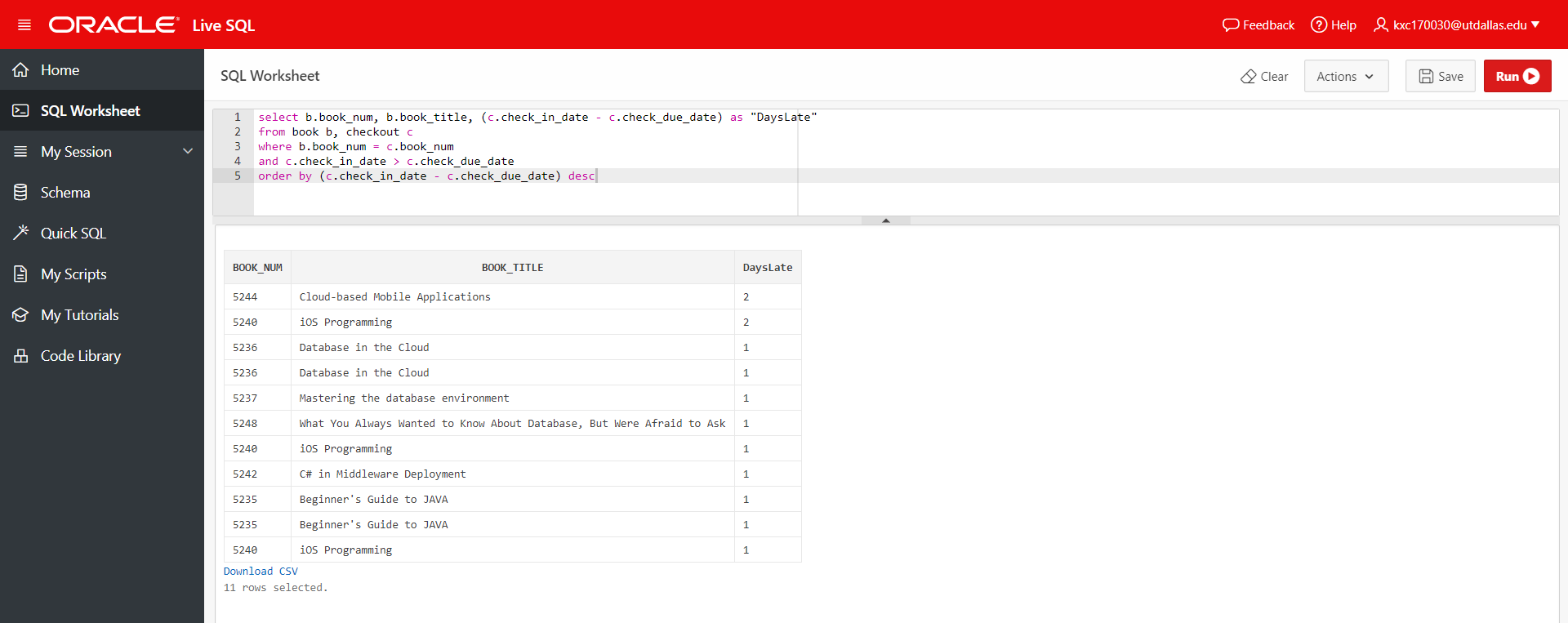


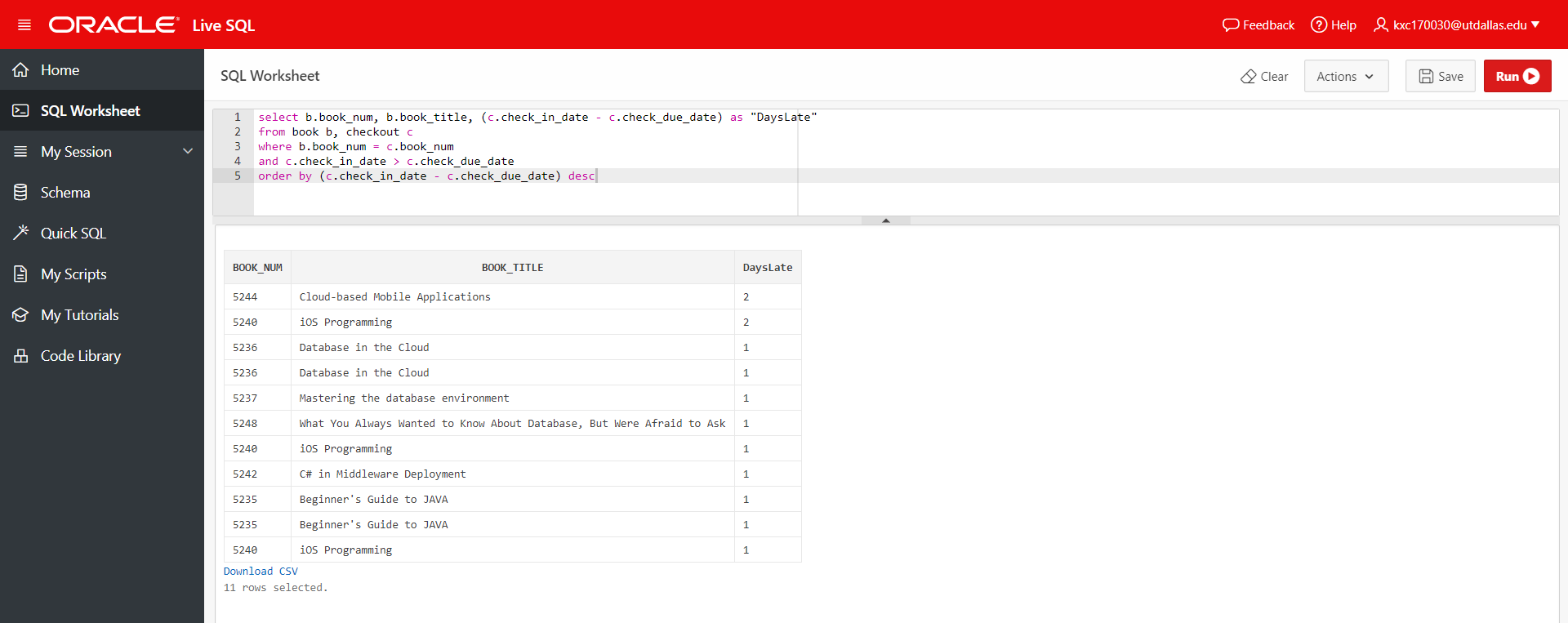
**select b.book\_num, b.book\_title, (c.check\_in\_date - c.check\_due\_date) as "DaysLate"**

**from book b, checkout c**

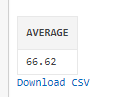
**where b.book\_num = c.book\_num**

**and c.check\_in\_date > c.check\_due\_date**

**order by (c.check\_in\_date - c.check\_due\_date) desc**

****

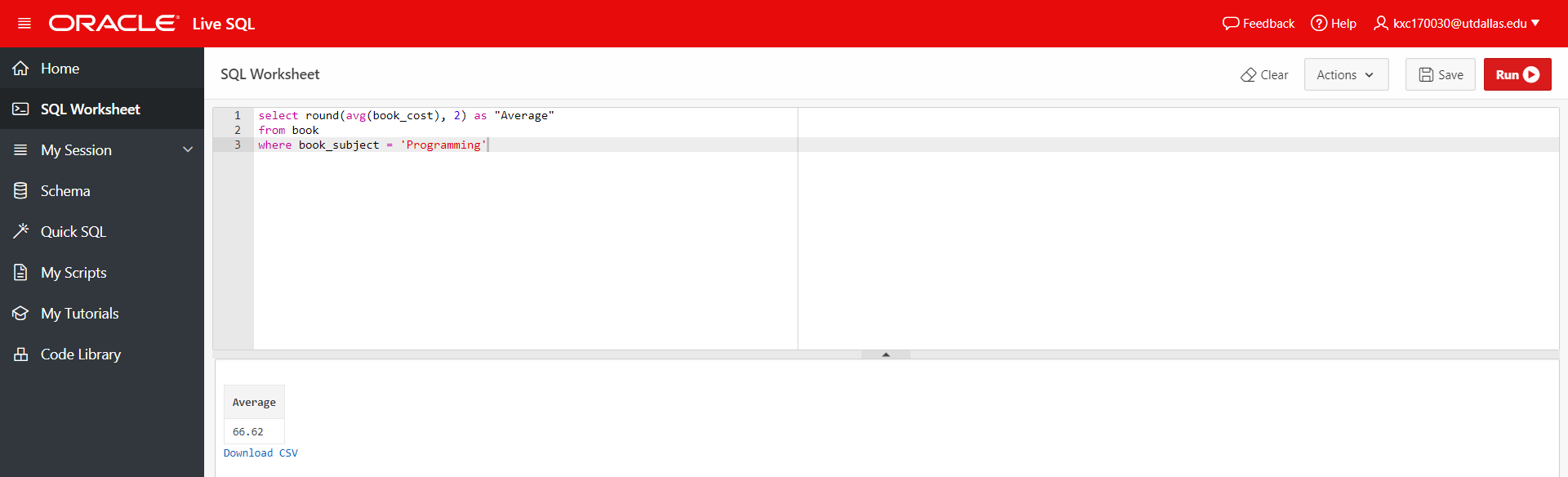
1. **Display the average book cost for all programming books. Round the output to 2 decimal places.** *Show the query and provide a screen shot.*

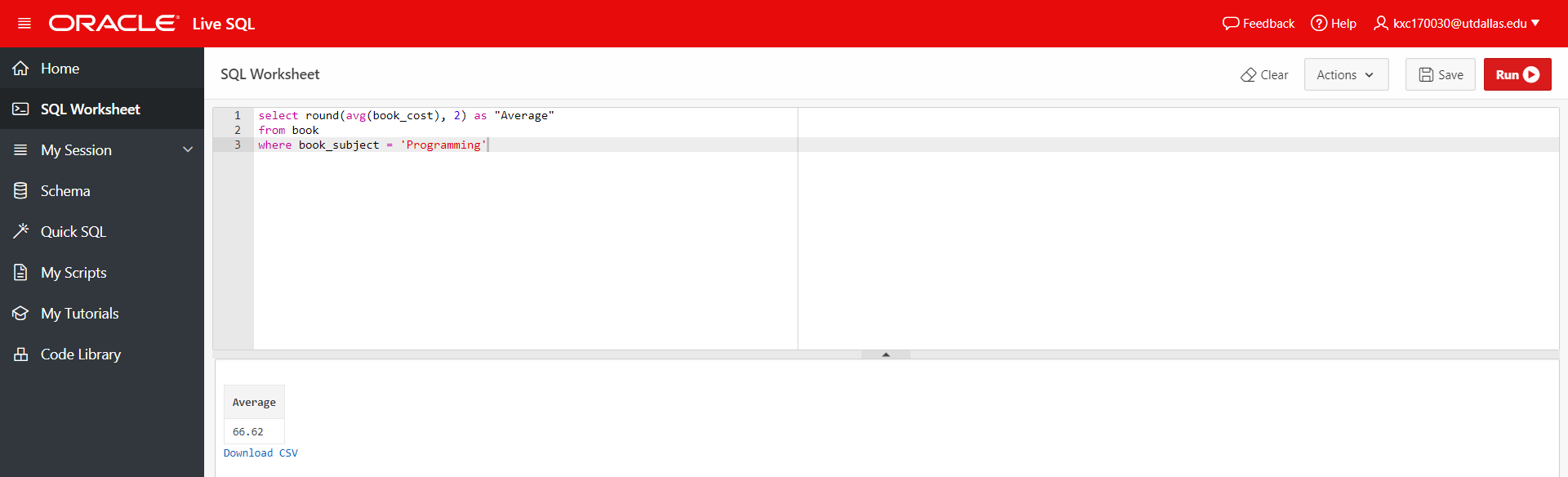


**select round(avg(book\_cost), 2) as "Average"**

**from book**

**where book\_subject = 'Programming'**

****

****