Assignment#1

MSIT 630 Database Systems
Total: 60 points
Renee Raven

1. Explain the concept of physical data independence and its importance in database systems (**4 points**)

A database system design, or schema, is divided into three levels: the physical level, the logical level, and the view level. The physical level determines where the data is stored, what devices the data is stored on, and the type of media used to store the data. In essence, the physical level (or schema) involves the actual physical storage of the data. The logical level focuses on which specific data is stored and how data structures are related. The view level allows users to access the data they need to perform a function through a specific view, like a dashboard.

Ideally, the complexity of each level is hidden from the higher level(s). For instance, a programmer writing new applications to work in the logical level or users accessing the data through a view level will not be affected when there are changes in the physical level (such as moving data to a new storage facility or cloud provider). Physical data independence describes the robust separation that shields, through abstraction, the logical and view levels from changes made at the physical level. This abstraction is fundamental to good database design and allows users to interact with the database at an appropriate level.

2. Explain what problems are caused by the design of the following table. (**4 points**)

customer_id	*customer_name*	*customer_street*	*customer_city*	*account_number*
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-101
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-201
677-89-9011	Hayes	3 Main St.	Harrison	A-102
182-73-6091	Turner	123 Putnam St.	Stamford	A-305
321-12-3123	Jones	100 Main St.	Harrison	A-217
336-66-9999	Lindsay	175 Park Ave.	Pittsfield	A-222
019-28-3746	Smith	72 North St.	Rye	A-201

The Customer table

There are several obvious problems with this table, but all the problems relate to the lack of irreducibility, or the uniqueness property. The first two rows have duplicate entries for each column besides the account_number column. However, the account_number column has A-201 listed for two different customer_name entities. It is unclear if this is a mistake or if there can be multiple customers associated with a single account_number (for instance, a business account could belong to several sales reps). There is no single-column primary key because of duplicates, and it would take at least three columns for a composite key to act as a primary key and provide a unique identification of the tuple (for example the combination of customer_id, customer_street, and account_number uniquely identify a row).

3. List four significant differences between a file-processing system and a DBMS. (**4 points**)

Data availability, data consistency, cost, and atomicity are four key differences between file-processing systems (FPS) and database management systems (DBMS). In a DBMS, all applications have access to the same data and retrieval is convenient and efficient. An FPS utilizes multiple applications to store the same data and accessing specific data is often convoluted and inefficient. Similarly, the way FPS maintains the same data in different applications invites data inconsistency. Multiple copies of the same data may be updated at different times by different applications. DBMS's built-in control of data redundancy limits the risk of data inconsistency.

Interestingly, changing DBMS is cheaper to maintain because there is no need for new application programs with any change in the data structure. Maintaining FBS can become quite expensive since any change in the data structure requires modification to code or new applications. Atomicity requires transactions must happen in their entirety or not at all. For instance, transferring funds between accounts must be reflected in both accounts at the same instant. A DBMS includes constraints that ensure atomicity, while FBS lack a structure to enforce atomicity.

4. Describe the differences in meaning between the terms *relation* and *relation schema*. (**4 points**)

A relation schema can be thought of as the organizational structure of a table, while relation refers to the table itself. Further, a relation instance is a specific row (tuple) or collection of rows. The relation schema defines each column's attributes and domains. For instance, when the relation schema defines a column type as a number the schema prevents a text entry in that column.

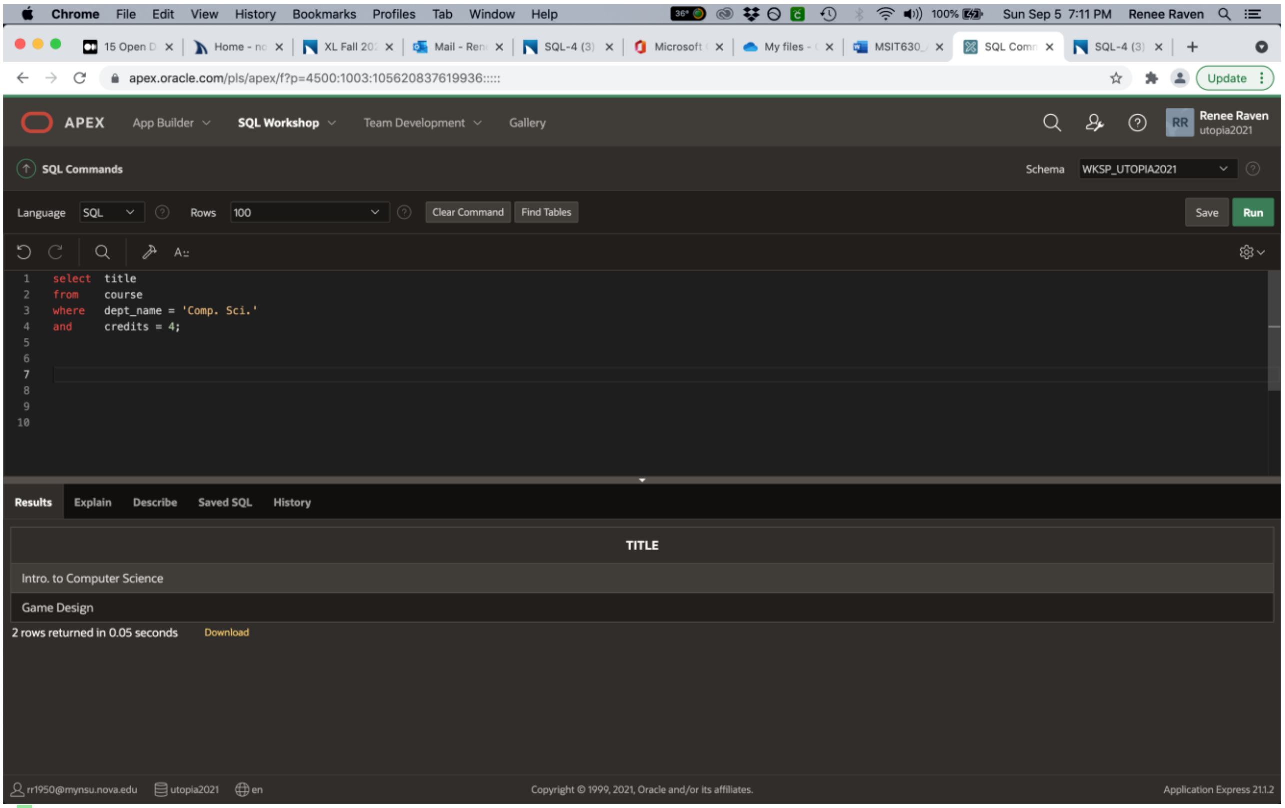
5. List two reasons why null values might be introduced into the database. (**4 points**)

Null values indicate an unknown value or a value that does not exist. In a database of exotic plants, there may be a column for country_of_orgin to indicate the country the plant first grew in. For some plants that column would be null because there may be no authenticated record of the plants origins.

6. Write the following queries in SQL, using the university schema, execute your SQL statement on the sample database with small relations and show me both the SQL statements and the query results. (https://www.db-book.com/db7/university-lab-dir/sample_tables-dir/index.html) (**20 points, 4 points each**)

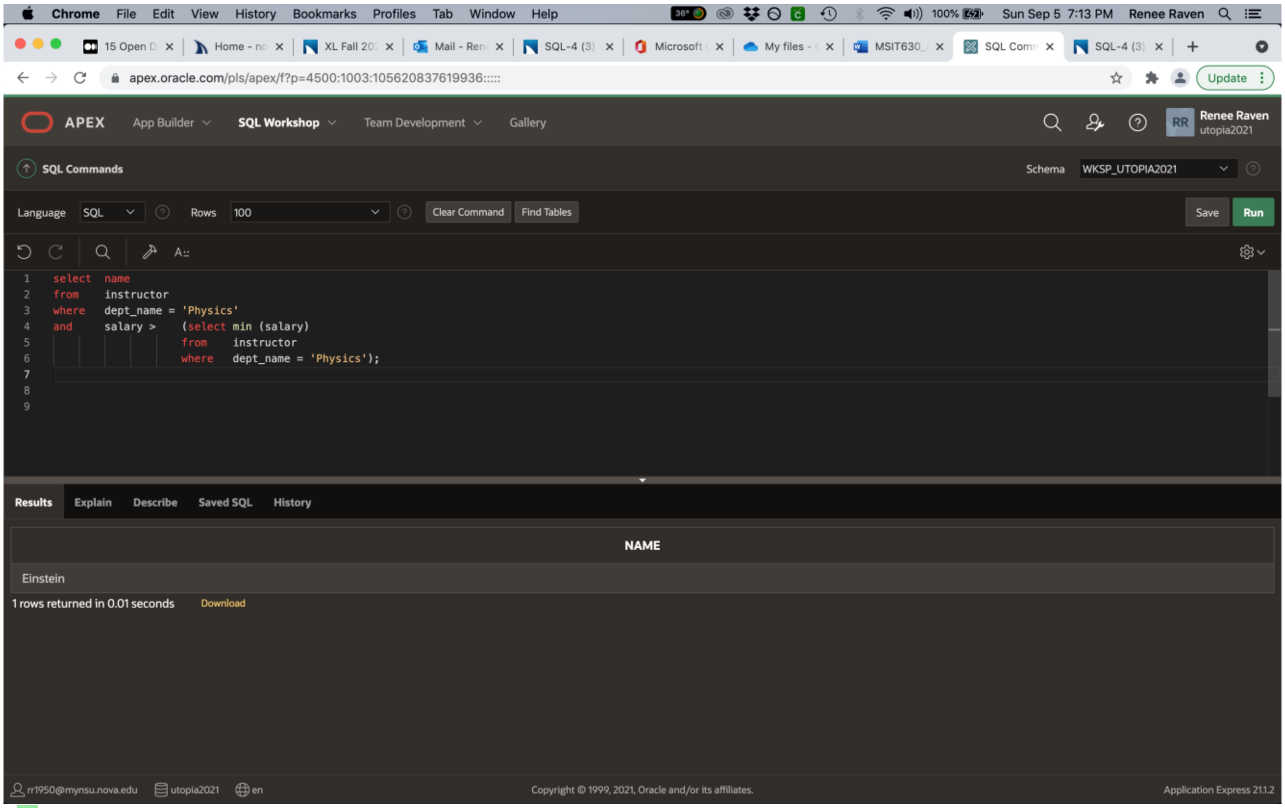
a. Find the titles of courses in the Comp. Sci. department that have 4 credits.

```
select title
from course
where dept_name = 'Comp. Sci.'
and credits = 4;
```



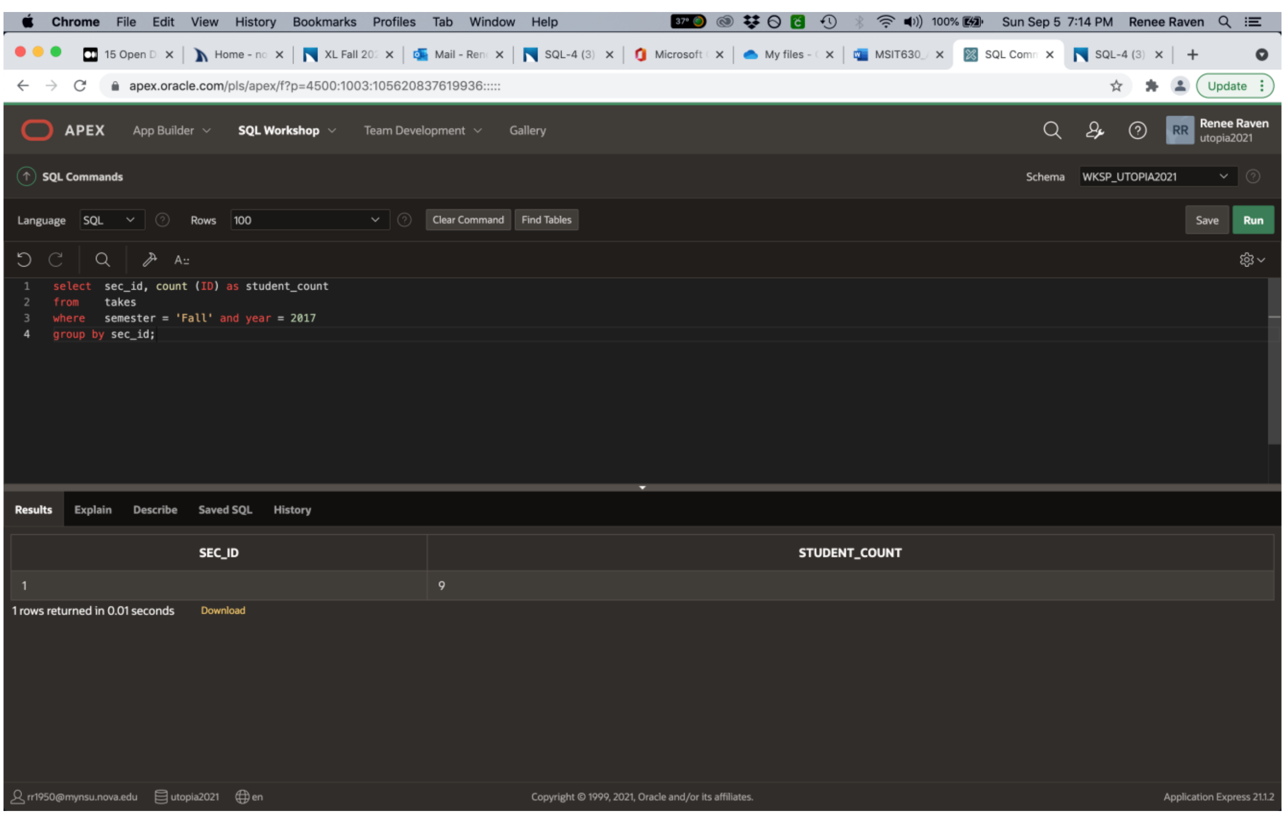
b. Find the name(s) of the instructor(s) who DON'T earn the lowest salary in Physics department.

```
select name
from instructor
where dept_name = 'Physics'
and salary > (select min (salary)
from instructor
where dept_name = 'Physics');
```



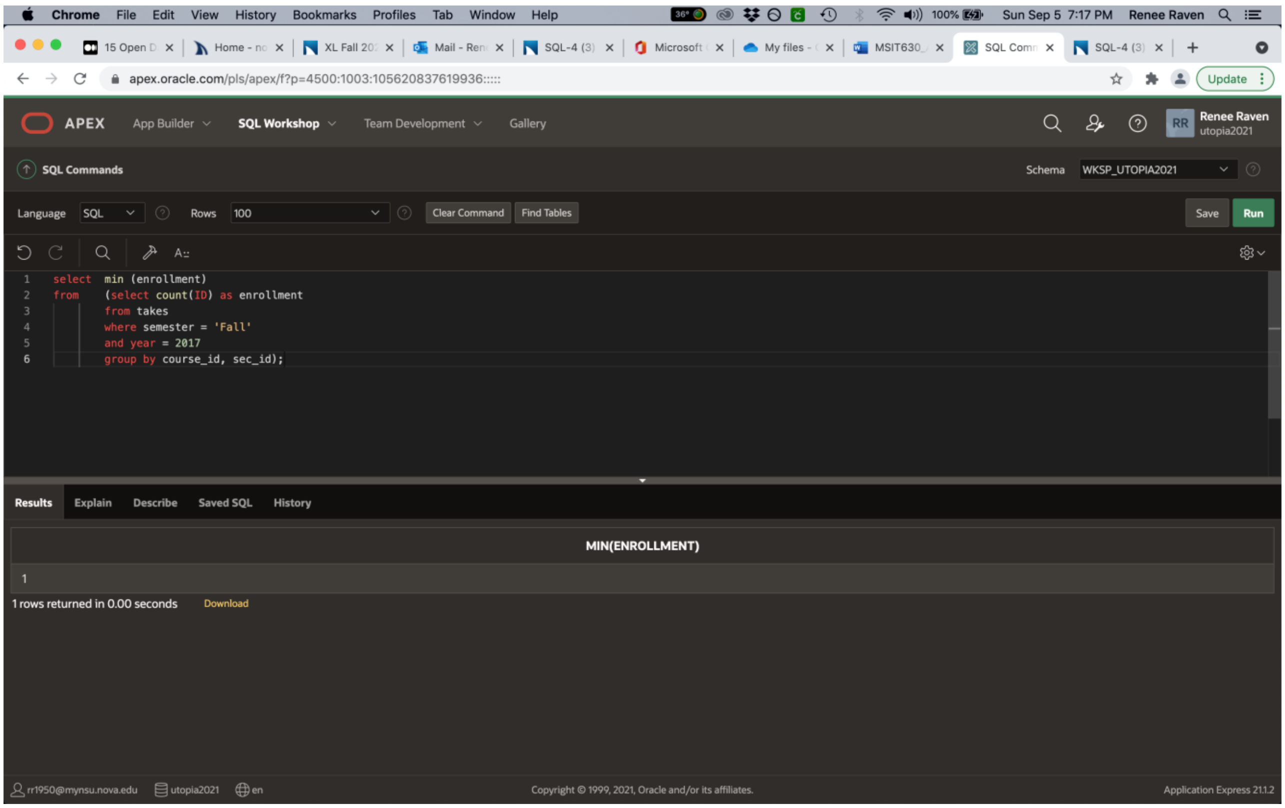
c. Find the enrollment of each section (number of students enrolled) that was offered in Fall 2017.

```
select sec_id, count (ID) as student_count
from takes
where semester = 'Fall' and year = 2017
group by sec_id;
```



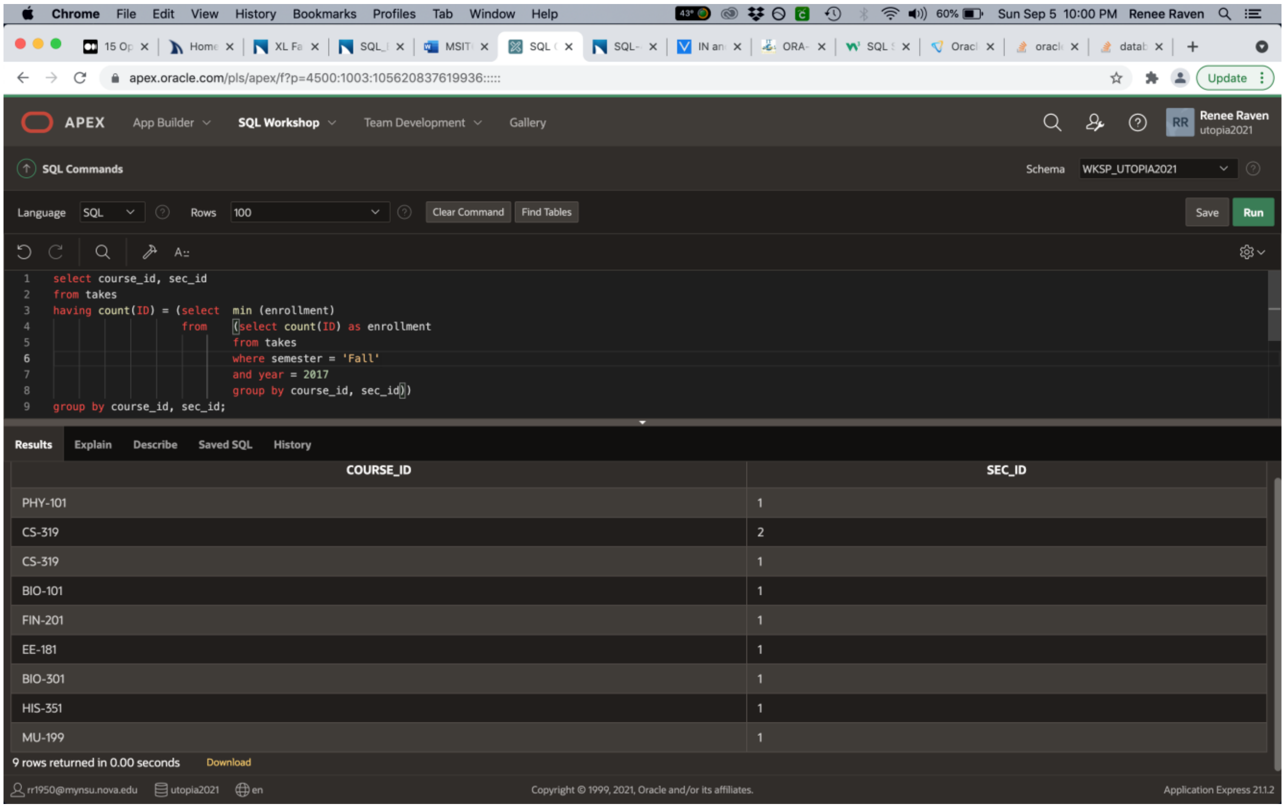
d. Find the minimum enrollment, across all sections offered in Fall 2017.

```
select min (enrollment)
from (select count(ID) as enrollment
from takes
where semester = 'Fall'
and year = 2017
group by course_id, sec_id);
```



e. Find the course ID and section ID of the sections that had the minimum enrollment in Fall 2017.

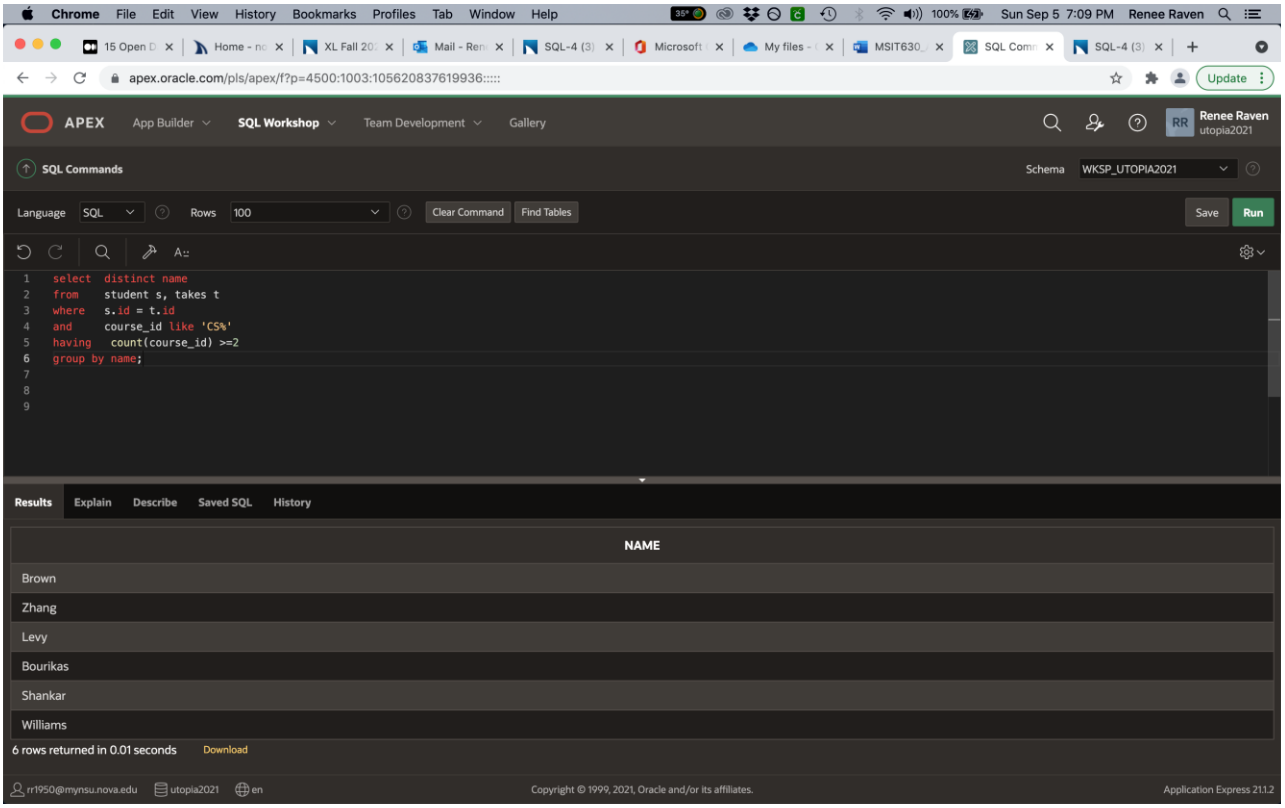
```
select course_id, sec_id
from takes
having count(ID) = (select min (enrollment)
from (select count(ID) as enrollment
from takes
where semester = 'Fall'
and year = 2017
group by course_id, sec_id))
group by course_id, sec_id;
```



7. Write the following queries in SQL, using the university schema, execute your SQL statement on the sample database with small relations and show me both the SQL statements and the query results (https://www.db-book.com/db7/university-lab-dir/sample_tables-dir/index.html) (**20 points, 4 points each**)

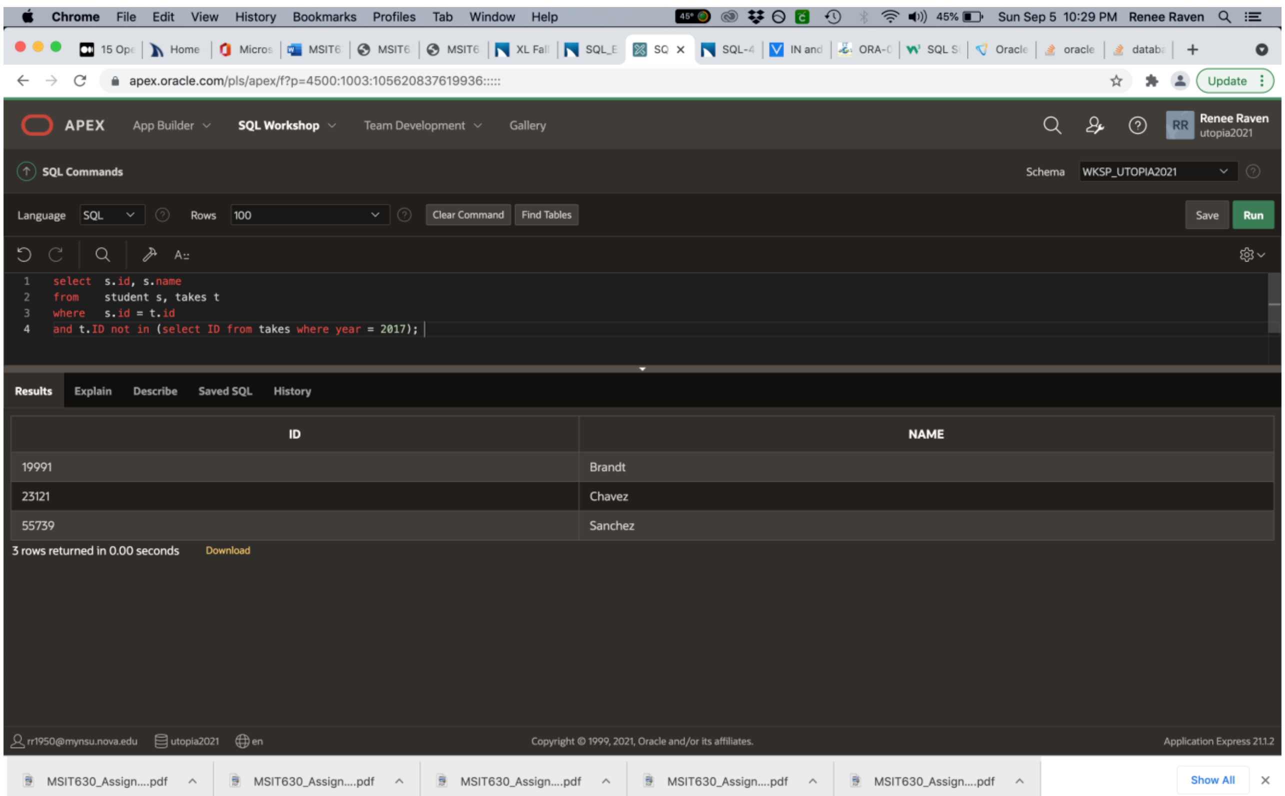
a. Find the names of all students who have taken at least two courses offered by Comp. Sci. department; make sure there are no duplicate names in the result. Note that student in other departments can take courses from Comp. Sci. as well.

```
select distinct name
from student s, takes t
where s.id = t.id
and course_id like 'CS%'
having count(course_id) >=2
group by name;
```


SQL Commands
Schema WKSP_UTOPIA2021
Language SQL
Rows 100
Clear Command
Find Tables
Save
Run
select distinct name
from student s, takes t
where s.id = t.id
and course_id like 'CS%'
having count(course_id) >=2
group by name;
Results
Explain
Describe
Saved SQL
History
NAME
Brown
Zhang
Levy
Bourikas
Shankar
Williams
6 rows returned in 0.01 seconds
Download
Copyright © 1999, 2021, Oracle and/or its affiliates.
Application Express 21.1.2

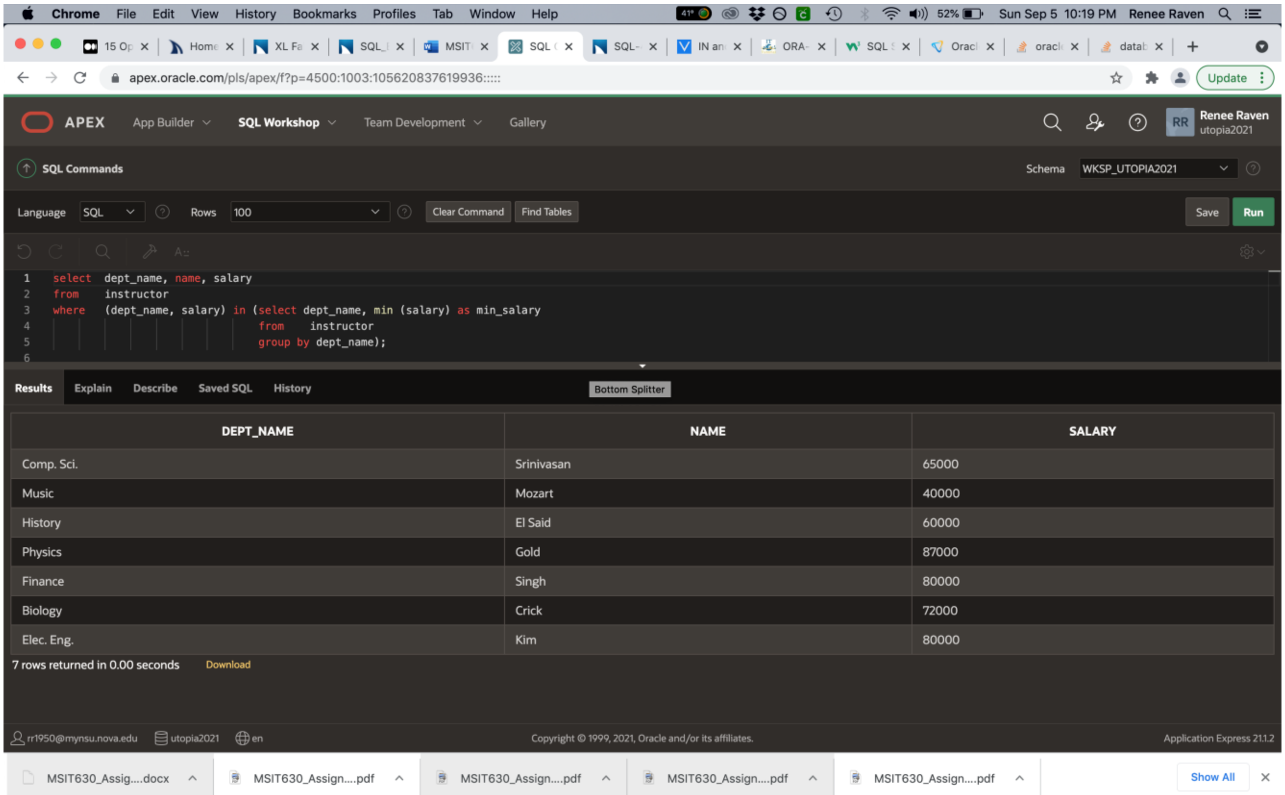
b. Find the IDs and names of all students who have not taken any course offering in 2017.

```
select s.ID, s.name
from student s, takes t
where s.ID = t.ID
and t.ID not in (select ID from takes where year = 2017);
```



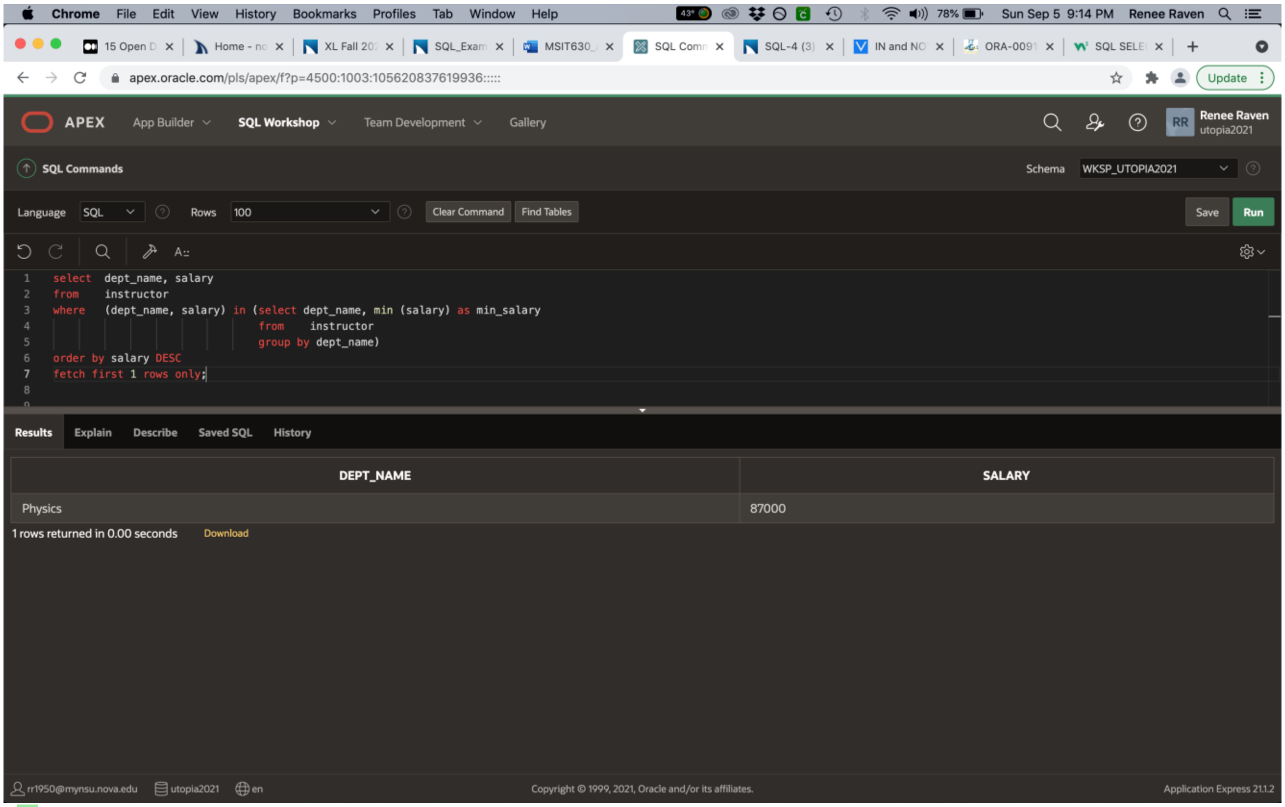
c. For each department, find the name and salary of the instructor who earns the minimum salary in that department. You may assume that every department has at least one instructor.

```
select dept_name, name, salary
from instructor
where (dept_name, salary) in (select dept_name, min (salary) as min_salary
from instructor
group by dept_name);
```



d. Find the highest, across all departments, of the per-department minimum salary computed by the preceding query (part 7.c).

```
select dept_name, salary
from instructor
where (dept_name, salary) in (select dept_name, min (salary) as min_salary
from instructor
group by dept_name)
order by salary DESC
fetch first 1 rows only;
```



e. Find the course titles of all prerequisite courses of "CS-319".

```
select c.title
from course c, prereq p
where c.course_id = p.prereq_id
and p.course_id = 'CS-319';
```