CS4710/6710 Database Systems (Fall 2017)

Instructor: Liyang Yu

Assignment #4: Index and SQL Lab, Due: October 24th, 2017

Solution

Name: Sandhya Ramadasan , Panther Id: #002351924

Consider the "index calculation" problem we have studied in the class (see lecture note), where we have a relation of 8,000,000 tuples. Again, make the following assumptions:

- the size of disk block (also called disk page): it is usually 4K or 8K bytes, to make our calculation easier, we will use **4000** bytes as a block size
- the size of the search key value: say we index the relation by using an attribute that is part of the primary key, and it is **40** byte string
- the size of physical page addresses: this is needed since every node in the tree will store search key values as well as references to disk pages, i.e., the address of the disk page being referenced. For ease of calculation, we use a **10**-byte page address
- each swap takes about 10 milliseconds

1. For this relation, calculate how long it will take to find a particular record if we use dense index, as described on lecture note, page 13? [16 points]

Solution:

Given:

Size of disk block (disk Page) = 4000 bytes

Size of each Search Key value = 40 bytes

Size of physical page address = 10 bytes

Time per Swap = 10 milliseconds

Total tuples = 8,000,000 tuples

Calculation:

Size of each index = 40+10 = 50 bytes

Number of Records per block(Index Entry per page) = (4000)/50 = 80 records

Total Number of blocks for 8000000 tuples = 8000000/80 = 100000 blocks

To search inside the index, Number of Swaps = $\log_2(100000)$ = 17 swaps

Using Dense index, Swaps needed = 17+1 = 18 Swaps.

Time taken to search a particular record (Dense Index)= 18*10 = **180 milliseconds**

=> 180 milliseconds

2. Consider again the situation in question 1, with the same dense index, but add one extra level of sparse index, as described by lecture note page 16-17. Now, how long it will take to find one particular record? [16 points]

Solution:

Given:
Size of disk block (disk Page) = 4000 bytes

Size of each Search Key value = 40 bytes

Size of physical page address = 10 bytes

Time per Swap = 10 milliseconds

Total tuples = 8,000,000 tuples

Calculation:

Size of each index = 40+10 = 50 bytes

Number of Records per block(Index Entry per page) = (4000)/50 = 80 records

Total Number of blocks for 8000000 tuples = 8000000/80 = 100000 blocks

Adding one Extra level of Sparse Index :
For 100000 blocks of dense Index= 100000/80 = 1250

Number of Swaps = $\log_2 1250$ = 11 swaps

Total Number of Swaps = 11 + 1 + 1 = 13 swaps

Time taken to search a particular record (with One extra level of Sparse Index)=
= 13*10 = **130 milliseconds**

=> 130 milliseconds

The following are exercises you need to finish and *submit screen copies*:

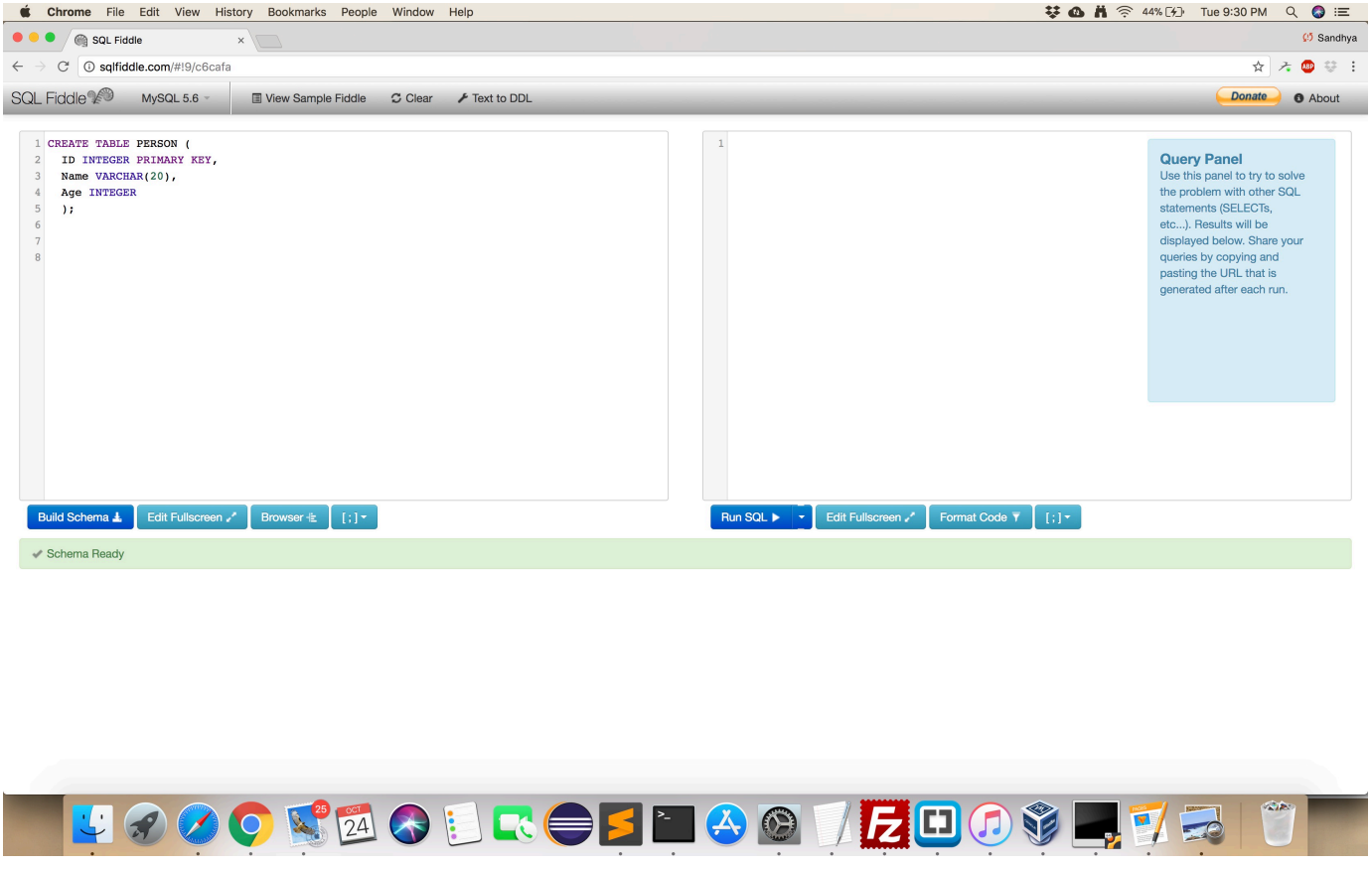
1. create a table named PERSON (4 points)

This is very a simple table, which has the following attributes:

id: integer, also the primary key
name: varchar(20)
age: integer

Use the right SQL statements to do this, submit a screen copy showing the statements you have used and the executing results.

Solution



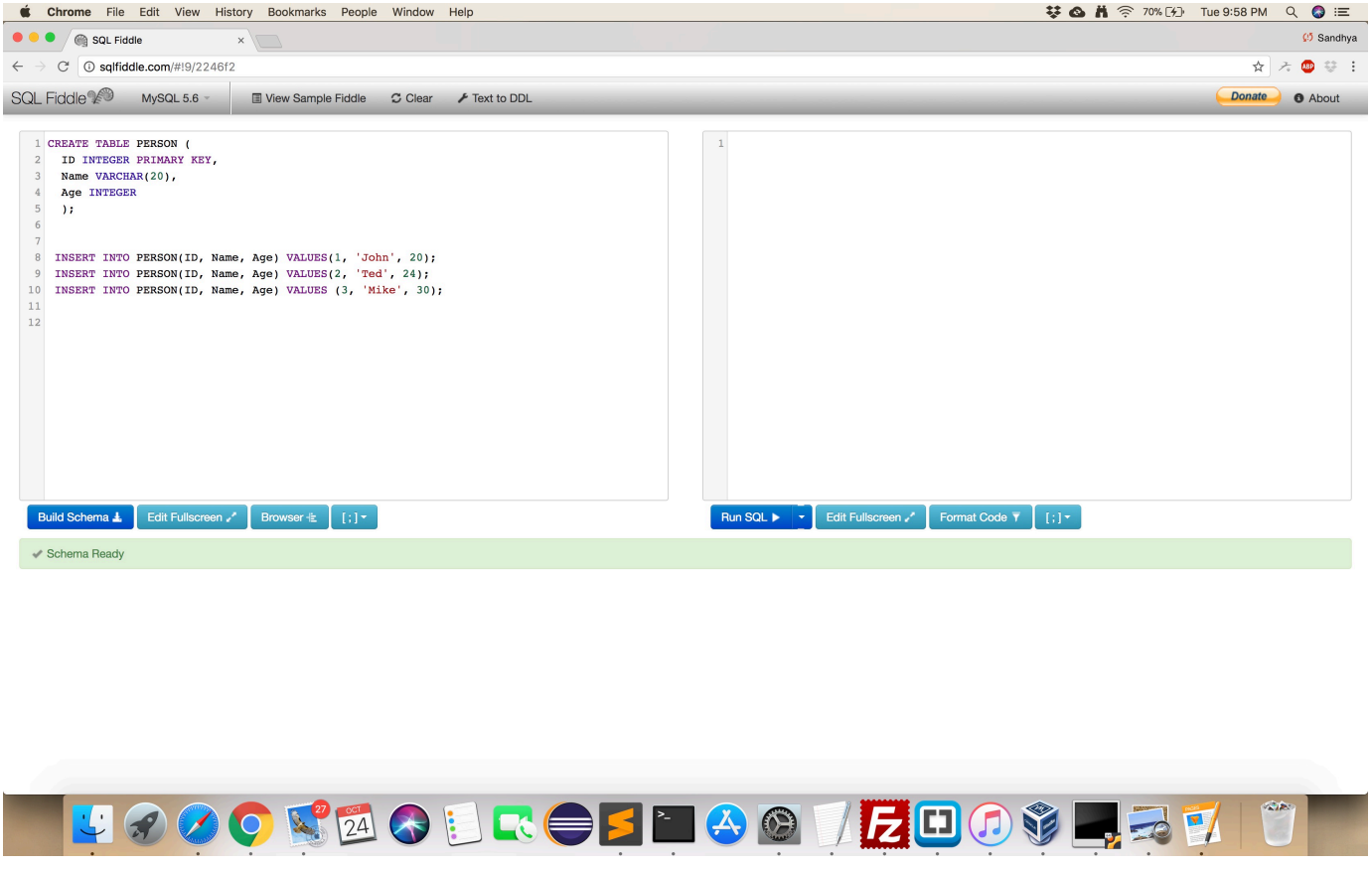
2. insert a few records into PERSON table (6 points)

Use correct SQL statements to insert the following person into the PERSON table:

John, id is 1, age is 20
Ted, id is 2, age is 24
Mike, id is 3, age is 30

Submit a screen copy showing the statements you have used and the results you see.

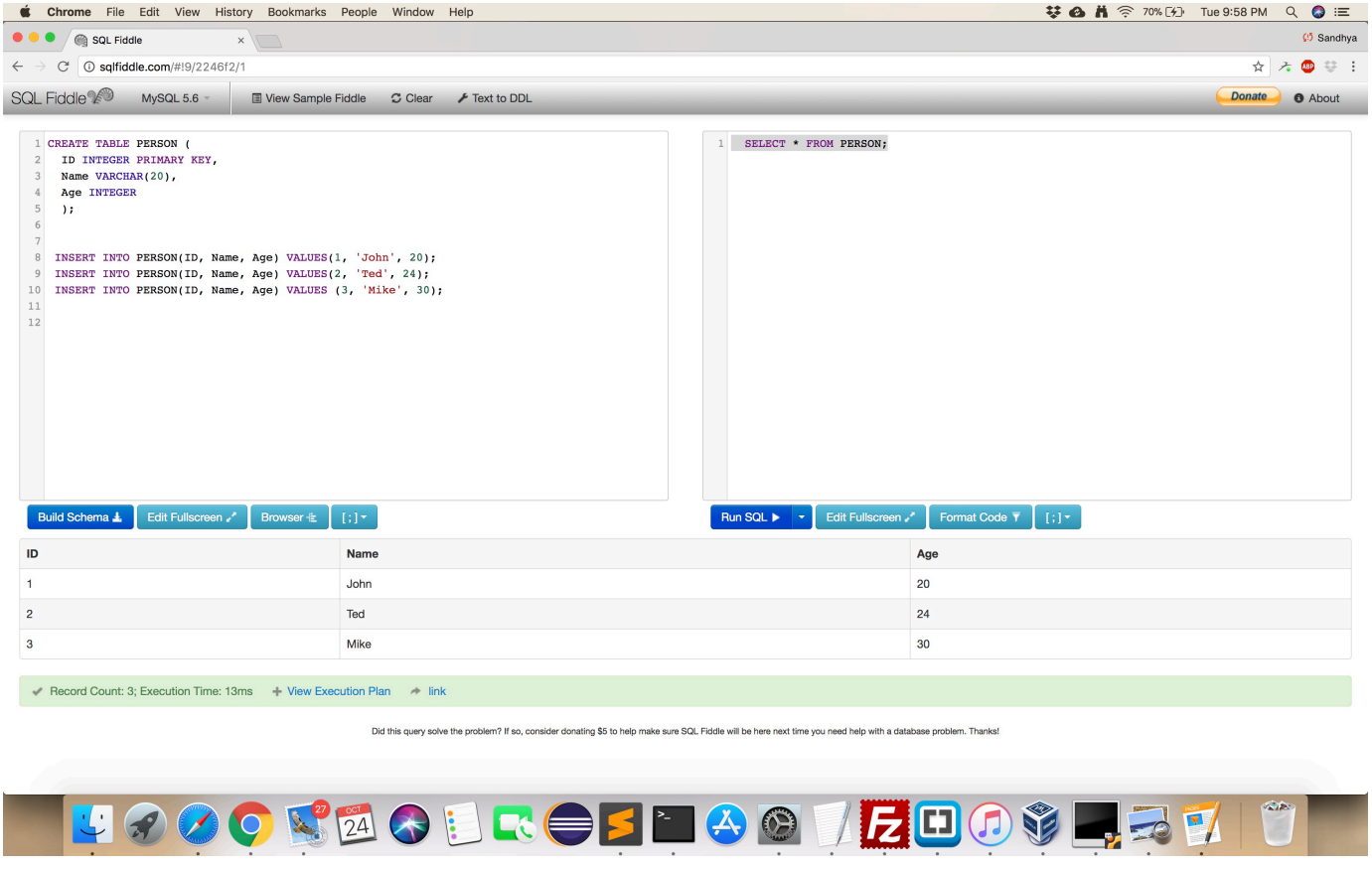
Solution:



3. use basic SELECT statement to show all the records in PERSON table (4 points)

Use the correct SQL statement to show all the records you have inserted into the PERSON table. Submit a screen copy showing what statement you have used and what is the result of the statement.

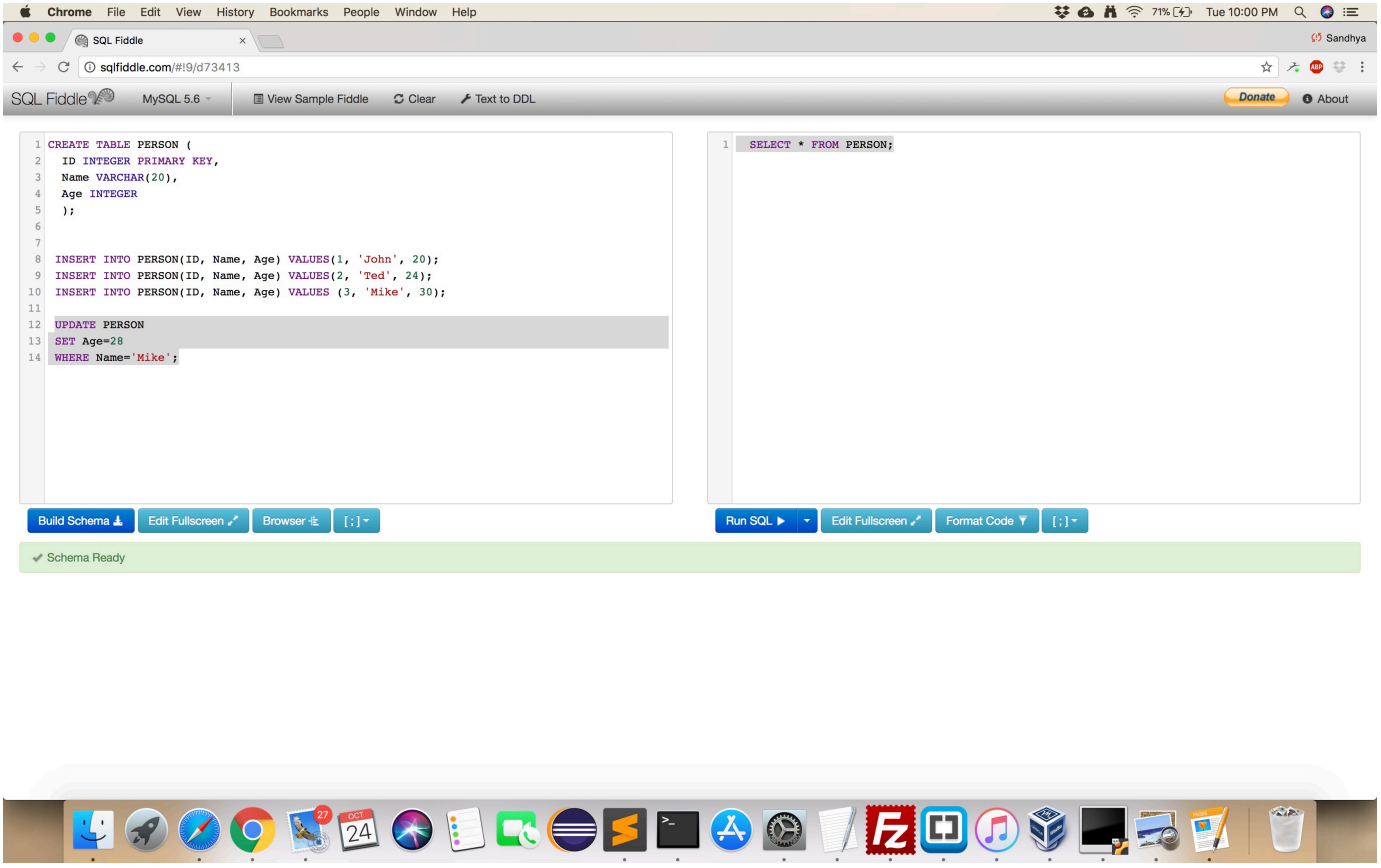
Solution:

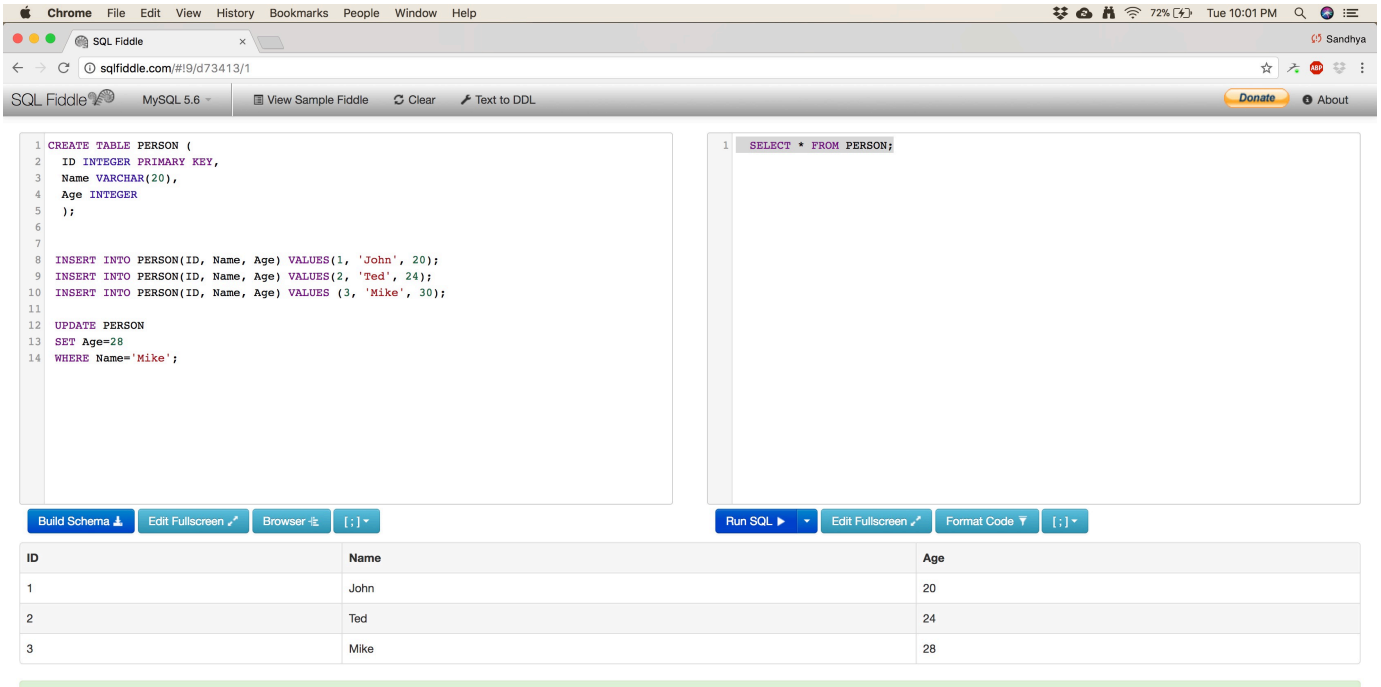


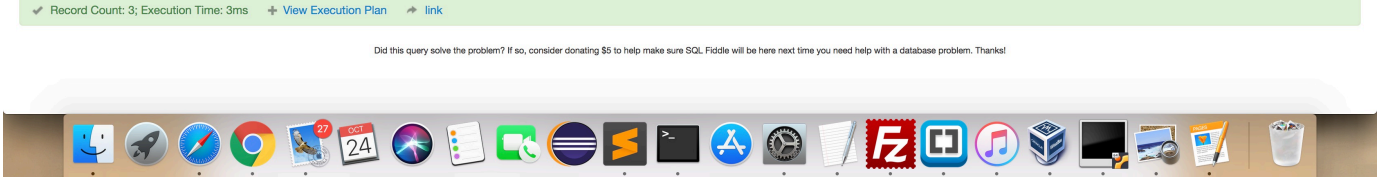
4. update one row. (2 points)

Use the correct SQL statement to update Mike's age, changing his age from 30 to 28. Submit a screen copy showing the statement you have used and the result of the update statement.

Solution:


Chrome File Edit View History Bookmarks People Window Help
Tue 10:00 PM
SQL Fiddle
Sandhya
sqlfiddle.com/#!9/d73413
SQL Fiddle
MySQL 5.6
View Sample Fiddle
Clear
Text to DDL
Donate
About
CREATE TABLE PERSON (
ID INTEGER PRIMARY KEY,
Name VARCHAR(20),
Age INTEGER
);
INSERT INTO PERSON(ID, Name, Age) VALUES(1, 'John', 20);
INSERT INTO PERSON(ID, Name, Age) VALUES(2, 'Ted', 24);
INSERT INTO PERSON(ID, Name, Age) VALUES (3, 'Mike', 30);
UPDATE PERSON
SET Age=28
WHERE Name='Mike';
SELECT * FROM PERSON;
Build Schema
Edit Fullscreen
Browser
[;]
Run SQL
Edit Fullscreen
Format Code
[;]
Schema Ready


Chrome File Edit View History Bookmarks People Window Help
Tue 10:01 PM
SQL Fiddle
Sandhya
sqlfiddle.com/#!9/d73413/1
SQL Fiddle
MySQL 5.6
View Sample Fiddle
Clear
Text to DDL
Donate
About
CREATE TABLE PERSON (
ID INTEGER PRIMARY KEY,
Name VARCHAR(20),
Age INTEGER
);
INSERT INTO PERSON(ID, Name, Age) VALUES(1, 'John', 20);
INSERT INTO PERSON(ID, Name, Age) VALUES(2, 'Ted', 24);
INSERT INTO PERSON(ID, Name, Age) VALUES (3, 'Mike', 30);
UPDATE PERSON
SET Age=28
WHERE Name='Mike';
SELECT * FROM PERSON;
Build Schema
Edit Fullscreen
Browser
[;]
Run SQL
Edit Fullscreen
Format Code
[;]
ID
Name
Age
1
John
20
2
Ted
24
3
Mike
28
Record Count: 3; Execution Time: 3ms
View Execution Plan
link
Did this query solve the problem? If so, consider donating $5 to help make sure SQL Fiddle will be here next time you need help with a database problem. Thanks!



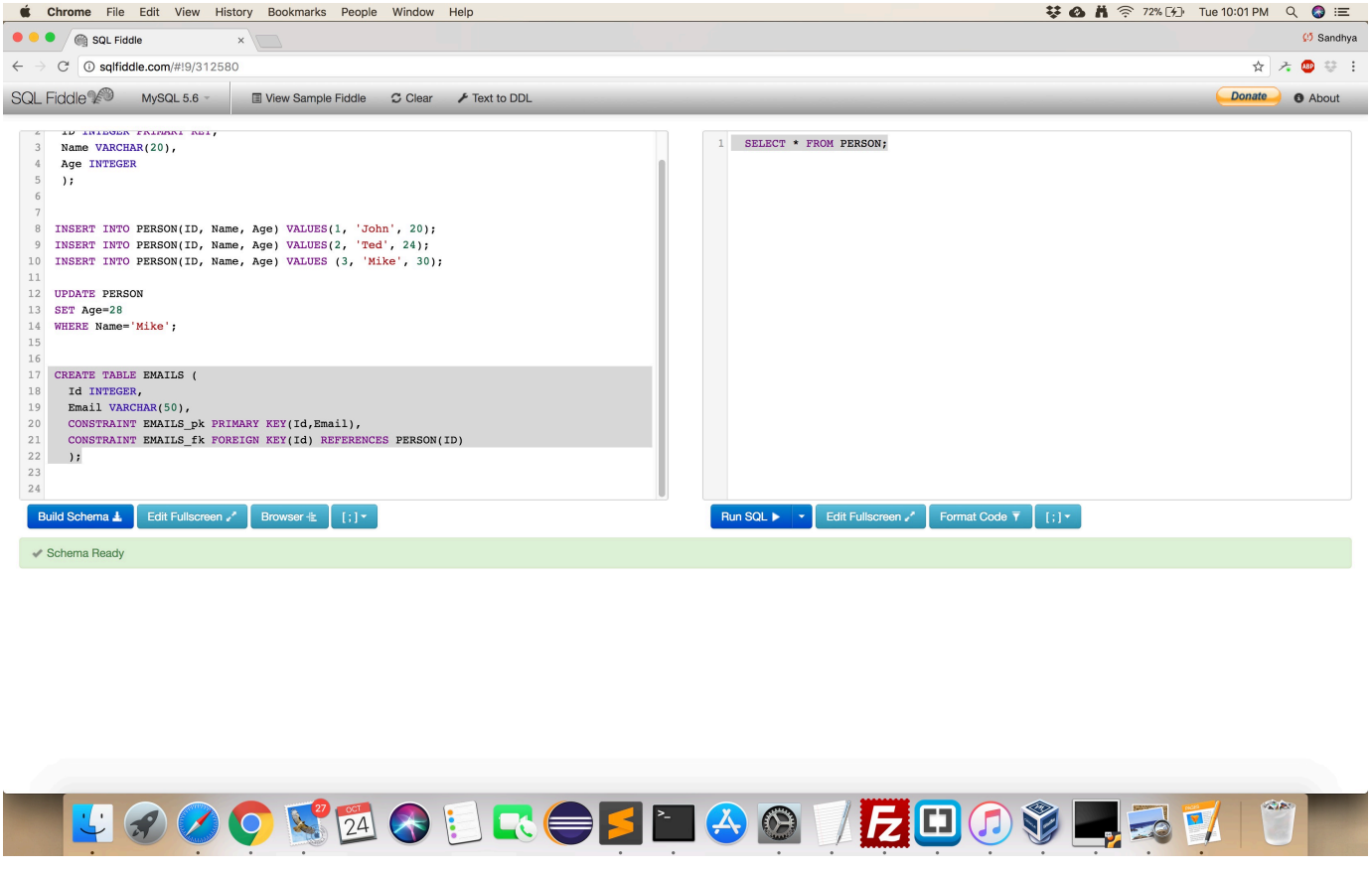
5. add another table (6 points)

Use the correct SQL statement to add another table, called EMAILS, which has the following attributes:

id: integer
email: varchar(50)
id + email: primary key, where id is a foreign key referencing the id attribute in PERSON table

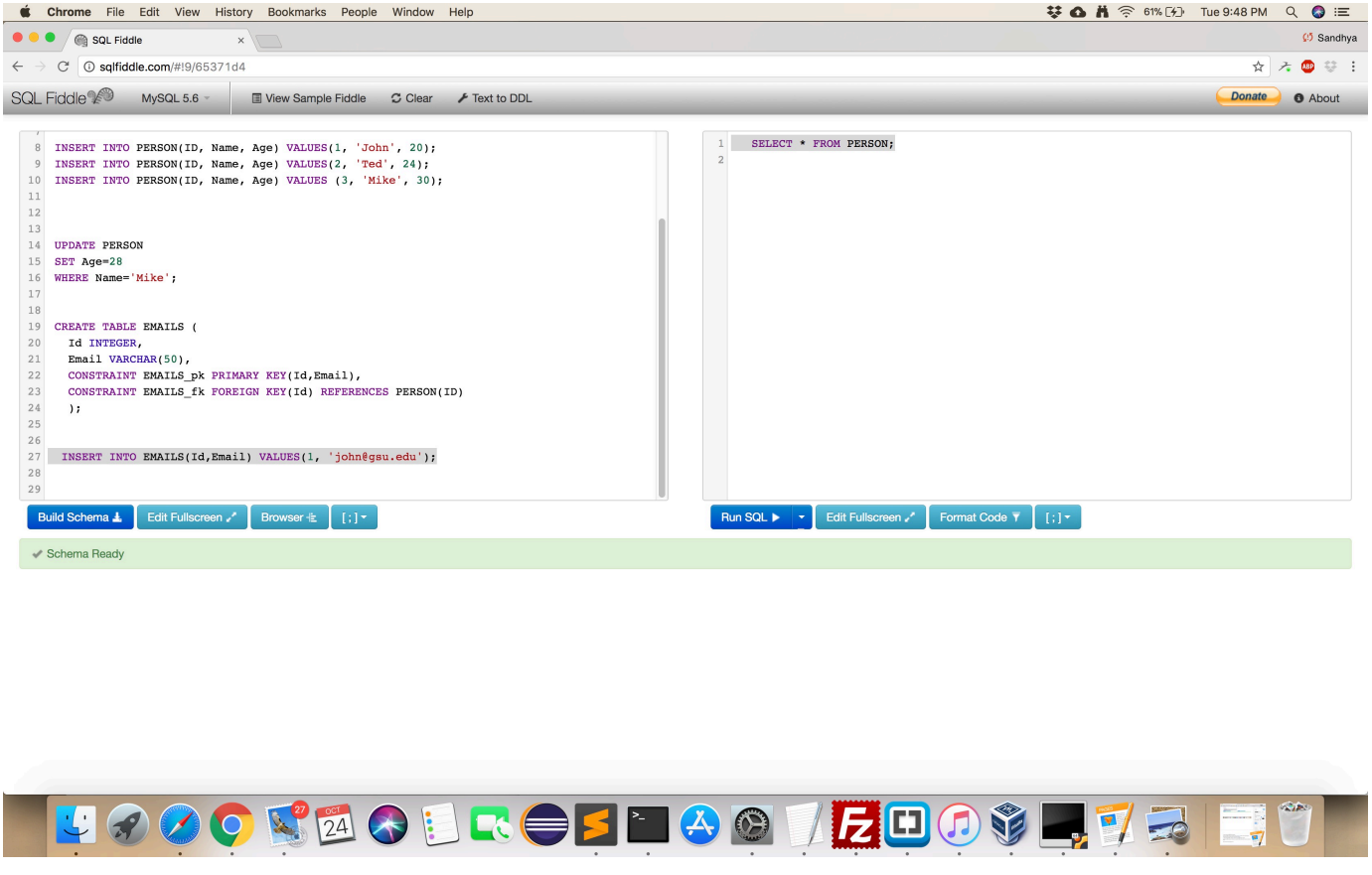
Submit a screen copy showing the statements you have used and the result of executing these statements.

Solution:



6. insert a record into EMAILS table (2 points)

Use the correct SQL statement to store John's email, john@gsu.edu. Submit a screen copy showing the statements you have used and the result of executing the statement.



7. finally, use basic SELET statement to show what is inside EMAIL table (4 points)

Use the correct SQL statement to show what is contained by EMAIL table, submit a screen copy showing the result.

Note: full score for this assignment is 64.