# Brandon Zink

# **CSCI 3287: Homework 1**

**Due Date**

9:55p Wed. Sep 20, 2017

No late submissions will be accepted.

**Delivery Method**

Submit a Microsoft Word or PDF document containing your answers using Moodle.

**Problem 1:** Matrix Manipulations

While it is not optimal to use SQL for matrix operations, implementation of some of the simple ones can get you thinking about attributes the right way.

Assume that a matrix is represented by an index/value representation, each tuple has the form: <*i*, *j*, *value*>. For example, a 2x2 identity matrix would look like this in a SQL table:

|  |  |  |
| --- | --- | --- |
| i | j | value |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

For each part, you should provide your query and sample input and output from a mysql run.

For all of part 1, the following tables will be used.

CREATE TABLE `A` (

`row` int(11),

`col` int(11),

`num` int(11)

);

INSERT INTO `A` (`row`, `col`, `num`) VALUES

(0, 1, 0),

(0, 0, 1),

(1, 1, 1),

(1, 0, 0);

CREATE TABLE `B` (

`row` int(11),

`col` int(11),

`num` int(11)

);

INSERT INTO `B` (`row`, `col`, `num`) VALUES

(0, 1, 0),

(0, 0, 1),

(1, 1, 1),

(1, 0, 0);

1. Write a SQL query that will add two 2-dimensional matrices of arbitrary but equal size, A and B.

SELECT A.row, B.row, A.num + B.num FROM A LEFT JOIN B on B.row = A.row AND B.col = A.col;

OUTPUT

|  |  |  |
| --- | --- | --- |
| row | col | A.num+B.num |
| 0 | 0 | 2 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 2 |

1. Write a SQL query that will multiply a matrix A by a scalar, *s*. **Do not** use an SFW query and it should be independent of the matrix dimensions.

SELECT A.row, A.row, A.num \* 3 FROM A;

OUTPUT

|  |  |  |
| --- | --- | --- |
| row | col | A.num\*3 |
| 0 | 0 | 3 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 3 |

1. Write a SQL query to multiply two matrices together. The first matrix *A* must have the same number of rows as the second matrix *B* has columns. Matrix multiplication is defined as *C*(*i*,*j*) = *A*(*i*,*k*) \* *B*(*k*,*j*).

SELECT A.row, B.col, SUM(A.num\*B.num)

FROM A, B

WHERE A.col = B.row

GROUP BY A.row, B.col;

OUTPUT

|  |  |  |
| --- | --- | --- |
| row | col | SUM(A.num\*B.num) |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**Problem 2:** The Pets Database

There is another way to import data into mysql. When you start mysql, you should use the **–-enable-local-infile** option:

**sudo mysql –-enable-local-infile –u root**

This allows you to import individual tables into a database. Create a new database called **pets**.

1. Create two tables:

**Pet(name:VARCHAR(20),owner:VARCHAR(20),**

**species:VARCHAR(20),sex:CHAR(1),birth:DATE,death:DATE)**

and

**Event(name:VARCHAR(20),date:DATE,type:VARCHAR(15),**

**remark:VARCHAR(255))**

1. Load data into these tables using the two file provided on Moodle, *Pet Table Data* and *Pet Events Table Data*. You can use the following command syntax example for *Pet Table Data*:

**mysql> LOAD DATA LOCAL INFILE “Pet Table Data.txt” INTO TABLE pet FIELDS TERMINATED BY ‘,’;**

1. Write a SQL query against this database. We’d like to see all the male cats and all the female dogs.

SELECT \* from Pet WHERE (species = 'dog' AND sex = 'f') OR (species = 'cat' AND sex = 'm');

OUTPUT

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| name | owner | species | sex | birth | death |
| Claws | Gwen | cat | m | 1994-03-17 | 0000-00-00 |
| Buffy | Harold | dog | f | 1989-05-13 | 0000-00-00 |

1. Write a SQL query that provides the age of each pet that has had a litter.

SELECT Pet.name, Floor(DATEDIFF(CURRENT\_DATE, birth)/365) as Age FROM Pet LEFT JOIN Event on Event.name = Pet.name WHERE type = 'litter';

The floor function calculates the age of the pet based on when it was born from the current date.

OUTPUT

|  |  |
| --- | --- |
| Name | Age |
| Fluffy | 24 |
| Buffy | 28 |
| Buffy | 28 |