CS 143 Discussion Session

Week1

Mingda Li

Today Topics

- Say hi~
- Introduction to Relational Algebra (Related to HW 1)
- Introduction to PHP and examples (Related to Proj 1)
- Other things related to Proj 1 like virtural machine

During Discussion:

- Lecture review
- Some information about the homework and project
- Some practice (no need to submit)
- If you have questions:
  - Check Piazza firstly
  - Email is not recommended if not very urgent
  - Office hour: Each Thursday 1:00 pm – 3:00 pm Bolter Hall 3551 Session K (If any change, will post on Piazza to let you know).

Remind

- Homework 1 has been released on CCLE.
- Due: Next Wednesday.
- About the students not enrolled in Class. how to submit the homework?

Introduction to Relational Algebra

- For this, you will need to use it to finish homework. And is necessary knowledge to know for midterm.
- What is an "Algebra":
- Algebra is a Mathematical System consisting of:
  - Operands --- variables or values from which new values can be constructed.
  - Operators --- symbols denoting procedures that construct new values from given values.

What is Relational Algebra?

- Relation Algebra is an algebra whose operands are relations or variables that represent relations.

- Operators are designed to do the most common things that we need to do with relations in a database.
  - The result is an algebra that can be used as a query language for relations.

Core Relational Algebra Operators

- Union, intersection, and difference.
  - Usual set operations, but both operands must have the same relation schema.
- Selection: picking certain rows.
- Projection: picking certain columns.
- Products and joins: compositions of relations.
- Renaming of relations and attributes.

Selection

- $R1 := \sigma_C (R2)$
  - C is a condition (as in "if" statements) that refers to attributes of R2.
  - R1 are all those tuples of R2 that satisfy C.

Example: Selection

Relation Sells:

bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

JoeMenu := $\sigma_{bar="Joe's"}(Sells)$:

bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75

Projection

- $R1 := \pi_L (R2)$
  - L is a list of attributes from the schema of R2.
  - R1 is constructed by looking at each tuple of R2, extracting the attributes on list L, in the order specified, and creating a tuple for R1 from those components.
  - Eliminate duplicate tuples, if any.

Example: Projection

Relation Sells:

bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

Prices := $\pi_{beer,price}$(Sells):

beer	price
Bud	2.50
Miller	2.75
Miller	3.00

Product (Cross product)

- R3 := R1 X R2
  - Pair each tuple t1 of R1 with each tuple t2 of R2.
  - Concatenation t1t2 is a tuple of R3.
  - Schema of R3 is the attributes of R1 and then R2, in order.
  - But beware attribute A of the same name in R1 and R2: use R1.A and R2.A.

Example: R3 := R1 X R2

R1(

A,	B
1	2
3	4

)

R2(

B,	C
5	6
7	8
9	10

)

R3(

A,	R1.B,	R2.B,	C
1	2	5	6
1	2	7	8
1	2	9	10
3	4	5	6
3	4	7	8
3	4	9	10

)

Theta-Join

- R3 := R1 $\bowtie_C$ R2
  - Take the product R1 X R2.
  - Then apply σ_C to the result.
- The σ_C can be any boolean-valued condition.
  - Historic versions of this operator allowed only A θ B, where θ is =, <, etc.; hence the name "theta-join."

Example: Theta Join

Sells(

bar,	beer,	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Coors	3.00

)

Bars(

name,	addr
Joe's	Maple St.
Sue's	River Rd.

)

BarInfo := Sells $\bowtie_{\text{Sells.bar = Bars.name}}$ Bars

BarInfo(

bar,	beer,	price,	name,	addr
Joe's	Bud	2.50	Joe's	Maple St.
Joe's	Miller	2.75	Joe's	Maple St.
Sue's	Bud	2.50	Sue's	River Rd.
Sue's	Coors	3.00	Sue's	River Rd.

)

Natural Join

- Natural join:
  - connects two relations by:
  - Equating attributes of the same name, and
  - Projecting out one copy of each pair of equated attributes.
  - Denoted R3 := R1 ⋈ R2.

Example for Natural Join

Sells(

bar,	beer,	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Coors	3.00

)

Bars(

bar,	addr
Joe's	Maple St.
Sue's	River Rd.

)

BarInfo := Sells ⋈ Bars

Note: Bars.name has become Bars.bar to make the natural join "work."

BarInfo(

bar,	beer,	price,	addr
Joe's	Bud	2.50	Maple St.
Joe's	Milller	2.75	Maple St.
Sue's	Bud	2.50	River Rd.
Sue's	Coors	3.00	River Rd.

)

Renaming

- The ρ operator gives a new schema to a relation.

R1 := $\rho_{R1(A1,\ldots,An)}$(R2) makes R1 be a relation with attributes A1,…,A*n* and the same tuples as R2.

Summary

- Relational Algebra
  - Used to query databases to find out answers
  - Relational Algebra is a Formal approach
    - While the Practical approach is SQL.
  - Set semantics: duplicates removed
    - SQL: bag semantics

PHP introduce

- PHP script starts with <?php and ends with ?>.
- One example:

```
<html>
<body>
 <?php
      echo "Hello World";

      /*
      I'm comment
      */
?>
</body>
</html>
```

PHP Variables

- In PHP, a variable starts with the $ sign, followed by the name of the variable:
  - $txt="Hello World";
  - concatenation operator (.)
    - <?php
      $txt1="Hello World";
      $txt2="1234";
      echo $txt1 . " " . $txt2;
      ?>

Operators

Arithmetic

- **+, -, *, /, %, ++, --**

Assignment

- **=, +=, -=, *=, /=, .=, %=**

Comparison

- ==, !=, >, <, >=, <=

Logical

- &&, ||, !

If…else

if (*condition*)

- *code to be executed if condition is true;*

elseif (*condition*)

- *code to be executed if condition is true;*

else

- *code to be executed if condition is false;*

Example for if else

```
<html>
<body>

<?php
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
elseif ($d=="Sun")
  echo "Have a nice Sunday!";
else
  echo "Have a nice day!";
?>

</body>
</html>
```

switch

- Use the switch statement to select one of many blocks of code to be executed.

```
<?php
switch ($x)
{
case 1:
  echo "Number 1";
  break;
case 2:
  echo "Number 2";
  break;
case 3:
  echo "Number 3";
  break;
default:
  echo "No number between 1 and 3";
}
?>
```

Arrays

```
$names = array("Peter","Quagmire","Joe");
$names[0] = "Peter";
$names[1] = "Quagmire";
$names[2] = "Joe";
```

Associative Arrays

- Associative arrays are arrays that use named keys that you assign to them.

```
<?php
    $ages['Peter'] = "32";
    $ages['Quagmire'] = "30";
    $ages['Joe'] = "34";

    echo "Peter is " . $ages['Peter'] . " years old.";
?>
```

Loop: while

```
<?php
    $i=1;
    while($i<=5)
     {
     echo "The number is " . $i . "<br />";
     $i++;
     }
?>
```

Loop: for

- for (*init*; *cond*; *incr*)
  { *code to be executed;* }
- Example

```
<html>
<body>

<?php
for ($i=1; $i<=5; $i++)
{
  echo "Hello World!<br />";
}
?>

</body>
</html>
```

functions

- function functionName() {
- code to be executed;
- }

```
<?php
    function writeMyName()
     {
     echo "Kai Jim Refsnes";
     }

    echo "Hello world!<br />";
    echo "My name is ";
    writeMyName();
    echo ".<br />That's right, ";
    writeMyName();
    echo " is my name.";
?>
```

PHP Form Handling

- 1. HTML Form:
- The HTML <form> element defines a form that is used to collect user input:
- <form>
- form elements
- </form>
- 1) The <input> Element:
- E.g. <input type="text"> Defines a one-line text input field

- E.g.

```
<form>
  First name:<br>
  <input type="text"
name="firstname"><br>
  Last name:<br>
  <input type="text" name="lastname">
</form>
```

Submit Button

- <input type="submit"> defines a button for submitting the form data to a form-handler.

- The form-handler is specified in the form's action attribute:

E.g.


```
<form action="/action_page.php">
  First name:<br>
  <input type="text" name="firstname" value="Mickey"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Mouse"><br><br>
  <input type="submit" value="Submit">
</form>
```

The Action Attribute

The action attribute defines the action to be performed when the form is submitted.

<form action="/action_page.php">

In the example above, the form data is sent to a page on the server called "/action_page.php". This page contains a server-side script that handles the form data:

The Method Attribute:

The method attribute specifies the HTTP method (GET or POST) to be used when submitting the form data:
<form action="/action_page.php" method="get">

PHP get the data

handle.pjp

```
<html>
<body>

<form action="handle.php" method="GET">
Name: <input type="text" name="name" />
Age: <input type="text" name="age" />
<input type="submit" />
</form>
<?php
if($_GET["name"]){
    echo "Welcome ".$_GET["name"]."<br />";
    echo "You are ".$_GET["age"]." years old.";
}
?>
</body>
</html>
```

When the user fills out the form above and clicks the submit button, the form data is sent for processing to a PHP file named ”handle.php“. The form data is sent with the HTTP GET method.(default is GET)

Then, in PHP, the data is collected:
$_GET[“”] and
$_POST[“”]

Here the _GET is used.

Project 1 related

- 1. Project 1 will be released soon.

Team

- Form the team to do the projects.
- (At most two, one person is also ok)

- Attach "team.txt" in project submission
- "Marry once, Divorce once" (we'll check)
  - Form team starting from any time if you'd like.
  - But if a team is separated, both members can not team up (with other students) again!

VM Installation

- Oracle Virtual box (Ver >= 4.0.2)
- CS143.ova (virtual machine image)
- Make sure you follow instructions on the class website to avoid common issues.

VM Issues

- Make sure to follow instructions **exactly**.
  - http://yellowstone.cs.ucla.edu/cs143/project/virtualbox/index.html

- Common pitfalls
  - Folder Path: select the shared folder that you created (on host machine).
  - Folder Name: **vm-shared**.
  - ~~"Read-only"~~ "Auto-mount"