Virginia Tech. **CS 4604 – Introduction to DBMS**
Computer Science **Spring 2013, Prakash**

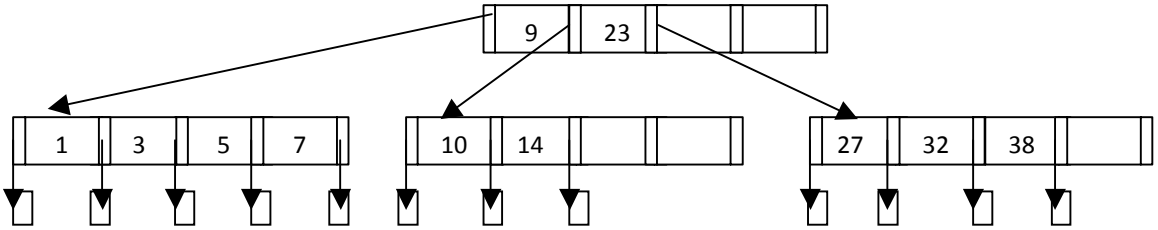
Homework 5: Miscellanea
(due April 26th, 2013, 9:05am, in class—hard-copy please)

Reminders:

a. Out of 100 points.
b. Rough time-estimates: ~4-6 hours.
c. Please type your answers. Illegible handwriting may get no points, at the discretion of the grader. Only drawings may be hand-drawn, as long as they are neat and legible.
d. There could be more than one correct answer. We shall accept them all.
e. Whenever you are making an assumption, please state it clearly.

Q1: B-Tree [15 points]

Assume the following B-tree exists with d = 2:



(3x5=15 points) Sketch the state of the B-tree after each step in the following sequence of insertions and deletions:

Insert 28, Insert 4, Delete 38, Delete 14, Delete 9

Note: Use the insertion and deletion algorithms given in the lecture slides.

Q2: MDs and 4NF [15 points]

Consider the relation R(A, B, C, D) with the following set of dependencies:

$$A\,B \rightarrow C,\ C \rightarrow D,\ D \twoheadrightarrow B$$

Q2.1 (6 points) Use the chase process to prove that the MD $C \twoheadrightarrow B$ also holds.

Q2.2 (9 points) Is this relation in 4NF? Explain your answer: if it is not in 4NF, decompose it into a set of 4NF relations. Show your steps and make sure your decomposition is lossless.

Q3: Transactions [14 points]

Consider the following Schedule #1:

Transaction 1	Transaction 2
Read(A)	
Read(B)	
	Read(A)
	Read(B)
	A = A -10
	B = B / 5
	Write(A)
A = A + 10 * B	
Write(A)	
	Write(B)
Write(B)	

Answer the following questions (in each case explain your answer). Let's assume A = 10 and B = 5, before Schedule #1 is executed.

Q3.1 (2 points) What are the values of A and B at the end of Schedule #1?

Q3.2 (2 points) Is this schedule conflict serializable?

Consider the following (different) Schedule #2:

Transaction 1	Transaction 2	Transaction 3
	Read(B)	
		Read(C)
	Write(A)	
Read(A)		
		Write(C)
	Write(B)	
Read(C)		
	Read(C)	
	Commit	
Write(A)		
Commit		
		Abort

Answer the following questions (in each case explain your answer). Assume every Read() is lock-shared and every Write() is lock-Exclusive.

Q3.3 (3 points) Is Schedule# 2 conflict serializable? Please draw the dependency graph of this schedule.

Q3.4 (2 points) Is this schedule cascadeless (i.e. does not have cascading aborts)?

Q3.5 (2 points) Is this schedule possible under a (non-strict) 2PL protocol?

Q3.6 (2 points) Is this schedule possible under a strict 2PL protocol?

Q4: XML [12 points]

Q4.1 (4 points) Write the XML Schema definition of Fig. 11.20 (page 513 in textbook) as a DTD.

Q4.2 (5 points) Consider the following table:

pizza_name	price	ingredient_name	amount
The Works	15	Flour	3
The Works	15	Sausage	4
The Works	15	Tomato	4
The Meats	12	Tomato	3
The Meats	12	Pepperoni	4
Pepperoni	10	Pepperoni	2
Pepperoni	10	Flour	6

The table records the pizza name, price, which ingredient is used in a particular pizza, and how much ingredient is used (recall the DB-Pizza Store example in Homework #1). Translate it to **pizzas.xml**, like in Figure 12.7 (Page 532 in textbook).

Q4.3 (3 points) Write the following query in XQuery: Return the names of pizzas that contain 4 units of the ingredient "Tomato".

Q5: Data Mining and Warehousing [12 points]

Data Mining is the examination of data for patterns or trends. For example, by analyzing the sales data in a company like Amazon, we can understand which item was the most profitable during a particular period; which is the most popular color for a given dress; and so on. Let us consider the following relation:

clothes_sale (clothes_type, category, size, color, season, amount)

In this relation, every tuple has a clothes_type (e.g., "skirt", "dress", and "shirt"), category (e.g., "men's wear", "women's wear", and"kid's wear"), size (e.g., "Large", "Medium", "Small"), color (e.g., "Red"), season (e.g., "Winter"), and amount (e.g., "20") that is how many items are sold.

Q5.1 (2 points) Write the SQL statement to create the CUBE(clothes_sale) by constructing a MATERIALIZED VIEW.

Q5.2 (3 points) What is the ratio of the size of CUBE(clothes_sale) to the size of clothes_sale? Assume that the fact table clothes_sale has 5 dimensions each with 5 different values. Also assume that there are 500,000 tuples in clothes_sale.

Q5.3 (7 points) Answer the following queries by using the materialized view you created before in Q5.1.

Q5.3.1 (2 points) Find the total sales of "red shirt" for each season.

Q5.3.2 (3 points) Find the most popular color for each clothes_type.

Q5.3.3 (2 points) Find which clothes_type is sold most for each category.

Q6: Query Optimization (Pen-n-Paper) [18 points]

Consider the following relational design (the primary key of the relations have been underlined):

Product (pid, pname, price, description)
Customer (cid, cname, address, city, phone)
Order (cid, pid, orderdate, quantity)

Assume that there are 400,000 different products, 20 million customers, and 1 billion order records. Furthermore, we assume there are 20,000 cities, the maximum price is 10,000 dollars and the minimum price is 10 dollars.

Consider the following three queries on the given relations:

Query 1:

```
SELECT c.cid, cname
FROM Customer as c, Order as o, Product as P
WHERE c.cid = o.cid
  AND o.pid = p.pid
  AND city = "Blacksburg"
  AND pname = "FIFA 2012";
```

Query 2:

```
SELECT o.cid, o.pid, orderdate
FROM Customer as c, Product as p, Order as o
WHERE c.cid = o.cid
  AND o.pid = p.pid
  AND price > 1000;
```

Query 3:

```
SELECT cname, city
FROM Customer as c, Order as o, Product as p
WHERE c1.cid = o.cid
  AND o.pid = p.pid
  AND pname = "Nike Air Force"
  OR pname = "Adidas Soccer";
```

Q6.1 (1x3=3 points) Describe in plain English what each query does.

Q6.2 (3x3=9 points) Translate the three queries each into a left-deep-join parse tree, and then transform them to canonical form by using the transformation rules given in the lecture slides. Make sure you mention exactly which rules you apply. Note that your canonical form for each query should be a left-deep-join tree too.

Q6.3 (2x3=6 points) Compute the selectivity of the predicates in the canonical form of each query you get in Q6.2. Show your steps.

Q7: Query Optimization (Hands-on) [14 points]

Consider the following two relations:

hw5_pub (id, title, type, ee, ee_pdf, titlesignature, dblp_key, publisher, doi)
hw5_author_pub (id, author, authornum)

These two tables are very similar to the dblp_pub_new and dblp_author_ref_new tables of your project. In the table "hw5_pub", id is the identifier of the publication, type is the type of publications, ee is url of publications and so on. In the table hw5_author_pub, "author" is author's name and "authornum" is the rank of this author in a particular publication.

We have created sample instances of these two tables on the cs4604.cs.vt.edu server. Use the following commands to copy the tables to your *private* database:

- pg_dump -U YOUR-PID -t hw5_pub s13dblp | psql -d YOUR-PID
- pg_dump -U YOUR-PID -t hw5_author_pub s13dblp | psql -d YOUR-PID

Then answer the following questions:

Q7.1 (3 points) What is the number of disk pages occupied by the two tables respectively? Also write down the query you used to find this information.

Q7.2 (4 points) Consider the following query which lists the paper titles and years of the authors who are collaborators of "Philip S. Yu":

Query 4:

```
SELECT title, year
FROM hw5_author_pub AS t1, hw5_author_pub AS t2,
        hw5_author_pub AS t3, hw5_pub AS t4
WHERE t1.author = 'Philip S. Yu'
   AND t1.id = t2.id
   AND t2.author <> 'Philip S. Yu'
   AND t2.author = t3.author
   AND t3.id = t4.id;
```

Use the EXPLAIN command to get the query plan returned by the optimizer for *Query 4*. Copy-paste the output. In addition, draw the plan using the tree/relational algebra notation as in the lecture slides (see slides #61-62, lecture 19). What is the estimated cardinality of the result of this query? Next, run the query and report the number of rows actually returned. Also report the total runtime of the query.

Q7.3 (2 points) Create an index on the attribute "author" in the table "hw5_author_pub" and an index on the attribute "id" in the table "hw5_pub". Write the SQL commands you used. Update the statistics by running VACUUM and then ANALYZE.

Q7.4 (5 points) Use the EXPLAIN command again on *Query 4* (after doing Q7.3), copy-paste the output and also draw the query plan returned in the tree/relational algebra notation. What is the estimated result cardinality? Report the total runtime of the query as well. Has the performance of *Query 4* improved (compared to before you had the indexes)?

Hints:

1. Check the statistics collected by PostgreSQL:
   http://www.postgresql.org/docs/8.4/static/planner-stats.html

2. How to use EXPLAIN command and understand its output:
   http://www.postgresql.org/docs/8.4/static/sql-explain.html
   http://www.postgresql.org/docs/8.4/static/performance-tips.html

3. You can use EXPLAIN ANALYZE to get the runtime of a query.