Name_____

Class Account:

UNIVERSITY OF CALIFORNIA
Department of EECS, Computer Science Division

CS186 Hellerstein/Olston
Fall 2006 Midterm Exam

Midterm Exam: Introduction to Database Systems

This exam has four problems and one extra credit question, worth different amounts of points each. Each problem is made up of multiple questions. You should read through the exam quickly and plan your time-management accordingly. Before beginning to answer a problem, be sure to read it carefully and to *answer all parts of every problem!*

You **must** write your answers on the exam. Two pages of extra answer space have been provided at the back in case you run out of space while answering. If you run out of space, be sure to make a "forward reference" to the page number where your answer continues.

Good luck!

1. Relational Query Languages [25 points]

Recently, Bob the Builder coded up a Relational Query Language Translator (RQLT) and passed it along to his lesser-known colleague, Ted the Software Test Engineer, for testing. Fortunately for Ted, the RQLT is simple and only operates on a single schema:

`Tool(tid, brand, cost)`
`Jobsite(location, compensation, task)`
`Toolbox(tbid, location)` – location is a foreign key to Jobsite
`Holds(tbid, tid)` – tbid is a foreign key to Toolbox, tid is a foreign key to Tool.

a. *(5 points)* Ted sets the RQLT to translate the following SQL query to Relational Algebra:

```
SELECT T.tid
FROM Tool T, Holds H, Toolbox B, Jobsite J
WHERE T.tid = H.tid AND H.tbid = B.tbid
AND B.location = J.location AND J.task = 'Plumbing'
```

Which of the following would be an equivalent Relational Algebra query?

1. $\pi_{Ttid}(\rho(C(1 \rightarrow Ttid),$ $(Tool \bowtie Holds \bowtie Toolbox \bowtie \sigma_{task = \text{'Plumbing'}}(\text{Jobsite}))))$
2. $\pi_{T.tid}(\sigma_{task = \text{'Plumbing'}}(\rho(C(1 \rightarrow Ttid),$ $Tool \bowtie (Holds \bowtie Toolbox) \bowtie \text{Jobsite}))$
3. $\sigma_{task = \text{'Plumbing'}}(\pi_{tid}(Tool) \bowtie Holds \bowtie Toolbox \bowtie \text{Jobsite})$
4. A and B
5. A, B, and C
6. None of the above

YOUR ANSWER HERE (1a): ___5_____

b. *(10 points)* Ted switches the RQLT to translate Relational Algebra into Relational Calculus and passes in:

$$\pi_{tbid}(Toolbox) - \pi_{tbid}(\quad (\pi_{tbid}(Toolbox) \times \pi_{brand}(Tool)) \\ - \pi_{tbid,brand}(Holds \bowtie Tool) \quad)$$

Fill in the blanks to complete the equivalent Relational Calculus Query. *Note:* the correct answer may not require all blanks to be filled in.

$\{ B1 \mid B \in Toolbox$ ____∧_____ $B.tbid = B1.tbid$ ____∧_____

____∀_____ $T \in Tool$

(____∃_____ $H \in Holds\ (T.tid = H.tid$ ____∧_____ $H.tbid = B.tbid)) \}$

c. *(5 points)* Ted thinks that he has found a bug in the RQLT. It translates the following Relational Calculus query:

$\{ H1 \mid H \in Holds \wedge H1.tbid = H.tbid \wedge$
$\forall\ T \in Tool\ (\ (H.tid = T.tid)$
$\Rightarrow (T.brand = \text{'Craftsman'} \vee T.brand = \text{'Channellock'})) \}$

into the following incorrect SQL query (marked with line numbers for the purpose of this question):

```
1. SELECT H.tbid
2.   FROM Tool T, Holds H
3.  WHERE T.tid = H.tid AND T.brand = 'Craftsman'
4. INTERSECT
5. SELECT H1.tbid
6.   FROM Tool T1, Holds H1
7.  WHERE T1.tid = H1.tid AND T1.brand = 'Channellock';
```

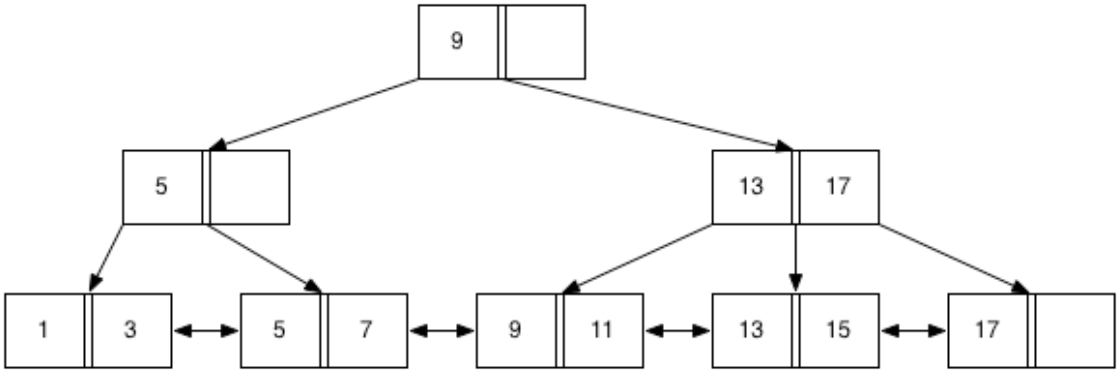
Fix this SQL query to match the Relational Calculus query by changing exactly ONE line:

Answer: Change line ___4___ into:

_____UNION_____

Name_____

2. **B+ Trees [22 points]**
Consider the following B+ tree index of order 1:



Assume that upon split, two entries stay on the left, and one stays on the right.

a. *(4 points)* Circle all *nodes* (not index entries, but entire nodes) in the above figure that must be fetched to satisfy the query "Get all records with search key greater than or equal to 7 and less than 15".

b. *(8 points)* Assume we modify the B+ tree by adding the following keys **in the following order**: **20, 27, 18, 30, 19**

In the answer-boxes below, each row refers to a key being inserted in order, and each column asks if the insertion of that key results in a split of particular nodes. Place a Y mark in each box whose answer is "Yes". Blank boxes will be interpreted as "No".

Key	Leaf Node Split?	Non-Leaf Split?	Root Split?
20			
27	Y	Y	
18			
30			
19			

c. *(10 points)* Suppose, given the tree in part (b), we were to insert all integers in the range 31 to 144, inclusive (i.e. 31, 32, 33, ..., 143, 144), one at a time. At most how many levels would the resulting B+-tree have? (Hint: You should not need to draw a B+-tree to figure this out!)

YOUR ANSWER HERE (2c): _____

Name_____

3. Query Optimization [38 points]
Consider the following schema:

```
Guitars (gid, brand, price)
Players (pid, name, age)
LastPlayed (gid, pid, date)
```

And the following query:

```
SELECT P.name
  FROM Guitars G, Played P, LastPlayed L
 WHERE G.gid = L.gid AND P.pid = L.pid
   AND P.age < 25 AND G.brand = 'Gibson'
   AND G.price > 3000;
```

In the schema, Guitars.gid is the primary key of Guitars, Played.pid is the primary key of Played, and LastPlayed.gid and LastPlayed.pid are foreign keys referencing the primary keys of Guitars and Players (respectively). Assume that the data is evenly distributed, and that the following properties hold:

Players.age ranges from 10 to 85
Guitars.brand has 15 distinct values
Guitars.price ranges from 1,000 to 5,000
Guitars.gid has 1,000 distinct values
Players.pid has 15,000 distinct values

a. *(12 points)* Computer the selectivity for each individual term in the WHERE clause, then provide a final selectivity estimate for what fraction of the total tuples will appear in the output. You must fill in each blank below!

G.gid = L.gid : _____

P.pid = L.pid : _____

P.age < 25 : _____

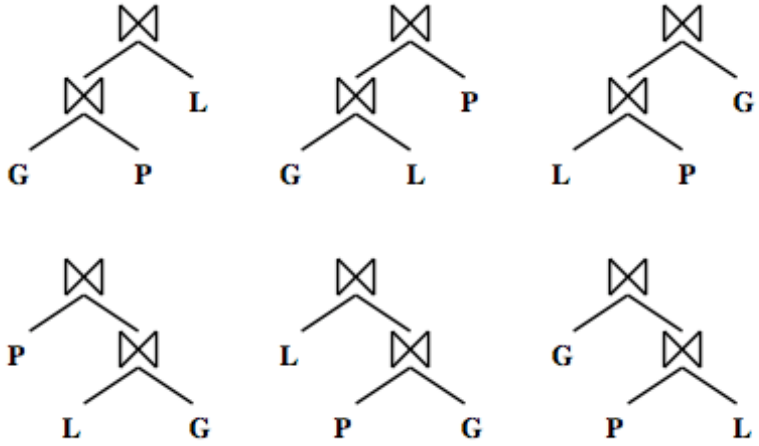
G.brand = 'Gibson' : _____

G.price > 3000 : _____

TOTAL : _____

Name_____

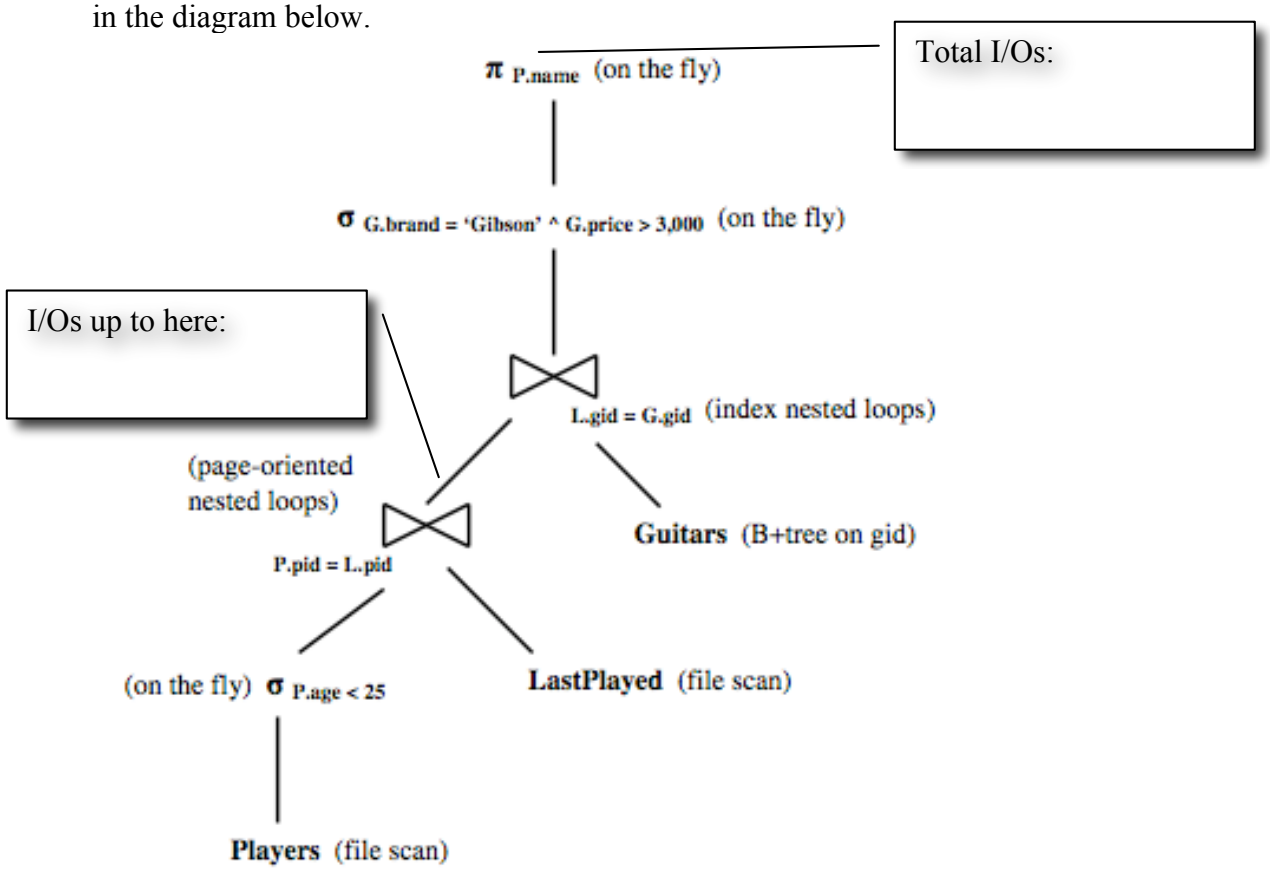
b. *(6 points)* According to the System R query optimizer that we studied, circle all the following join orders that would be considered:



c. *(10 points)* Now consider the following. There is a B+ tree index on Guitars.gid – it is unclustered and it uses Alternative 2 to represent data entries. Assume that, on average, it takes 3 I/Os to retrieve a given data entry, and that the following properties hold:

Guitars →	40 bytes/tuple, 100 tuples/page, 10 pages
Players →	80 bytes/tuple, 50 tuples/page, 300 pages
Lastplayed →	25 bytes/tuple, 150 tuples/page, 90,000 pages

Assuming *no buffering occurs*, fill in the two blanks for the number of I/Os spent in the diagram below.



Name_____

d. *(10 points)* Consider the plan from part (c) again. For each of the following changes to it, mark the line "Y" (yes) or "N" (no) depending on whether it could have led to a further reduction in the number of I/Os. Consider each change individually! Assume that any cost associated with setting up the described change is *not* included in the execution cost, and that buffering is now being used.

_____ Pushing down the selection on G.brand and G.price below the join

_____ Creating a temporary file to store the results of the selection on P.age

_____ Having the index on Guitars.gid be clustered

_____ Projecting out Players.age and Players.date before the join

_____ Changing the first join to block-oriented nested loops join.

4. Query Execution [25 points]

You have been hired to advise a major hot-dog vending franchise called "Dunce Dog" to tune their database server installation. They run a commercial DBMS called Tentacle version 9y. One of the startup parameters in Tentacle is called `query_space`; it tells Tentacle how many disk-blocks worth of RAM it should allocate to use for sorting and hash joins. Note that memory used for `query_space` is separate from the Tentacle buffer pool.

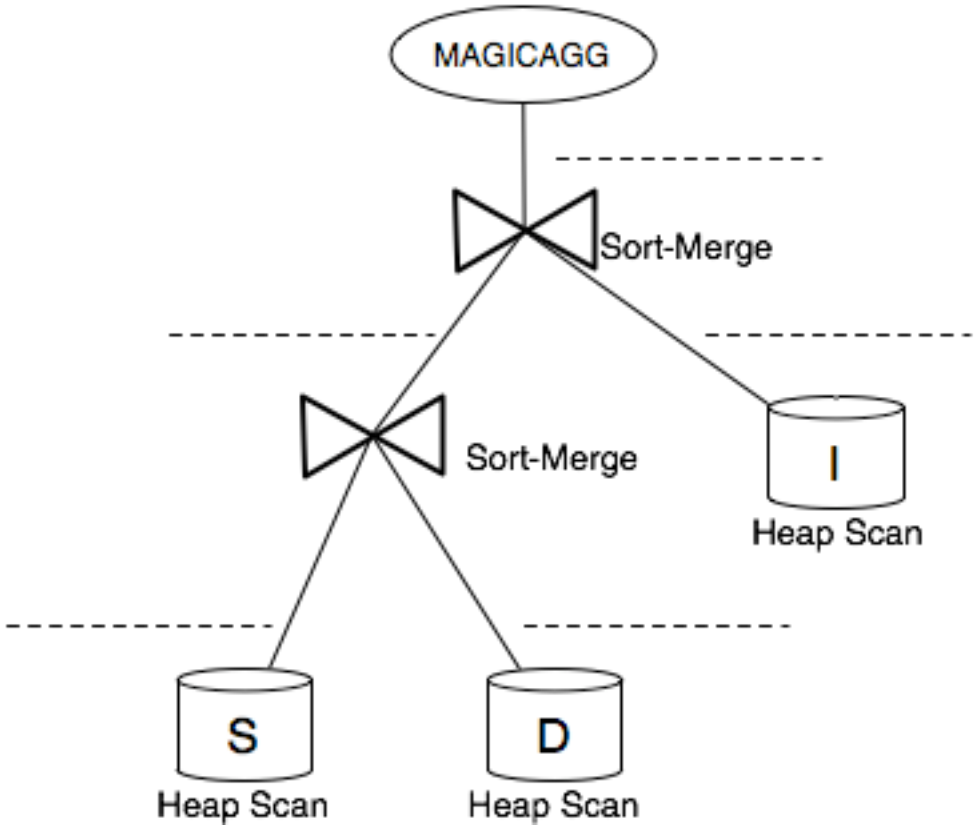
Dunce Dog is having problems running one of their monthly reporting queries. It uses the following tables:

Store(sid, location, owner)
ItemsForSale(iid, name, description, cost, price)
DetailedSales(receiptno, iid, sid)

The query is:

```
SELECT S.sid, S.location, SUM(I.price - I.cost) AS PROFIT
  FROM Stores S, ItemsForSale I, DetailedSales D
 WHERE S.sid = D.sid and D.iid = I.iid
GROUP BY S.sid, S.location
```

You determine that the query plan that Tentacle chooses is:



Name_____

a) *(5 points)* Tentacle performs aggressive projection – that is, it discards any attributes that it does not need as soon as possible. For each edge in the query plan, write down the smallest list of attributes that needs to be *retained* (i.e. not discarded) on the corresponding dotted line.

b) *(5 points)* Given your answer to (a), assume that the result of the scan of ItemsForSale fits in 1000 pages, and the result of the scan of DetailedSales fits in 10,000 pages. Approximately how many blocks of `query_space` should Tentacle need *at minimum* to perform the join of ItemsForSale and DetailedSales in two passes? Feel free to round up or down by as many as 2 blocks in any equations that you use, but if you do so, show your equations!

c) *(5 points)* Given your answer to (a), assume that the result of the first join fits in 2,500 pages, and the result of the scan of Stores fits in 400 pages. Approximating and showing work as in part (b), approximate the number of blocks of `query_space` that Tentacle should need to perform the second join in the plan.

d) *(5 points)* How much memory should Tentacle's MAGICAGG operator need to perform GROUP BY and aggregation efficiently? Again, feel free to round up or down by as many as 2 blocks.

e) *(5 points)* Given your answers to (b), (c), and (d) above, write down a formula to describe how you should compute the minimum value of `query_space` that your client should use. Please treat your answers to (b), (c) and (d) as variables *b, c,* and *d,* and write the formula in terms of those variables; this way you can get a correct answer here even if for some unlikely reason you have an error above.

Name_____

EXTRA CREDIT

Name_____

EXTRA JUNK

d. In preparation for the next phase of testing, Ted diligently prepares flashcards with queries, taking special care to group the equivalent queries. Unfortunately, as fate would have it, Ted trips and drops the flashcards just as he is about to pass the first group of queries into the RQLT, scrambling the ordering of the cards in the process.

Ted: Oh no, I've dropped the cards. *Can we fix it?*
CS186 Students: *Yes we can!*

Help Ted re-create groups of one or more equivalent queries from the following (labeled with letters):

A. SELECT DISTINCT J.location FROM Jobsite J, Tool T WHERE J.compensation < T.cost;

B. SELECT DISTINCT J.location FROM Jobsite J, Tool T WHERE J.compensation > T.cost;

C. $\{ J1 \mid J \in Jobsite \wedge J1.location = J.location \wedge \forall T \in Tool\ (J.compensation > T.cost) \}$

D. $\{ J1 \mid J \in Jobsite \wedge J1.location = J.location \wedge \exists T \in Tool\ (J.compensation < T.cost) \}$

E. $\pi_{location}\ (\ Jobsite \bowtie_{Jobsite.compensation < Tool.cost} Tool\ )$

F. $\pi_{location}\ (\ Jobsite \bowtie_{Jobsite.compensation > Tool.cost} Tool\ )$