CISC437/637 Database Systems Final Exam

You have from 1:00 to 3:00pm to complete the following questions. The exam is closed-note and closed-book. Good luck!

**Multiple Choice** (2 points each; 52 total)
x marks the answer.

1. If the SQL statement `SELECT C1, C2, C3 FROM T4 WHERE C2='Smith'` is frequently executed, which column(s) should be considered for indexing based only on the statement itself?
   (a) C1 only 
   (b) x C2 only
   (c) C3 only
   (d) C1 and C2
   (e) C1, C2, and C3

2. If the SQL statement `SELECT R1.A, R2.B FROM R1, R2 WHERE R1.K = R2.F AND R2.K = 10` is frequently executed, which indexes will prove most useful?
   (a) x Index on R1.K and index on R2.K
   (b) Index on R1.A and index on R2.B
   (c) Index on R1.K and index on R2.F
   (d) Composite index on (R2.K, R2.F)

3. What can indexes accomplish (compared to having no index)?
   (a) Reduce the total number of disk pages read during query processing.
   (b) Reduce the total disk seek time during query processing.
   (c) Reduce the total number of disk pages written during updates.
   (d) x a and b
   (e) All of the above

4. Which of the following is true?
   (a) x Primary indexes are always on primary keys
   (b) Secondary indexes are always on foreign keys
   (c) Primary indexes are always clustered
   (d) a and b
   (e) none of the above

5. Which index or storage type is ideal for the query `SELECT C1 FROM R1 WHERE C2 BETWEEN 12 AND 20`? You may assume most records have values of C2 outside of the range 12–20.
   (a) Hash index on C2
   (b) x Clustered B+-tree index on C2
   (c) Unclustered B+-tree index on C2
   (d) Sorted file on C2
   (e) Composite B+-tree index on C1, C2
6. What if all of the records in problem 5 had C2 values between 12 and 20?
   
   (a) Hash index on C2
   (b) Clustered B+-tree index on C2
   (c) Unclustered B+-tree index on C2
   (d) x Sorted file on C2
   (e) Composite B+-tree index on C1, C2

7. Consider the SQL query SELECT * FROM R1 WHERE C1=4 AND C2=10 AND C3=11. Suppose R1 has one million records stored in 100 disk pages, C1 has 10,000 unique values, C2 has 100,000 unique values, and C3 has 1,000 unique values, and values are distributed uniformly. Which relational algebra expression leads to the most efficient query execution plan?

   (a) \( \sigma_{C1=4}(\sigma_{C2=10}(\sigma_{C3=11}(R1))) \)
   (b) \( x \sigma_{C3=11}(\sigma_{C1=4}(\sigma_{C2=10}(R1))) \)
   (c) \( \sigma_{C2=10}(\sigma_{C1=4}(\sigma_{C3=11}(R1))) \)
   (d) \( \sigma_{C3=11}(\sigma_{C2=10}(\sigma_{C1=4}(R1))) \)
   (e) all four are equally efficient

8. Does your answer to number 7 change if there is a clustered B+-tree index on C3?

   (a) \( x \sigma_{C1=4}(\sigma_{C2=10}(\sigma_{C3=11}(R1))) \)
   (b) \( \sigma_{C3=11}(\sigma_{C1=4}(\sigma_{C2=10}(R1))) \)
   (c) \( \sigma_{C2=10}(\sigma_{C1=4}(\sigma_{C3=11}(R1))) \)
   (d) \( \sigma_{C3=11}(\sigma_{C2=10}(\sigma_{C1=4}(R1))) \)
   (e) all four are equally efficient

9. The efficiency of the nested-loop algorithm for computing natural or equi-joins can be improved by:

   (a) using an index on the table scanned by the outer loop to find needed records
   (b) using an index on the table scanned by the inner loop to find needed records
   (c) scanning the larger table in the outer loop
   (d) scanning the smaller table in the outer loop
   (e) a and c
   (f) x b and d

10. After translating a query into a relational expression, the expression can be optimized by

    (a) using equivalence rules to find alternative expressions that might cost less
    (b) using equivalence rules to make the expression tree more balanced
    (c) using equivalence rules to do the cheaper operations first when possible
    (d) x a and c
    (e) none of the above
11. Heuristics for query optimization include:
   (a) performing selection operations as early as possible
   (b) performing projections early
   (c) restricting multiple join expressions to simple forms such as left-deep trees
   (d) a and b
   (e) x all of the above

12. If you can build an index that supports index-only execution plans, is there any reason not to?
   (a) The queries for which index-only plans would be available are rare.
   (b) The index would include almost all fields in the table itself.
   (c) The expected cost of keeping the index updated outweighs the savings in query processing.
   (d) x All of the above.
   (e) None of the above—I would always support index-only execution plans if possible.

13. In SQL, users get privileges
   (a) only from the database administrator
   (b) from users with the same privileges and GRANT OPTION privilege
   (c) by means of a GRANT statement
   (d) a and c
   (e) x b and c

14. Which of the following is not true about granted privileges in SQL?
   (a) If user A grants a privilege to user B and user A subsequently loses that privilege, user B may
       still have that privilege.
   (b) x Privileges can be granted to users before the users are known to the DBMS.
   (c) A user can be granted a privilege without giving that user authorization to grant that privilege
       to others.
   (d) If privileges are revoked from one user, similar privileges may be automatically revoked from other
       users as well.

15. What does ACID stand for in the context of DBMS transactions?
   (a) x Atomicity, Consistency, Isolation, and Durability
   (b) Analysis Console for Intrusion Databases
   (c) Atomicity, Consistency, Isolation, and Data
   (d) Automatic Classification and Interpretation of Data
   (e) Advanced Computing Information Database
16. Enforcing serializability in concurrent schedules ensures which two of the four desired properties for transactions?
   (a) Atomicity and consistency
   (b) Atomicity and isolation
   (c) Atomicity and durability
   (d) Consistency and isolation
   (e) Consistency and durability
   (f) Isolation and durability

17. The recovery manager ensures which two of the four desired properties for transactions?
   (a) Atomicity and consistency
   (b) Atomicity and isolation
   (c) Atomicity and durability
   (d) Consistency and isolation
   (e) Consistency and durability
   (f) Isolation and durability

18. Suppose a database is read-only—no transactions change any data in the database. If serializability must be supported, which of the following is true?
   (a) No locking is necessary.
   (b) Only read locks are necessary and they need to be held until end of transaction.
   (c) Only read locks are necessary, but they can be released as soon as the read is complete.
   (d) Both read and write locks are necessary and locking must be done in two phases.
   (e) None of the above.

19. Checkpointing is a technique that can reduce recovery time after a crash. Which of the following is true?
   (a) When recovering, the log only needs to be scanned back to the most recent checkpoint.
   (b) Checkpointing is automatically performed after every transaction commit.
   (c) Checkpoints are saved after every update to the database.
   (d) a and c
   (e) a, b, and c

20. Which of the following is true about updated pages in the buffer pool?
   (a) Updated pages must be written immediately after the update.
   (b) Updated pages must be written after a transaction commits but before the transaction log is written to disk.
   (c) Updated pages must be written after a transaction commits but after the transaction log is written to disk.
   (d) An updated page must be written when it is swapped out of the buffer pool.
21. Which of the following is true of a distributed DBMS?
   (a) There is always one central server that distributes processing to other servers.
   (b) x Table columns may be stored on servers in different physical locations.
   (c) Data can always be redistributed for optimal query processing.
   (d) All of the above.

22. What is the purpose of classification algorithms?
   (a) Determine the most likely value of an unknown categorical field of a record.
   (b) Determine the most likely value of an unknown numerical field of a record.
   (c) Use data in which all field values are known to produce classification rules to be applied to new
       data.
   (d) x a and c
   (e) b and c
   (f) a, b, and c

23. Which property is guaranteed by the two-phase locking protocol?
   (a) serial schedules
   (b) x serializable schedules
   (c) recoverable schedules
   (d) avoiding cascading rollback

24. What information can an inverted index contain?
   (a) the presence of words in documents
   (b) the number of times words appear in documents
   (c) the relative positions at which words appear in documents
   (d) the number of documents a word appears in
   (e) some of the above
   (f) x all of the above

25. What kind of conflict (if any) is present in the transaction schedule below?
   \[ r_1(A); w_1(A); r_2(A); w_2(A); r_2(B); w_2(B); commit_2; r_1(B); w_1(B); commit_1 \]
   (a) read-write conflict
   (b) x write-read conflict
   (c) write-write conflict
   (d) no conflict

26. With regards to pages in the buffer pool, which pair of buffer management policies give the most
    efficient operation of the recovery manager?
   (a) stealing and forcing
   (b) x stealing and no forcing
   (c) no forcing and stealing
   (d) no forcing and no stealing
Short Answer (48 points+12 extra credit) Answer the following questions.

1. Query processing (8 points) Suppose we have the following relational schema:
   
   Students(sid:integer, name:string, gpa:real)
   Courses(course:string, credits:integer)
   Enrolled(sid:integer, course:string, semester:string)
   
   Also suppose there are hash indexes on Students.sid, Enrolled.sid, and Enrolled.course.
   
   Describe how you would compute each operation in the query execution tree below. For each operation, provide an algorithm name or write “on the fly”. Indicate whether the results of the operation will be materialized to disk or pipelined to the next operation. Be sure to indicate where indexes are used.

   Process $\sigma_{\text{course}= \text{CISC}437}$ using the hash index on Enrolled.course, pipeline to $\bowtie_{\text{sid} = \text{sid}}$, which uses index nested loop to get the matching records in Students (by using the hash index on Students.sid).
   
   Results are pipelined up to $\sigma_{\text{gpa}>3}$, which processes on-the-fly and pipelines to $\pi_{\text{name}}$, which processes on-the-fly and pipelines to output.
2. **Query optimization** (8 points) Suppose a user wants to obtain a list of students with GPAs under 3 that have taken 4-credit hour courses at any point in their academic career. The following relational algebra expressions are equivalent for the Students/Courses/Enrolled schema defined in problem 1. Which one is most likely to provide the most efficient query processing time? Justify your answer by explaining why none of the other expressions could be more efficient than the one you choose. (You do not have to compute total execution costs, though you may if you wish. Assume all joins are natural joins.)

- \( \pi_{\text{name, course, semester}}(\sigma_{\text{credits}=4 \land gpa<3}(\text{Students} \bowtie (\text{Enrolled} \bowtie \text{Courses}))) \)
- \( \pi_{\text{name, course, semester}}((\sigma_{\text{credits}=4}(\text{Courses}) \bowtie \text{Enrolled}) \bowtie \sigma_{\text{gpa}<3}(\text{Students})) \)
- \( \pi_{\text{name, course, semester}}(\sigma_{\text{gpa}<3}(\text{Students}) \bowtie (\sigma_{\text{credits}=4}(\text{Courses}) \bowtie \text{Enrolled})) \)

The second seems best: it’s left-deep and it pushes the selections down as far as possible. The first waits to perform selection on a very large set of records. The third is right-deep, which will require materializing \( \sigma_{\text{credits}=4}(\text{Courses}) \bowtie \text{Enrolled} \) and reading the result for every record or block in \( \sigma_{\text{gpa}<3}(\text{Students}) \).

3. **Recovery management** (8 points) Transactions \( T_1 \) and \( T_2 \) access data objects \( A, B, C, D, E \). Consider the following sequence of log records for UNDO logging:

<START T1>
<T1,A,10>
<START T2>
<T2,B,20>
<T1,C,30>
<T2,D,40>
<COMMIT T2>
<T1,E,50>
<COMMIT T1>

Suppose the last record that appears on disk at the time of a crash is <COMMIT T2>. What will the recovery manager do to recover from this crash in terms of updates to the disk and to the log?

It will go backwards from <COMMIT T2>. Since \( T_2 \) committed, nothing for \( T_2 \) needs to be undone. Recovery manager will skip <T2,D,40>. At <T1,C,30> it will realize that \( T_1 \) never committed, so it will undo that by setting \( C \) to 30 on disk. It will skip <T2,B,20> and <START T2>, and then undo <T1,A,10> by setting \( A \) to 10 on disk. Done.
4. **Serializability** (8 points) Suppose the transaction manager produces the follow schedule for transactions $T_1, T_2, T_3$ access data objects $A, B, C$:

$$r_1(A); r_2(A); r_3(B); w_1(A); r_2(C); r_2(B); w_2(B); w_2(C)$$

($r_i(O)$ indicates a read by transaction $T_i$ on data object $O$; $w_i(O)$ indicates a write by transaction $T_i$ on data object $O$.)

(a) Identify and list any conflicts between transactions.

- read-write conflicts: $r_2(A)$ and $w_1(A)$, $r_3(B)$ and $w_2(B)$
- write-read conflicts: none
- write-write conflicts: none

(b) Is this schedule serializable? If so, give the equivalent serial schedule. If not, explain why not.

Yes. $r_1(A); w_1(A); r_2(A); r_2(C); r_2(B); w_2(B); w_2(C); r_3(B)$ is serialized.

5. **Concurrency control** (8 points) Consider the following transactions:

$$T_1 : r_1(X); w_1(Y);$$
$$T_2 : r_2(Y); w_2(X);$$

For each of the following schedules, indicate whether it can be generated by 2PL, strict 2PL, both, or neither by circling your answer from the choices below.

x marks the answer.

- $\ell_1(X); r_1(X); \ell_1(Y); u_1(X); \ell_2(X); w_1(Y); u_1(Y); \ell_2(Y); r_2(Y); w_2(X); u_2(Y); u_2(X)$
  - 2PL / S-2PL / both / neither
- $\ell_2(Y); r_2(Y); u_2(Y); \ell_2(X); w_2(X); u_2(X); \ell_1(X); r_1(X); \ell_1(Y); w_1(Y); u_2(X); w_2(Y)$
  - 2PL / S-2PL / both / x neither
- $\ell_1(X); \ell_1(Y); r_1(X); w_1(Y); u_1(X); u_1(Y); \ell_2(Y); r_2(Y); w_2(X); u_2(Y); u_2(X)$
  - 2PL / S-2PL / x both / neither
6. **Parallel query processing** (8 points) Consider the following tables:

<table>
<thead>
<tr>
<th>ssn</th>
<th>name</th>
<th>firmName</th>
<th>firmLoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>111-11-1111</td>
<td>Bob Loblaw</td>
<td>Dewey, Cheatham, and Howe</td>
<td>Boston</td>
</tr>
<tr>
<td>222-22-2222</td>
<td>Ally McBeal</td>
<td>Payne and Feares</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>222-22-3333</td>
<td>Maury Levy</td>
<td>Baker and Launder</td>
<td>Baltimore</td>
</tr>
<tr>
<td>333-44-5555</td>
<td>Saul Goodman</td>
<td>Recht and Greef</td>
<td>Albuquerque</td>
</tr>
<tr>
<td>555-55-6666</td>
<td>Atticus Finch</td>
<td>Baker and Launder</td>
<td>Baltimore</td>
</tr>
</tbody>
</table>

(a) Lawyers(ssn:string, name:string, firmName:string, firmLoc:string)

<table>
<thead>
<tr>
<th>firmName</th>
<th>firmLoc</th>
<th>employees</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dewey, Cheatham, and Howe</td>
<td>Boston</td>
<td>72</td>
</tr>
<tr>
<td>Dewey, Cheatham, and Howe</td>
<td>San Francisco</td>
<td>95</td>
</tr>
<tr>
<td>Payne and Feares</td>
<td>Los Angeles</td>
<td>55</td>
</tr>
<tr>
<td>Recht and Greef</td>
<td>Albuquerque</td>
<td>120</td>
</tr>
<tr>
<td>Pope and Gentile</td>
<td>Milwaukee</td>
<td>100</td>
</tr>
<tr>
<td>Boring and Leach</td>
<td>Los Angeles</td>
<td>66</td>
</tr>
</tbody>
</table>

(b) Firms(firmName:string, firmLoc:string, employees:integer)

Suppose there is a parallel database system operating on two loosely-coupled (share-nothing) processors. Processor $P_1$’s disk stores rows 1–3 of Lawyers and rows 4–6 of Firms. Processor $P_2$’s disk stores rows 4–5 of Lawyers and row 1–3 of Firms. Describe the steps that need to be taken to compute a parallel join of the two tables on firmName and firmLoc. Be sure your description allows both processors to be working simultaneously (even though in this particular case it might be more efficient for one processor to do all the work).

We just need to transfer some data on disk 2 to disk 1 and vice versa so that both disks will have the records that match. We can do that with a hash function on (firmName, firmLoc). Once data has been moved, both processors use some join algorithm (doesn’t really matter which one) to join their respective parts of the data, and then the final result is collected at one of the processors for final display.
I didn’t cover these topics this year.

7. **XML** (6 points; extra credit) Write out a well-formed, valid XML file that uses the DTD below. Include at least 10 elements and at least one attribute.

```xml
<!ELEMENT SONG (TITLE, COMPOSER*, PRODUCER*, PUBLISHER*, LENGTH?, YEAR?, ARTIST*)>
<!ELEMENT TITLE (#PCDATA)>
<!ELEMENT COMPOSER (#PCDATA)>
<!ELEMENT PRODUCER (#PCDATA)>
<!ELEMENT PUBLISHER (#PCDATA)>
<!ELEMENT YEAR (#PCDATA)>
<!ELEMENT ARTIST (#PCDATA)>
<!ATTLIST SONG LENGTH CDATA #IMPLIED>
```

8. **Association rules** (6 points; extra credit) The table below provides lists of items that a customer purchased together.

<table>
<thead>
<tr>
<th>Sale</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>Bread, Jelly, PeanutButter</td>
</tr>
<tr>
<td>$t_2$</td>
<td>Bread, PeanutButter</td>
</tr>
<tr>
<td>$t_3$</td>
<td>Bread, Milk, PeanutButter</td>
</tr>
<tr>
<td>$t_4$</td>
<td>Beer, Bread</td>
</tr>
<tr>
<td>$t_5$</td>
<td>Beer, Milk</td>
</tr>
</tbody>
</table>

- Provide two association rules with support greater than or equal to 60%. (Hint: $X \rightarrow Y$ and $Y \rightarrow X$ are two different rules.)

- Provide four association rules with confidence greater than or equal to 50%.