Mid-Term Exam

⊕ **Avoid asking questions** - If something is unclear, document your assumptions and move on.

⊕ **Do not forget to write your name on the first page. Initial each subsequent page.**

⊕ **Be neat and precise. We will not grade answers we cannot read.**

⊕ **We will only look at the front page.** Use the backs for scratch paper, but will not look at any text except on the front.

⊕ **If you have written something incorrect along with the correct answer, you should not expect to get all the points. I will grade based upon what you wrote, not what you meant.**

⊕ **You should draw simple figures if you think it will make your answers clearer.**

⊕ **Good luck and remember, brevity is the soul of wit.**

<table>
<thead>
<tr>
<th>Num</th>
<th>Poss</th>
<th>Pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>
1. (10 pts) True/False:

   (a) Deriving $A \rightarrow C$ from $A \rightarrow B$ and $B \rightarrow C$ is an example of pseudo transitivity.
   \textbf{answer:} false

   (b) It is always possible to get a dependency-preserving decomposition into BCNF.
   \textbf{answer:} false

   (c) The keyword \texttt{where} in SQL maps to the $\pi$ operation in Relational Algebra.
   \textbf{answer:} false

   (d) 3NF is guaranteed to preserve dependencies.
   \textbf{answer:} true

   (e) FALSE OR UNKNOWN = UNKNOWN.
   \textbf{answer:} true

   (f) The entire set of attributes of a relation is always a candidate key for the relation.
   \textbf{answer:} false

   (g) Deriving $AB \rightarrow CB$ from $A \rightarrow C$ is an example of augmentation.
   \textbf{answer:} true

   (h) 4NF improves on 3NF by guaranteeing lossless decomposition.
   \textbf{answer:} false

   (i) A relation schema that is in 3NF is also in BCNF.
   \textbf{answer:} false

   (j) A multi-valued attribute has several parts and is often split into several attributes, one for each part.
   \textbf{answer:} false
2. (20 pts) Short Answer

(a) Compute the answer for:

```
SELECT A, SUM(C) FROM R GROUP BY A, B;
```

**answer:** \((\alpha, 4), (\alpha, 3), (\gamma, 3), (\beta, 4), (\gamma, 2), (\beta, 2)\)

(b) Compute the answer for:

```
SELECT A FROM R r1 WHERE EXISTS
(SELECT * FROM R r2 WHERE r2.A = r1.A AND r2.C < r1.C);
```

**answer:** \((\alpha), (\gamma), (\alpha)\)
(c) Give the result of the following query: $\sigma_{B \cdot D < 40}(S \bowtie T)$

\textbf{answer:} (13, biff, 3, 1)

(d) Give the result of the following query: $\pi_E(R \bowtie T)$

\textbf{answer:} (1)
3. (15 pts) E-R Diagrams and relational schema

(a) (7 pts) Convert the above E-R Diagram to relational schema. Do not eliminate or combine any relations in this schema.

answer: 
person(SSN, pname), bikes(bname, type), wheels(where, wname),
owns(SSN, bname), on(bname, where)

(b) (8 pts) Now reduce the schema.

answer: 
Git rid of owns by adding SSN to bikes. 
Get rid of on by adding bname to where.

person(SSN, pname), bikes(bname, SSN, type), wheels(bname, where, wname)
4. BobFriends (15 pts)

(a) Write a query to return all unique friends of bob. Your result should be a column called Friend, ordered by Friend.

   answer:
   SELECT * FROM ((SELECT A FROM friends WHERE B='bob') UNION (SELECT B FROM friends WHERE A='bob')) t
   ORDER BY A;

(b) Write the previous query in relational algebra. We do not care about the result name.

   answer: \( \pi_B(\sigma_A=bob(friends)) \cup \pi_A(\sigma_B=bob(friends)) \)

(c) Write a query returning for each student a count of their unique friends, together with the student’s name. Result columns will be Student and Num, order by Student.

   answer: 
   WITH t AS ((SELECT A as s, B as f FROM friends) UNION (SELECT B as s, A as f FROM friends))
   SELECT s as Student, COUNT(num) AS num FROM t GROUP BY s ORDER BY s;
5. (30 pts) Given the relation schema: \( R(A, B, C, D, E, F, G) \), and FDs on it:

\[
\begin{align*}
A & \rightarrow C \\
BD & \rightarrow EF \\
D & \rightarrow G \\
AG & \rightarrow D.
\end{align*}
\]

(a) List all candidate keys.

\textbf{answer:} AB must be there, so ABD, ABG

(b) Is the relation in BCNF? List one FD that violates it if it is not.

\textbf{answer:} They all violate BCNF.

(c) Decompose the relation into BCNF if it is not already in it.

\textbf{answer:} The resulting subrelations should be in BCNF, though won’t preserve depend.
(d) Is your decomposition into BCNF dependency-preserving? List one FD that is not carried if it is not.

**answer:** No. Prob either \( D \rightarrow G \) or \( AG \rightarrow D \).

(e) Is the relation \( (R) \) in 3NF? List one FD that violates it if it is not.

**answer:** No, both (and ONLY THESE) \( A \rightarrow C \) and \( BD \rightarrow EF \) violate.

(f) Are there any extraneous attributes in either \( BD \rightarrow EF \) or \( AG \rightarrow D \)? If so, list them.

**answer:** No:
- For G in \( AG \rightarrow D \), \((A)^+ = AC\), so no
- For D in \( BD \rightarrow EF \), \((B)^+ = b\), no
- For B in \( BD \rightarrow EF \), \((D)^+ = DG\), no
- E in \( BD \rightarrow EF \), start w/ \( BD \rightarrow F \), prove \( BD \rightarrow E \). No E on right of any FD.
- F in \( BD \rightarrow EF \), start w/ \( BD \rightarrow E \), prove \( BD \rightarrow F \). No F on right of any FD.
6. (10 pts) Use only Armstrongs Axioms to prove the soundness of the Union rule, i.e:

Given:

\[ A \rightarrow B \]  \hspace{1cm} (1)
\[ A \rightarrow C \]  \hspace{1cm} (2)

Prove:

\[ A \rightarrow BC \]

answer:

\[ A \rightarrow AB \text{ (aug of 1, set semantics)} \]
\[ AB \rightarrow BC \text{ (aug of 2)} \]
\[ A \rightarrow AB \rightarrow BC \text{ (trans)} \]