1. (20 points)
   Consider the following schemas:

   **Schema 1:** Note that ASSIGN is many-one.

   **Schema 2:** Note that ASSIGN is many-many.

   **Schema 3:** Note that ASSIGN is ternary, many-many-one (to TEACHER).

   In each of the above schemas, the keys of the entity sets are underlined. So the key of PUPIL in all schemas is \{P\#\}, the key of TEACHER in Schema 1 is \{Name, Subject\}, and the key of TEACHER in Schemas 2 and 3 is \{Name\}. Schema 3 uses an entity set for the attribute Subject of Schemas 1 and 2.
For each of the following descriptions of (partial) instances, state which schema will allow it by writing yes on the appropriate line. If it is not allowed by the schema, briefly explain why.

Note that in the following descriptions the first field in a pupil record is P# and the second field is Name; the first field in a teacher record is Name and the second field is Room#. Also, note that since the schemas differ on how they represent the association between teacher and subject, teacher tuples will have “...” and the subject will be specified separately as needed.

(a) (1pt) (1,Mary) is a pupil; (1,John) is a pupil.
Answer: 
None, since P# is a key.

(b) (1pt) (10, Susan) is a pupil; (12, Susan) is a pupil.
Answer: 
All

(c) (3pt) Pupil (1,Mary) is assigned to teacher (Smith,412...) for Math and to teacher (DiBernardi,320...) for Science.
Answer: 
(b), (c) since in (a) a pupil can only be assigned to one teacher.

(d) (3pt) (3,Mitch) is a pupil, but is not assigned to any teacher for any subject.
Answer: 
All. Because pupil is not a weak entity.

(e) (3pt) No pupils are assigned to teacher (Ambrose,191...) for Social Studies; however, Ambrose is known to teach Social Studies in room 191.
Answer: 
(a). Not (b) or (c), because relationship can not exist without the existence of entities it associated with.
(f)  (3pt) Pupil (4, Mark) is assigned to teacher (DiBernardi, 320...) for Science and to teacher (DiBernardi, 429...) for English.
   Answer:
   None. Not (a) because ASSIGN is many-one, and not (b)(c) because Name is a key for teacher.

(g)  (3pt) Pupil (5, Mike) is assigned to teacher (Dunn, 320...) for Science and to teacher (Dunn, 320...) for English.
   Answer:
   (c). Not (a) because ASSIGN is many-one, and not (b) because there is a single subject for each pupil-teacher relationship.

(h)  (3pt) Pupil (5, Mike) is assigned to teacher (Dunn, 119...) for Science and to teacher (Rounds, 543...) for Science.
   Answer:
   (b). Not (a) because ASSIGN is many-one, and not (c) because for each pupil, subject pair there can be at most one teacher.
2. (10 points)
Consider the ER diagrams in the first question.

(a) Which schemas (if any) express the constraint that a pupil must be assigned to exactly one instance of each subject?

*Answer: None*

(b) Consider the following relational version of Schema 3 (keys indicated by bold font):

\[
\text{Pupil}(P\#, PName) \\
\text{Teacher}(TName, Room\#) \\
\text{Subject}(Sname) \\
\text{Assign}(P\#, Sname, TName)
\]

i. Using the relational algebra (\(\Pi, \sigma, \cup, -, \times, \bowtie, \div, \delta\)), write the following query: “Print the \(P\#\) of all pupils who are assigned to every subject.”

*Answer:

\[
STemp = (\Pi_{P\#, Sname, Assign}(Assign) \bowtie Subject)
\]

ii. Extend your answer above to answer the following query: “Print the \(P\#\) of all students who have exactly one teacher for every subject.”

*Answer:

Next, we throw out \(P\#\) from \(STemp\) who are assigned to more than one teacher for some subject:

\[
TAssign = STemp \bowtie_{P\#} Assign \\
BadP\# = \Pi_{P\#} \sigma_{TName < T2Name}(\delta_{TName=TName TAssign}) \\
((\delta_{TName=TName TAssign}) \bowtie_{P\#, Sname}(\delta_{TName=T2Name TAssign})) \\
Result = STemp - BadP\#
\]
3. (15 points)
Consider a database of rides and their rating (an integer from 1 to 10, with 1 being easy and 10 being hard), children and the maximum rating that they are allowed to ride, and the rides that each child likes.

RIDES(RIDE, RATING)
CHILDREN(CHILD, MAX)
LIKES(CHILD, RIDE)

Write SQL expressions for each of the following queries.

(a) Print the number of rides for each rating.
   \(Answer:\)
   ```sql
   select rating, num=count(*)
   from rides
   group by rating
   ```

(b) Print all pairs (RIDE, CHILD) where the child is allowed on the ride and the child likes the ride.
   \(Answer:\)
   ```sql
   select r.ride, c.child
   from rides r, children c, likes l
   where r.rating \geq c.max and l.ride = r.ride and c.child = l.child
   ```

(c) Print the children who like some ride that they are not allowed to go on.
   \(Answer:\)
   ```sql
   select l.child
   from likes l, rides r, child c
   where l.ride = r.ride and c.child = l.child and c.max \leq r.rating
   ```

(d) Print the names of children who like no ride that they are allowed to go on.
   \(Answer:\)
   Note that they could also use their answer from b above.
   ```sql
   select c.child
   from children c
   where not exists
   (select *
    from children c, rides r, likes l
    where c.child = l.child
    and c.max \geq r.rating and l.ride = r.ride)
   ```
4. (25 points)

(a) Consider the following relation:

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_1</td>
<td>y_1</td>
<td>z_1</td>
</tr>
<tr>
<td>x_1</td>
<td>y_2</td>
<td>z_1</td>
</tr>
<tr>
<td>x_1</td>
<td>y_1</td>
<td>z_2</td>
</tr>
<tr>
<td>x_1</td>
<td>y_2</td>
<td>z_3</td>
</tr>
</tbody>
</table>

In this instance of the relation XYZ, list any non-trivial functional dependencies that hold. Does this mean that the functional dependencies hold for the relation scheme XYZ? (4pt) Why or why not? (3pt)

Answer:
Z \rightarrow X, Y \rightarrow X.

This does not mean that the functional dependencies hold for the relation schema; functional dependencies are a constraint that is maintained by the DBMS, a semantic assertion.

(b) Consider a relation $R$ with five attributes $ABCDE$. You are given the following dependencies: $A \rightarrow B, BC \rightarrow E, ED \rightarrow A$.

i. List all keys for R. (6pt)

Answer:
{$EDC, BCD, ACD$}. Note that D and C must be in any key since they do not appear on the right-hand side of any dependency.

ii. Which dependencies in $R$ violate 3NF? (4pt)

Answer:
None. Because B, E, A each is part of some key for R.

iii. Which dependencies in $R$ violate BCNF? (4pt)

Answer:
All of the dependencies violate BCNF. Because none of those function dependencies are trivial, and none of A, BC, or ED is a superkey.

iv. Give a lossless join, dependency preserving decomposition into 3NF. (4pt)

Answer:
The relation as given is fine. You also could decompose into AB, BCE, EDA, and one of the keysEDC, BDC, ADC, according to the algorithm in the lecture "Relational Design Theory"