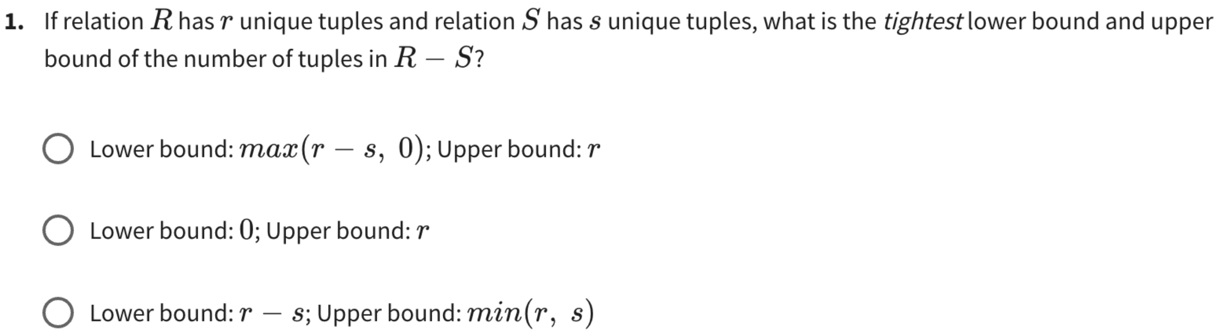
Week 4 Computing on Data

Assignment Solutions



1. If relation R has r unique tuples and relation S has s unique tuples, what is the *tightest* lower bound and upper bound of the number of tuples in $R - S$?

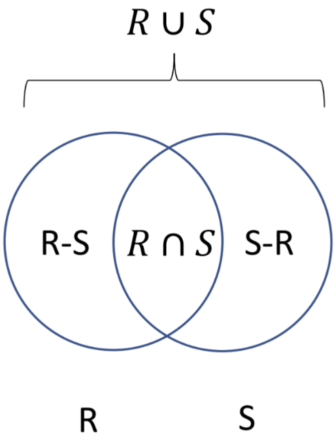
○ Lower bound: $max(r - s,\ 0)$; Upper bound: r

○ Lower bound: 0; Upper bound: r

○ Lower bound: $r - s$; Upper bound: $min(r,\ s)$

Answer: A

Explanation: The key is that we need to make use of the knowledge regarding their sizes: |R| = r, |S| = s, as the constraint to obtain the tightest lower bound and upper bound. This type of questions can be visualized in the figure on the right side.



Upper bound:

Upper bound achieves when R and S has no intersection. The cardinality of R-S is thus r.

Lower bound:

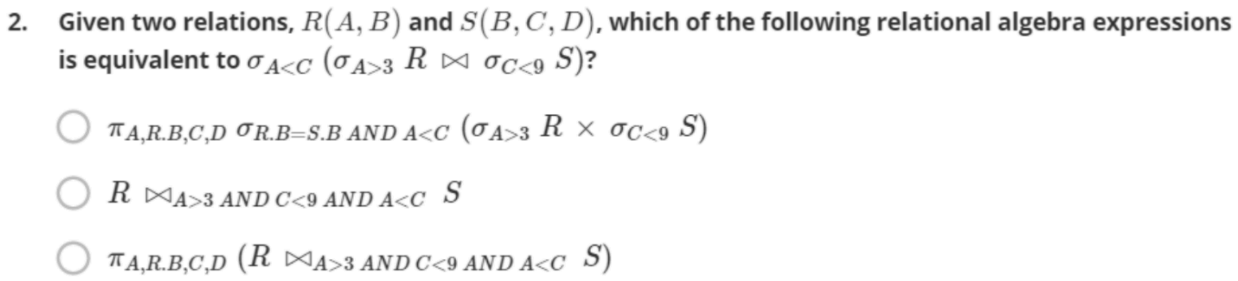
There are two exhaustive and mutually exclusive cases we can take regarding r and s:

Case 1. r ≥ s

In this case, it is clear that the tightest lower bound of |R−S| is r−s, because at most S can only take away s number of elements from R (when S is a subset of R), and no more!

Case 2. r < s

In this case, we see that the tightest lower bound of $|R-S|$ is 0, because now S is big enough to take away everything from R (when S is a proper superset of R), leaving an empty set. Now combining the two exhaustive cases above, we see that, regardless of how r and s compare, we have a general lower bound $\max(r-s, 0)$ that holds for any given r and s, and it is the tightest bound we can get, in particular tighter than the simple zero bound.



2. **Given two relations, $R(A, B)$ and $S(B, C, D)$, which of the following relational algebra expressions is equivalent to $\sigma_{A<C}\ (\sigma_{A>3}\ R \bowtie \sigma_{C<9}\ S)$?**

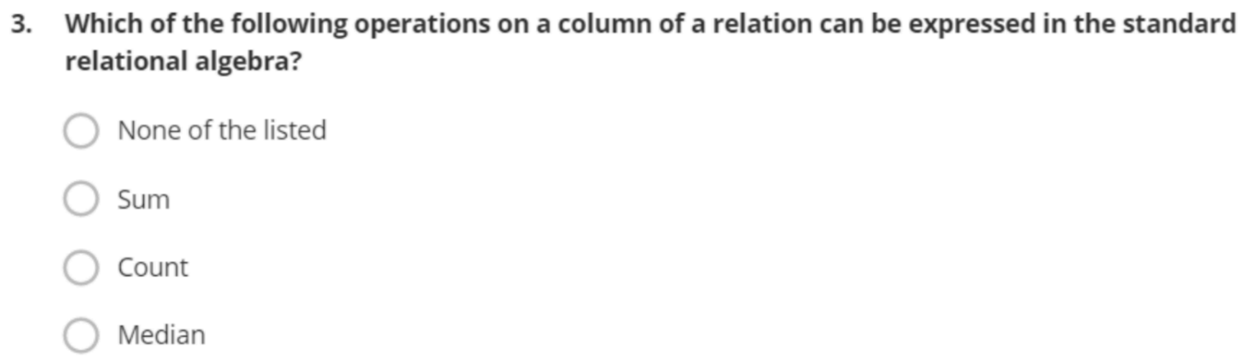
○ $\pi_{A,R.B,C,D}\ \sigma_{R.B=S.B\ AND\ A<C}\ (\sigma_{A>3}\ R \times \sigma_{C<9}\ S)$

○ $R \bowtie_{A>3\ AND\ C<9\ AND\ A<C}\ S$

○ $\pi_{A,R.B,C,D}\ (R \bowtie_{A>3\ AND\ C<9\ AND\ A<C}\ S)$

Answer: A

Explanation: A natural join is equivalent to a Cartesian product, followed by a selection to filter out tuples with non-matching values on common attributes, and then followed by a projection to merge common attributes. Option B misses selection and projection. Option C miss selection.



3. Which of the following operations on a column of a relation can be expressed in the standard relational algebra?

○ None of the listed

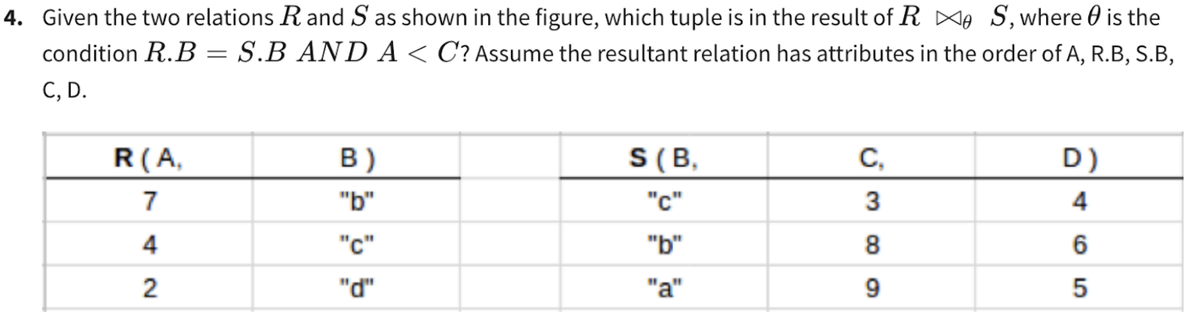
○ Sum

○ Count

○ Median

Answer: A

Explanation: The aggregate functions, such as count, sum, median, cannot be expressed in standard relational algebra (you can think that relational algebra has no memory to accumulate information as it goes through the tuples of a relation). But these useful aggregations are implemented in most relational database systems as extensions to standard RA.



4. Given the two relations R and S as shown in the figure, which tuple is in the result of $R \bowtie_{\theta} S$, where θ is the condition $R.B = S.B\ AND\ A < C$? Assume the resultant relation has attributes in the order of A, R.B, S.B, C, D.

R (A,	B)		**S** (B,	C,	D)
7	"b"		"c"	3	4
4	"c"		"b"	8	6
2	"d"		"a"	9	5

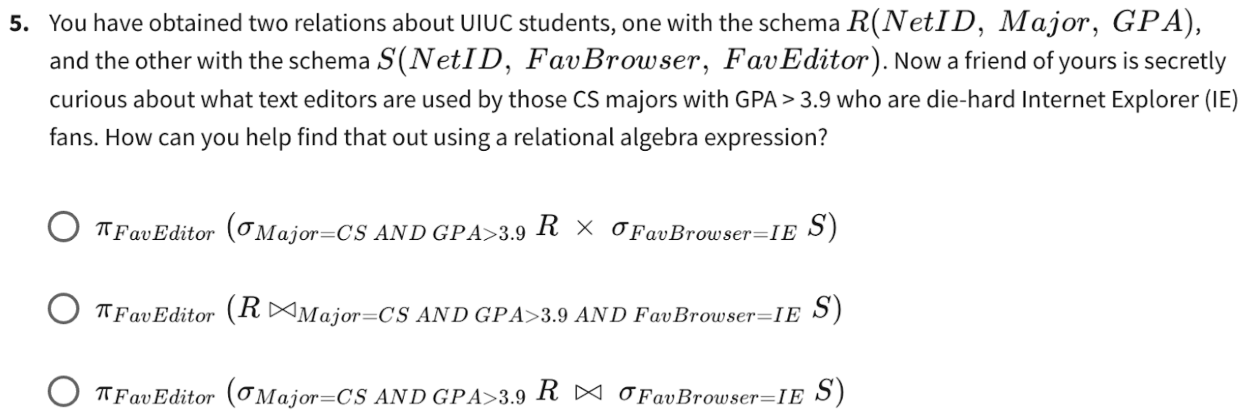
A: (7, “b", “b", 8, 6)

B: (4, “c", “c", 8, 6)

C: (4, “c", “c", 3, 4)

Answer: A

Explanation: A is the only tuple in joint result. Option B is not in Cartesian product of R and S. Option C violates A < C.



5. You have obtained two relations about UIUC students, one with the schema $R(NetID,\ Major,\ GPA)$, and the other with the schema $S(NetID,\ FavBrowser,\ FavEditor)$. Now a friend of yours is secretly curious about what text editors are used by those CS majors with GPA > 3.9 who are die-hard Internet Explorer (IE) fans. How can you help find that out using a relational algebra expression?

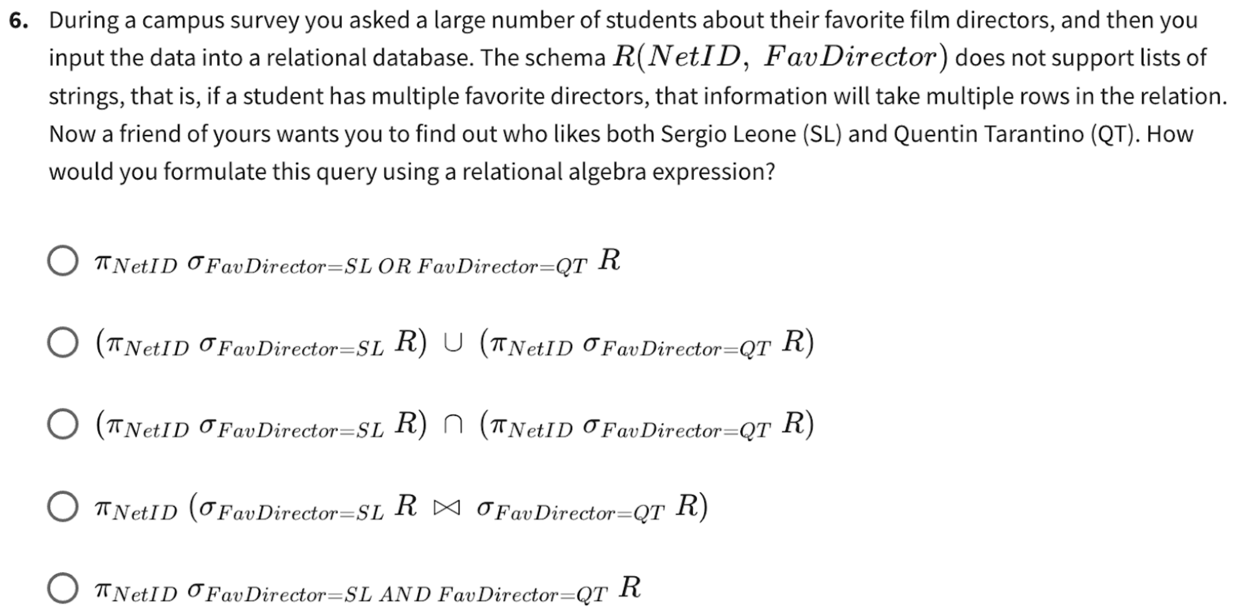
○ $\pi_{FavEditor}\ (\sigma_{Major=CS\ AND\ GPA>3.9}\ R\ \times\ \sigma_{FavBrowser=IE}\ S)$

○ $\pi_{FavEditor}\ (R \bowtie_{Major=CS\ AND\ GPA>3.9\ AND\ FavBrowser=IE}\ S)$

○ $\pi_{FavEditor}\ (\sigma_{Major=CS\ AND\ GPA>3.9}\ R\ \bowtie\ \sigma_{FavBrowser=IE}\ S)$

Answer: C

Explanation: Option A and B both miss R.NetID = S.NetID, since Cartesian product and theta join do not match common attributes. Natural join in option C automatically performs the selection.



6. During a campus survey you asked a large number of students about their favorite film directors, and then you input the data into a relational database. The schema $R(NetID,\ FavDirector)$ does not support lists of strings, that is, if a student has multiple favorite directors, that information will take multiple rows in the relation. Now a friend of yours wants you to find out who likes both Sergio Leone (SL) and Quentin Tarantino (QT). How would you formulate this query using a relational algebra expression?

○ $\pi_{NetID}\ \sigma_{FavDirector=SL\ OR\ FavDirector=QT}\ R$

○ $(\pi_{NetID}\ \sigma_{FavDirector=SL}\ R)\ \cup\ (\pi_{NetID}\ \sigma_{FavDirector=QT}\ R)$

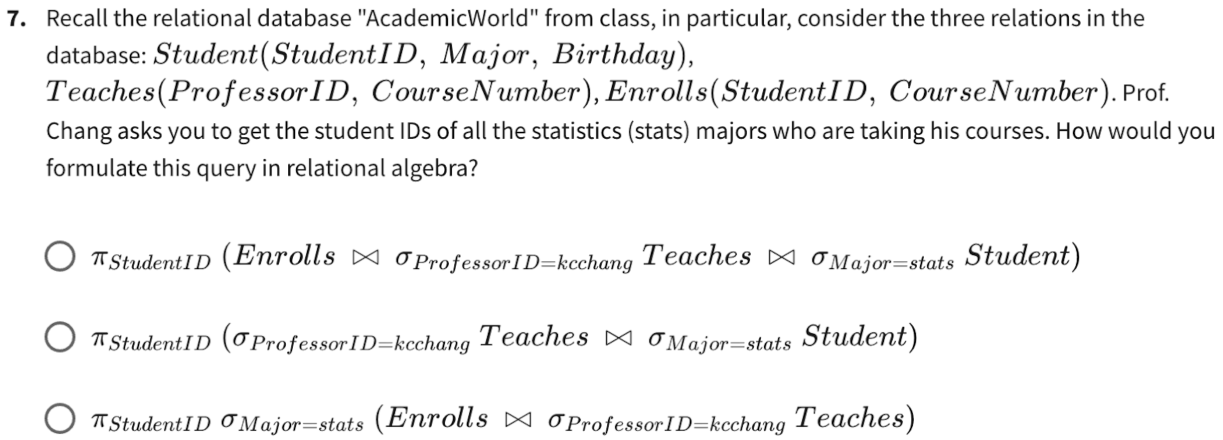
○ $(\pi_{NetID}\ \sigma_{FavDirector=SL}\ R)\ \cap\ (\pi_{NetID}\ \sigma_{FavDirector=QT}\ R)$

○ $\pi_{NetID}\ (\sigma_{FavDirector=SL}\ R\ \bowtie\ \sigma_{FavDirector=QT}\ R)$

○ $\pi_{NetID}\ \sigma_{FavDirector=SL\ AND\ FavDirector=QT}\ R$

Answer: C

Explanation: We want to find those who like SL, and those who like QT, via two selections on the same relation, and then take the intersection to get the set of those who like both.



7. Recall the relational database "AcademicWorld" from class, in particular, consider the three relations in the database: $Student(StudentID,\ Major,\ Birthday)$, $Teaches(ProfessorID,\ CourseNumber), Enrolls(StudentID,\ CourseNumber)$. Prof. Chang asks you to get the student IDs of all the statistics (stats) majors who are taking his courses. How would you formulate this query in relational algebra?

○ $\pi_{StudentID}\ (Enrolls\ \bowtie\ \sigma_{ProfessorID=kcchang}\ Teaches\ \bowtie\ \sigma_{Major=stats}\ Student)$

○ $\pi_{StudentID}\ (\sigma_{ProfessorID=kcchang}\ Teaches\ \bowtie\ \sigma_{Major=stats}\ Student)$

○ $\pi_{StudentID}\ \sigma_{Major=stats}\ (Enrolls\ \bowtie\ \sigma_{ProfessorID=kcchang}\ Teaches)$

Answer: A

Explanation: The first natural join connects student enrollment to professor teaching via the common attribute CourseNumber; the second natural join enables major filtering on the previous join result. Option B misses the relation between Student and Course. Option C doesn't have the Major attribute.

8. Recall the Food for Thought question from class: how can NoSQL database systems based on the document model largely do away with join? Check all the features that contribute to this advantage over relational database systems.

☐ Sub-document embedding

☐ Document referencing

☐ Arrays

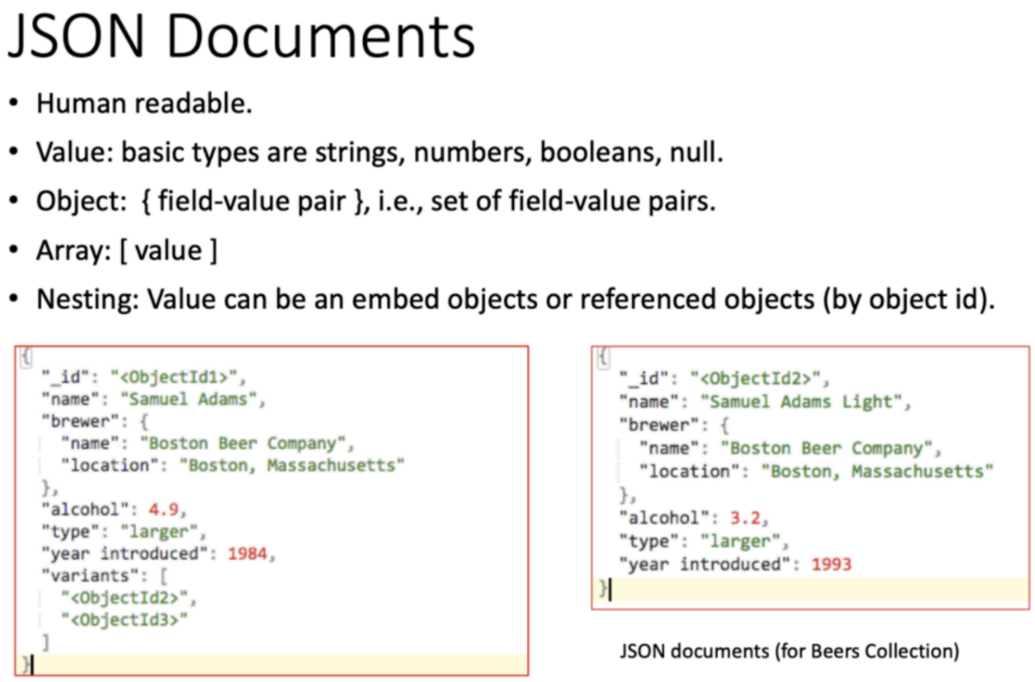
Answer: ABC

Explanation: Arrays support one-to-many relationships.

Document referencing references to other document objects which reduces data duplication in highly hierarchical documents and in representing many-to-many relationships. However, this is at the cost of incurring extra queries to follow the references (possibly to remote machines).

Sub-document embedding "Pre-joins" relevant data right into a single document thanks to the nesting capability of documents.

The three options above can be related to lecture notes on Json documents.



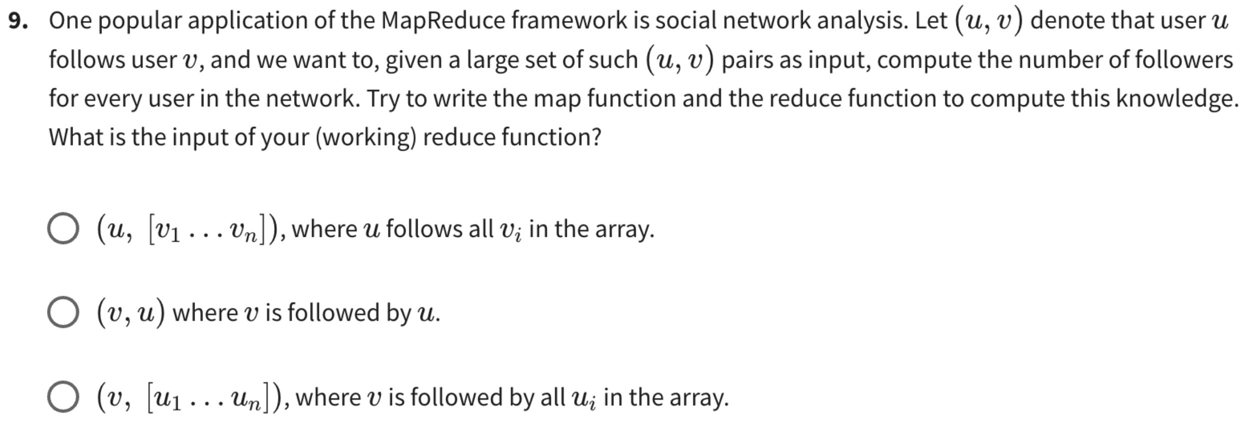
JSON Documents

- Human readable.
- Value: basic types are strings, numbers, booleans, null.
- Object: { field-value pair }, i.e., set of field-value pairs.
- Array: [value]
- Nesting: Value can be an embed objects or referenced objects (by object id).

```
{
  "_id": "<ObjectId1>",
  "name": "Samuel Adams",
  "brewer": {
    "name": "Boston Beer Company",
    "location": "Boston, Massachusetts"
  },
  "alcohol": 4.9,
  "type": "larger",
  "year introduced": 1984,
  "variants": [
    "<ObjectId2>",
    "<ObjectId3>"
  ]
}
```

```
{
  "_id": "<ObjectId2>",
  "name": "Samuel Adams Light",
  "brewer": {
    "name": "Boston Beer Company",
    "location": "Boston, Massachusetts"
  },
  "alcohol": 3.2,
  "type": "larger",
  "year introduced": 1993
}
```

JSON documents (for Beers Collection)



9. One popular application of the MapReduce framework is social network analysis. Let (u, v) denote that user u follows user v, and we want to, given a large set of such (u, v) pairs as input, compute the number of followers for every user in the network. Try to write the map function and the reduce function to compute this knowledge. What is the input of your (working) reduce function?

○ $(u,\ [v_1 \ldots v_n])$, where u follows all v_i in the array.

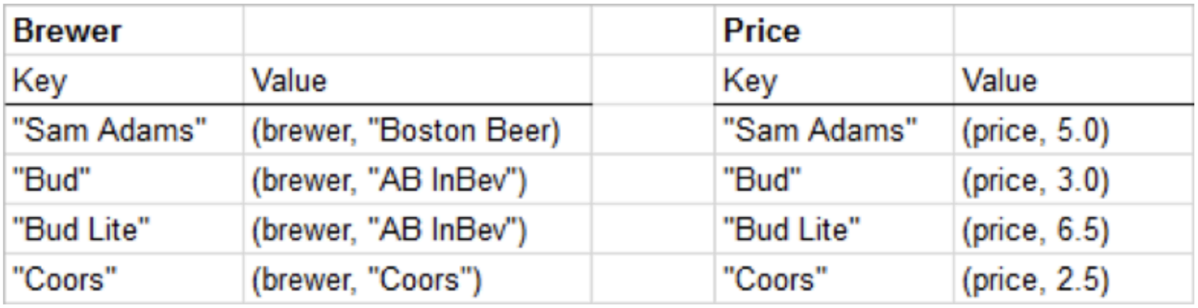
○ (v, u) where v is followed by u.

○ $(v,\ [u_1 \ldots u_n])$, where v is followed by all u_i in the array.

Answer: C

Explanation: This question asks to compute the number of followers for every user. Thus, the key of Group function is the user being followed, called followee. The Map function should reverse the input key-value pair and output (followee, follower) pairs, such as M(u,v) -> (v,u). The Group function will group these pairs according to the followee v key, before assigning each such group to one Reduce task, which then computes the length of the follower array as the desired output.

10. Recall the Food for Thought question from class: can we implement theta join under the MapReduce framework to achieve distributed computing? In particular, you were asked to compute $Brewer \bowtie_{brewer=ABInBev\ AND\ price>5.0} Price$ by MapReduce (and project the join result to the beer name). Using the tables below, try to work out the map function and the reduce function. What is the input of your (working) reduce function?



Brewer			Price	
Key	Value		Key	Value
"Sam Adams"	(brewer, "Boston Beer)		"Sam Adams"	(price, 5.0)
"Bud"	(brewer, "AB InBev")		"Bud"	(price, 3.0)
"Bud Lite"	(brewer, "AB InBev")		"Bud Lite"	(price, 6.5)
"Coors"	(brewer, "Coors")		"Coors"	(price, 2.5)

○ The input is ***(beer_name, values)***, where ***values*** is a list of length at most 2.

○ The input is ***(beer_name, values)***, where ***values*** is a list of length at least 2.

○ The input is ***(beer_name, values)***, where ***values*** is a list of length exactly 2.

Answer: A

Explanation: The Map function should emit a beer name (as the key of a key-value pair) if its brewer is ABInDev, OR its price is greater than 5.0. For example, the outputs of Map function of Brewer table are (“Bud”, “AB InBev”) and (“Bud Lite”, “AB InBev”) because they satisfy the condition of brewer = ABInDev. Likewise, the output of Map function of Price table is (“Bud Lite”, 6.5) since it’s the only tuple satisfying the condition of price > 5.0., The input of the Reduce function can therefore have at most two entries in the values list. Having exactly two entries in the list means that the beer satisfies both criteria and thus should be emitted as one of the join results.