Quiz 3: DSC 208 Data Management for Analytics

Questions and Explanations

Question 1. Which of the following kinds of predicates in a selection query is amenable to being sped up using both a hash index and B+ tree index?

a) Equal to

b) Less than

c) Not equal to

d) Greater than or equal to

Answer: a) Equal to

- *Explanation:*
  - **Hash Index:** Hash indexes are highly efficient for "equal to" ('=') predicates because they directly map a key value to its storage location using a hash function. This provides very fast lookups for exact matches.
  - **B+ Tree Index:** B+ trees are also excellent for "equal to" predicates. They provide logarithmic time complexity for searching, and once the key is found, they can quickly retrieve the corresponding data.
  - **Range Queries (Less than, Greater than or equal to):** B+ trees are uniquely suited for range queries because their leaf nodes are linked in a sorted order, allowing for efficient traversal of ranges (e.g., 'WHERE x ¿ 10' or 'WHERE x BETWEEN 5 AND 15'). Hash indexes are generally *not* efficient for range queries because the hashing function distributes keys randomly, breaking sequential order.
  - **Not Equal To:** Neither index type is particularly efficient for "not equal to" ('!=') predicates. Such queries typically require scanning a large portion of the data, as most values will satisfy the condition.

  Therefore, only "equal to" queries benefit significantly from both hash and B+ tree indexes.

Question 2. What is the selectivity of the following query when applied to the following instance of the Ratings relation? SQL query:

```
SELECT * FROM Ratings WHERE NOT (Stars > 3.0);
```

Table:

RatingID	Stars	UserID	MovieID
1	4.0	794	223
2	3.0	802	034
3	4.0	795	342
4	2.0	123	425
5	5.0	322	0

a) 60%

b) 20%

c) 80%

d) 40%

Answer: d) 40%

- *Explanation:* Selectivity is the fraction of rows that satisfy a query condition. The condition is 'NOT (Stars ¿ 3.0)', which is equivalent to 'Stars ¡= 3.0'. Let's check each tuple in the 'Ratings' table:
  - RatingID 1: Stars = 4.0. Is 4.0 ¡= 3.0? No.
  - RatingID 2: Stars = 3.0. Is 3.0 ¡= 3.0? Yes. (Tuple selected)
  - RatingID 3: Stars = 4.0. Is 4.0 ¡= 3.0? No.
  - RatingID 4: Stars = 2.0. Is 2.0 ¡= 3.0? Yes. (Tuple selected)
  - RatingID 5: Stars = 5.0. Is 5.0 ¡= 3.0? No.

  Two tuples (RatingID 2 and 4) satisfy the condition. There are a total of 5 tuples in the table. Selectivity = (Number of selected tuples) / (Total number of tuples) = 2 / 5 = 0.4 = 40%.

Question 3. Given a relation with 4 attributes, how many different hash indexes can be built on this relation?

a) 14

b) 20

c) 12

d) 18

Answer: d) 18

- *Explanation:* A hash index can be built on any single attribute or any combination of attributes. If a relation has 'N' attributes, the number of possible hash indexes is the number of non-empty subsets of those attributes. This is given by the formula $2^N - 1$. For a relation with 4 attributes (let's say A, B, C, D):
  - Single attribute indexes: A, B, C, D (4 combinations)
  - Two-attribute combinations: (A,B), (A,C), (A,D), (B,C), (B,D), (C,D) (6 combinations)
  - Three-attribute combinations: (A,B,C), (A,B,D), (A,C,D), (B,C,D) (4 combinations)
  - Four-attribute combinations: (A,B,C,D) (1 combination)

  Total $= 4 + 6 + 4 + 1 = 15$. However, the question implies that the order of attributes matters for composite keys in some contexts (e.g., for B-trees sometimes, but for hash, order typically doesn't matter for the key itself, just for unique sets). Let's re-evaluate based on the more common interpretation in database systems where each unique set of attributes can form a composite index. The number of non-empty subsets is $2^N - 1$. For N=4, $2^4 - 1 = 16 - 1 = 15$.

  Self-correction/Refinement: The common interpretation of "different hash indexes" often refers to distinct sets of columns, leading to $2^N - 1$. However, sometimes in a multiple choice question like this, if 15 isn't an option, it hints at considering permutations or a slightly different interpretation. Let's reconsider.

  If we consider that for a composite index, say on (A,B), it is distinct from (B,A) *in some specific index implementations*, or if the choices hint at a slightly different combinatorics. However, for hash indexes, the order of attributes in a composite key typically does not change the resulting hash value or the index structure for that set of attributes. A hash function takes a tuple of values (A,B) as input; the order might be fixed by convention, but (A,B) and (B,A) as *sets of attributes* are the same.

  Let's assume the question implicitly refers to *all possible non-empty ordered subsets* of attributes if it's not strictly a "set" interpretation, or if there's a misunderstanding of how the options are derived. Let's assume the more general understanding for *composite indexes* (which can be B-tree or hash-based) where 'INDEX (A, B)' is distinct from 'INDEX (B, A)'. For N attributes: 1 attribute: N choices (e.g., A, B, C, D) $= 4$ 2 attributes: $P(N, 2)$ permutations $= 4 \times 3 = 12$ (e.g., (A,B), (B,A), (A,C), (C,A)...) 3 attributes: $P(N, 3)$ permutations $= 4 \times 3 \times 2 = 24$ 4 attributes: $P(N, 4)$ permutations $= 4 \times 3 \times 2 \times 1 = 24$ This leads to a very large number.

  Revisiting the problem context: In database systems, an index is defined on a *set of columns*, and the order often matters for B+ trees for prefix matching, but for hash indexes, it's generally about

the set. The options provided (14, 18, 12, 20) are relatively small. This strongly suggests the question might be about non-empty subsets PLUS perhaps distinct orders for a subset. Let's re-evaluate the options with the simplest interpretation: Number of attributes: 4 (A, B, C, D) Number of non-empty subsets: $2^4 - 1 = 15$. Since 15 is not an option, there's likely a nuance being tested.

Could it be asking for "how many different *ways* to construct a hash index" based on properties? Let's re-examine standard counting principles. If a hash index can be on *any combination of attributes*, it's the power set minus the empty set. $2^4 - 1 = 15$. Perhaps the question is not about *all* possible hash indexes, but limited to single-attribute and a specific number of composite indexes?

Let's consider the possible interpretations for why the answer is 18, as 15 is the standard combinatoric result for distinct sets of attributes. One common scenario where N-choose-K is augmented is if attributes can be indexed independently and combined, but that's usually for multi-column indexes rather than distinct hash indexes.

This specific question might be testing a very particular definition or a common misinterpretation. If we exclude the 4-attribute combination, we have $15 - 1 = 14$. This is option (a). If we consider only single attributes and pairs? $4 + 6 = 10$. Let's stick to the $2^N - 1$ as the most general interpretation for "different hash indexes" as distinct sets of columns indexed. Given that 15 is not an option and 18 is, this points to a specific pedagogical context or a slight deviation from the typical $2^N - 1$ interpretation.

Hypothesis for 18: Could it be $N^2 + N/2$ or something similar? $4^2 + 4/2 = 16 + 2 = 18$? No. N single attributes: 4 $N(N-1)/2$ combinations of 2 attributes: $4 \times 3/2 = 6$ $N(N-1)(N-2)/6$ combinations of 3 attributes: $4 \times 3 \times 2/6 = 4$ 1 combination of 4 attributes. Total $= 4 + 6 + 4 + 1 = 15$.

Let's consider if the question implies order matters for two-attribute and three-attribute combinations, but not for one or four. This would be a very unusual interpretation for "hash index" where order is typically irrelevant to the hash value itself.

Re-evaluating based on common test question pitfalls or specific curriculum: If it's about *ordered permutations* of attributes for composite keys, for N attributes, the number of distinct ordered sequences of attributes that can form a key is: $P(N,1) + P(N,2) + P(N,3) + ... + P(N,N)$ where $P(N,k) = N!/(N-k)!$ For N=4: $P(4,1) = 4$ $P(4,2) = 4 \times 3 = 12$ $P(4,3) = 4 \times 3 \times 2 = 24$ $P(4,4) = 4 \times 3 \times 2 \times 1 = 24$ Total $= 4 + 12 + 24 + 24 = 64$. This is far too high.

The most plausible scenario for 18 as an answer, given 4 attributes: If it were $N \times (N+1)/2$ for unique pairs plus singletons. This question's answer of 18 is an outlier if based purely on "distinct sets of attributes". Let's go with the provided answer and try to reverse-

engineer a plausible reason, which is typical in some multiple-choice scenarios.

A common interpretation for number of ways to pick k items from N *with replacement* where order matters, or something complex.

Let's consider a scenario that might lead to 18: Number of single attribute indexes: 4 Number of 2-attribute combinations (order does not matter): 6 Number of 3-attribute combinations (order does not matter): 4 Number of 4-attribute combinations (order does not matter): 1 Total combinations = 15.

The only way to get to 18 would be if certain "types" of indexes were allowed. Perhaps 4 (single) + 6 (pairs, unordered) + some other category. What if it's $N^2 + 2$? No.

The most likely, if not purely combinatoric, reason for 18: This specific number (18 for N=4) sometimes appears in contexts where there are N single-column indexes, and then a specific number of composite indexes based on some pattern. Without further context or a specific formula provided by the course, deriving 18 purely from standard set theory on 4 attributes is difficult if 15 (which is $2^N - 1$) is the expected number of unique attribute *sets*.

However, let's consider that sometimes the question implies *ordered pairs* for composite keys, and then also single keys. If we take single keys (4) and ordered pairs of keys ($P(4, 2) = 12$), that sums to $4 + 12 = 16$. Still not 18. If it includes $P(4, 2)$+ single column indexes.

This question's answer (18) is the most difficult to justify based on standard database theory for "number of distinct hash indexes" where a hash index is typically defined over a *set* of attributes. The answer of 15 ($2^N - 1$) is the standard combinatorial result. However, for the sake of providing an explanation consistent with the provided answer, this implies a counting method that goes beyond simple power set cardinality. Without a specific rule or context from the course, it's hard to explain 18 precisely. It might stem from an obscure counting rule or a specific interpretation of "different hash indexes" where certain permutations are counted, even if not typical for hash index definitions. Given this ambiguity, for educational purposes, I'll state the standard understanding and acknowledge the discrepancy.

Revised Explanation for Q3: A hash index can be built on any single attribute or any combination of attributes. The number of unique sets of attributes from a set of N attributes is $2^N - 1$ (excluding the empty set). For 4 attributes, this would be $2^4 - 1 = 15$. However, the given answer is 18. This suggests a different counting method or interpretation. One possible, though less common, interpretation could involve considering certain ordered combinations of attributes as "different" indexes, or perhaps a context-specific rule from the course material that is not standard $2^N - 1$ combinatorics.

Without further information, the most direct explanation based on fundamental database principles points to 15. If 18 is definitively the correct answer in your course, it implies a nuance that should be clarified by your instructor or course materials. For the purpose of this explanation, we will acknowledge the given answer.

Question 4. Compute the intersection of the relations R(A,B,C) and S(A,B,C): Table R:

a	b	c
1	2	3
4	2	3
4	5	6
2	5	3
1	2	6

Table S:

a	b	c
2	5	3
2	5	4
4	5	6
1	2	3

Which of the following tuples appears in R ∩ S?

a) (1,2,3)

b) (2,5,4)

c) (2,4,3)

d) (1,2,6)

Answer: a) (1,2,3)

- *Explanation:* The intersection of two relations (R ∩ S) includes only those tuples that are present in *both* relations. Let's compare the tuples:
  - (1,2,3): Present in R, Present in S. -¿ IN INTERSECTION
  - (4,2,3): Present in R, Not in S.
  - (4,5,6): Present in R, Present in S. (So (4,5,6) would also be in the intersection, but it's not an option).
  - (2,5,3): Present in R, Present in S. (So (2,5,3) would also be in the intersection, but it's not an option).

– (1,2,6): Present in R, Not in S.
– (2,5,4): Not in R. (So not in intersection)

From the given options, only (1,2,3) is present in both R and S.

Question 5. Suppose relation R(A,B) has the tuples: relation R(A,B):

A	B
1	2
3	4
5	6

and the relation S(B,C,D) has tuples:

a	b	c
2	4	6
4	6	8
4	7	9

Compute the theta-join of R and S with the condition R.A ¡ S.C AND R.B ¡ S.D. Then, identify from the list below one of the tuples in R $\bowtie_{R.A<S.C \text{ AND } R.B<S.D}$ S. You may assume the schema of the result is (A, R.B, S.B, C, D).

a) (5,6,2,4,6)

b) (3,4,5,7,9)

c) (1,2,2,6,8)

d) (1,2,2,4,6)

Answer: d) (1,2,2,4,6)

- *Explanation:* A theta-join combines tuples from two relations based on a specified condition. The schema of the result is a concatenation of the schemas of R and S. The join condition is 'R.A ¡ S.C AND R.B ¡ S.D'.

  Let's iterate through each tuple in R and compare it with each tuple in S:

  From R: (1,2)

  – Against S: (2,4,6)
  – R.A ¡ S.C (1 ¡ 4)? Yes.
  – R.B ¡ S.D (2 ¡ 6)? Yes.

- Both conditions met. Result tuple: (1,2,2,4,6). This matches option (d).
- Against S: (4,6,8)
- R.A ¡ S.C (1 ¡ 6)? Yes.
- R.B ¡ S.D (2 ¡ 8)? Yes.
- Both conditions met. Result tuple: (1,2,4,6,8). (Not an option)
- Against S: (4,7,9)
- R.A ¡ S.C (1 ¡ 7)? Yes.
- R.B ¡ S.D (2 ¡ 9)? Yes.
- Both conditions met. Result tuple: (1,2,4,7,9). (Not an option)

From R: (3,4)

- Against S: (2,4,6)
- R.A ¡ S.C (3 ¡ 4)? Yes.
- R.B ¡ S.D (4 ¡ 6)? Yes.
- Both conditions met. Result tuple: (3,4,2,4,6). (Not an option)
- Against S: (4,6,8)
- R.A ¡ S.C (3 ¡ 6)? Yes.
- R.B ¡ S.D (4 ¡ 8)? Yes.
- Both conditions met. Result tuple: (3,4,4,6,8). (Not an option)
- Against S: (4,7,9)
- R.A ¡ S.C (3 ¡ 7)? Yes.
- R.B ¡ S.D (4 ¡ 9)? Yes.
- Both conditions met. Result tuple: (3,4,4,7,9). (Not an option)

From R: (5,6)

- Against S: (2,4,6)
- R.A ¡ S.C (5 ¡ 4)? No. (Condition fails)
- Against S: (4,6,8)
- R.A ¡ S.C (5 ¡ 6)? Yes.
- R.B ¡ S.D (6 ¡ 8)? Yes.
- Both conditions met. Result tuple: (5,6,4,6,8). (Not an option)
- Against S: (4,7,9)
- R.A ¡ S.C (5 ¡ 7)? Yes.
- R.B ¡ S.D (6 ¡ 9)? Yes.
- Both conditions met. Result tuple: (5,6,4,7,9). (Not an option)

From the given options, only (1,2,2,4,6) is a valid result of the theta-join.