CS143 In Person: Quiz 1

[redacted]

TOTAL POINTS

88.5 / 100

QUESTION 1

Relational Algebra 20 pts

1.1 Transaction 10 / 10

✓ **- 0 pts** **Correct**

Projection

- 0.5 pts Missing/incorrect projection symbol $$\pi$$ for final output

- 1 pts Missing/incorrect `T1.customer` column projection attributes (including missing relation prefix)

Selection

- 0 pts Click here to replace this description.

- 0.5 pts Missing/incorrect selection operator symbol $$\sigma$$

- 0.5 pts Missing/incorrect $$T1.customer = T2.customer$$ condition (possibly implied if natural join was used)

- 0.5 pts Missing/incorrect $$T1.item \neq T2.item$$ condition (not double-counted vs incorrect natural joins)

- 1 pts Missing/incorrect $$ T1.price \ge 100$$ condition (including ambiguity)

- 1 pts Missing/incorrect $$ T2.price \ge 100$$ condition (including ambiguity)

- 0.5 pts (Partial credit) Missing equality condition $$price = 100$$

Rename

- 0 pts Click here to replace this description.

- 1 pts Missing/incorrect rename operator $$\rho$$ of at least one `Transaction` relation

- 0.5 pts (Partial credit) Incorrect usage of rename operator (e.g., missing parentheses)

- 1 pts Missing/incorrect rename alias of at least one `Transaction` relation

Cross Product

- 0 pts Click here to replace this description.

- 3 pts Missing/incorrect cross product $$\times$$ operator (except natural join)

- 2 pts (Partial credit) Incorrect usage of natural join ($$\bowtie$$) instead of cross product ($$\times$$)

- 2 pts Unnecessary additional operations (e.g., set difference, intersection).

- 0.5 pts Each incorrect additional selection condition (e.g. on `date` or unequal `price`)

1.2 Nontrivial Property 10 / 10

✓ **- 0 pts** **$$A \rightarrow C$$ (Correct)**

- 5 pts All values in C are identical

- 3 pts A is a key

- 5 pts $$A \leftrightarrow C$$

- 10 pts C is a key ($$C \rightarrow A$$)

- 10 pts R is empty

- 7 pts AB is a key

- 5 pts A and B keys

- 10 pts Self natural joins return the original table

- 3 pts X is a key

- 10 pts ABC is a key

- 10 pts B is a key

- 3 pts A and C identical

- 8 pts $$V = W$$

- 5 pts A and C keys

- 10 pts Blank

- 10 pts Incorrect

- 6 pts All attributes are keys

- 10 pts A not unique

- 10 pts AC key

QUESTION 2

Query Equivalence 20 pts

2.1 1 5 / 5

✓ **- 0 pts** **Correct**

- 5 pts Yes (incorrect)

- 1 pts Correct, example violates constraints

- 1 pts Correct, no database instance

- 0 pts Correct, database has no `A` column

- 5 pts Correct answer, incorrect explanation / example

- 0 pts Correct, no description at all

2.2 2 5 / 5

✓ **- 0 pts** **Correct**

+ 3 pts Correct, empty table explanation

- 2.5 pts No mention that key means no duplicates

- 5 pts No without empty database (incorrect)

- 5 pts Incorrect explanation

- 5 pts Correct, no explanation

2.3 3 5 / 5

✓ **- 0 pts** **Correct**

- 5 pts Yes (incorrect)

- 5 pts Correct, example violates constraint

- 0 pts Correct, Null explanation

- 0 pts Correct, no database instance

- 5 pts Correct, incorrect single row example

- 5 pts Correct, incorrect example

- 5 pts Correct, no justification

2.4 4 5 / 5

✓ **- 0 pts** **Correct**

- 5 pts Yes (incorrect)

- 5 pts Correct response, example incorrect

- 5 pts Example violate constraints

- 1 pts Example violates constraints, but otherwise works

- 5 pts Correct, no justification

QUESTION 3

3 SQL Query 10 / 10

✓ **- 0 pts** **Correct**

- 1 pts minor error in recursion part of query

- 3 pts wrong recursion base case

- 3 pts wrong recursion step

- 4 pts wrong non-recursion part of the query (not using GROUY BY or subquery get number of descendants of every person)

- 3 pts (mostly) correctly finding Vanessa's descents not correctly finding others descendent

- 7 pts major syntax error

- 2 pts partially wrong non-recursion part of the query (Have some attempt of grouping, but syntax error or wrong result)

QUESTION 4

ER Model 20 pts

4.1 Diagram 10 / 10

✓ **- 0 pts** **Correct**

- 1 pts Luggage to Passengers cardinality incorrect

- 1 pts Passengers to Flights cardinality incorrect

- 1 pts Flights to Destinations cardinality incorrect

- 1 pts Flights to Luggage cardinality incorrect

- 1 pts Flights to Luggage not an identifying relationship (double diamond)

- 2 pts Missing a relationship (no cardinality points deducted)

- 4 pts Missing 2 relationships (no cardinality points deducted)

- 2 pts Didn't follow instructions regarding binary relationships (graded by 'splitting' into binary relationships)

- 1 pts Extra relationship

- 1 pts 1x extra identifying relationship (extra double diamond)

- 3 pts 3x extra identifying relationship (extra double diamonds)

- 0 pts Mixed usage of numerical and arrow/line cardinality notation (Number notation is graded if present on both sides, otherwise only arrow-number notation is graded except P-L. Handled individually per relationship.)

- 0.25 pts Invalid cardinality notation (e.g., $$\infty$$ instead of *, single number or list in place

of range)
(Partial credit on otherwise correct answer)

- **2 pts** Added unnecessary attributes on relationships
- **2 pts** dashed line between Flight and Luggage
- **1 pts** wrong arrow direction (pointing to relationship)
- **1 pts** Luggage not a weak entity (double boxed)
- **1 pts** Missing `Luggage`
- **1 pts** Missing `Passengers`
- **1 pts** Missing `Destinations`
- **1 pts** Missing `Flights`
- **0.5 pts** Wrong/missing key on 1 table
- **1 pts** Wrong/missing key on 2 tables
- **2 pts** All 4 keys wrong/missing
- **0.25 pts** No dashed underline on luggage number (weak key)
- **3 pts** Missing the key and attributes for all entities
- **2 pts** 2x extra identifying relationship (extra double diamonds)
- **1 pts** extra weak entities
- **2 pts** All 4 keys wrong / missing

4.2 Schema 8.5 / 10

- **0 pts** Correct
- **2 pts** Extra Table
- **1.5 pts** Table ``Passengers`` incorrect
- **1.5 pts** Table ``Flight`` incorrect
- **1.5 pts** Table ``Destinations`` incorrect
- **1.5 pts** Table ``Luggage`` incorrect
- **1.5 pts** Table ``Take`` incorrect

✓ **- 1.5 pts Table ``BelongTo`` incorrect**

- **1.5 pts** Table ``FlyTo`` incorrect
- **1.5 pts** Missing table ``Take``
- **1.5 pts** Missing Table ``FlyTo``
- **1.5 pts** Missing table ``BelongTo``
- **1.5 pts** Missing table ``Destination``
- **1.5 pts** Missing table ``Luggage``
- **0.5 pts** Missing key for table ``Luggage``
- **0.5 pts** Missing key for table ``Destination``
- **0.5 pts** Incorrect keys for ``BelongTo``
- **0.5 pts** Incorrect keys for Table ``Luggage``
- **0.5 pts** Incorrect keys for Table ``FlyTo``
- **0.5 pts** Incorrect keys for Table ``Take``
- **0.5 pts** Incorrect keys for Table ``Destination``
- **1.5 pts** Missing keys for Tables
- **1.5 pts** Incorrect keys for relation table
- **4.5 pts** relation related table incorrects
- **4.5 pts** All relation related table missing
- **10 pts** empty solution

QUESTION 5

Relational Design Theory 30 pts

5.1 Cardinality 0 / 10

- **0 pts** Correct
- **4 pts** Fail to calculate the correct number if pi_A(R)
- **2 pts** Partial error in calculating pi_A(R) (A is a key so the number of pi_A(R) is a determined value, not a range)
- **2 pts** Fail to get the minimum number of pi_C(R)
- **2 pts** Fail to get the maximum number of pi_C(R)
- **2 pts** Wrong final calculation of the cross product
- **1 pts** Know how to calculate cross product but wrong final answer because of wrong pi_A(R) or pi_C(R)
- **1 pts** The final value is two specific values, not a range

✓ **- 10 pts No answer or no value calculation (need to be specific value).**

5.2 Losslessness 10 / 10

✓ **- 0 pts Correct**

- **10 pts** Wrong answer of whether the decomposition is lossless while the explanation is in wrong direction
- **8 pts** Wrong answer of whether the decomposition is lossless. But the explanation is in the correct direction though the conclusion is wrong
- **5 pts** Correct in choosing NO but the added FD is wrong or missing
- **2 pts** Correct in choosing NO but the added FD is not simplest, or partially wrong

5.3 BCNF 10 / 10

✓ **- 0 pts** Correct

- 10 pts Wrong answer of whether BCNF

- 2 pts Fail to handle A->B in decomposition

- 2 pts Fail to handle CD->E in decomposition

- 2 pts Fail to handle E->D in decomposition

- 0.5 pts Key of Table (A,B) is wrong or missing

- 0.5 pts Key of Table (A,C,D) is wrong or missing

- 0.5 pts Key of Table (C,E) is wrong or missing

- 0.5 pts Key of Table (D,E) is wrong or missing

- 2 pts Use a wrong FD, see details in the quiz paper

- 0.5 pts Key of Table (A,C,E) is wrong or missing

- 10 pts No answer

UCLA
Computer Science Department
Fall 2021

Instructor: J. Cho

CS143 Quiz 1: 2 hours

Student Name: [redacted]

Student ID: [redacted]

(IMPORTANT PLEASE READ **):**

- The exam is *closed book* and *closed notes.* You may use *one double-sided cheat-sheets.* You can use a calculator.
- *Simplicity and clarity of your solutions will count.* You may get as few as 0 point for a problem if your solution is far more complicated than necessary, or if we cannot understand your solution.
- If you need to make any assumption to solve a question, *please write down your assumptions.* To get partial credits, you may want to write down how you arrived at your answer step by step.
- If a question asks for a numeric answer, you don't have to calculate. You may just write down a numeric expression.
- Please, write your answers neatly.

Problem	Score	
1	20	
2	20	
3	10	
4	20	
5	30	
Total	100	

Problem 1: Relational Algebra (20 points)

1. Consider the relation

   Transaction(customer, item, price, date).

   The tuple ('John', 'Macbook Air', 999, 2021-10-01) in the relation means that John purchased a Macbook Air at the price of $999 on 10/01/2021.

   set semantics

   Write a relational algebra expression that returns all customers who purchased at least two different items of price $100 or above. You may assume that customer and item names are unique. You may also assume no NULL values in the table.

   Note that relational algebra does **NOT** have an aggregate function and simplicity and clarity of your answer count as well as correctness.

$$\pi_{T1.customer}\left(\sigma_{T1.customer = T2.customer \wedge T1.item <> T2.item \wedge T1.price \geq 100 \wedge T2.price \geq 100}\left(\rho_{T1}(Transaction) \times \rho_{T2}(Transaction)\right)\right)$$

2. Consider a relation $R(A, B, C)$. You may assume there are no NULL values or duplicate tuples in R.

   Consider the following relational query:

   $$\sigma_{V \neq W}(\rho_{R_1(X,G,V)}(R) \bowtie \rho_{R_2(X,H,W)}(R))$$

   Assume the result of the above query is always guaranteed to be empty. Using 30 words or less, state a nontrivial property/constraint/fact on R that is guaranteed to hold if and only if this assumption is true. Keep in mind that there is a very concise correct answer (e.g., "C is a key of R") corresponding to a concept covered in class. (10 points)

$\rho_{R_1}(X,G,V)(R) \bowtie \rho_{R_2}(X,H,W)(R)$
will only contain tuples that have the same X values.
Since the final result is empty, it means
when two tuples have the same X values, they must also have
V value of 1st tuple = W value of 2nd tuple

in other words, when two tuples have the same A values, they are guaranteed to have the same C values.

This means that

A functionally determines C, or $A \rightarrow C$

Problem 2: Query Equivalence (20 points)

Two queries are considered equivalent if they return exactly the same results for all database instances.

Assume the following for the relation $R(\underline{A}, B)$ referenced in the queries.

- The attribute A is a key.
- The attribute B is not a key.
- The attribute A may not have NULL, but the attribute B may have NULL.

Be mindful of the above assumptions since they are critical to get the correct answers. Do not make any other assumptions.

For each of the pair of queries listed below, write "YES" if the two queries are equivalent and "NO" otherwise. If your answer is YES, briefly explain your answer. If your answer is NO, write down a simple database instance (e.g., $\{(1, 2), (3, 4)\}$) that returns different answers to the two queries.

Pay special attention to NULL values and duplicates. Note that you may get as low as zero point if your YES/NO answer is correct but your justification/example is not.

1. YES/NO

   1. `SELECT SUM(B)/COUNT(*) FROM R;`
   2. `SELECT AVG(B) FROM R;`

NO

fa.

{(1, NULL), (2,3), (4,7)}, the first query will return $\frac{10}{3} = 3.33$, the second query will return $\frac{10}{2} = 5$.

2. YES/NO

   1. ```
      SELECT A
      FROM R
      WHERE A >= ALL (SELECT A FROM R);
      ```

   2. `SELECT MAX(A) FROM R;`

YES. The only way the first query could not be equivalent to the second is the presence of duplicate max values, in which case the first query would return all duplicates, while the second would only return tuple. But since A is a key, this can never happen. Thus, they're equivalent.
      ```

3. YES/NO

   1. ```
      SELECT B
      FROM R R1
      WHERE B <= ALL (SELECT B FROM R R2 WHERE R1.A <> R2.A);
      ```

   2. `SELECT MIN(B) FROM R;`

R1 R2

T, 1 (5) ≤ (1 (5), 2 (5))
2 (5)

NO.
Consider the case {(1, 5), (2, 5)}
the first query would return (5, 5),
but the second query would only
return 5.

5 ≤ ALL (5)
5 ≤ ALL (5)

4. YES/NO

   1. ```
      (SELECT B FROM R) EXCEPT (
          SELECT R1.B FROM R R1, R R2
          WHERE R1.B = R2.B AND R1.A <> R2.A);
      ```
      ] ALL B's
      ]– All B's that occur more than once
      all B's that occur exactly once.

   2. `SELECT B FROM R GROUP BY B HAVING COUNT(*) = 1;` group by all B's, occuring exactly once. does count nulls.

NO.
~~Consider~~ {(1, NULL), (2, NULL)}
The first query will not select anything in the
second part of the except clause, and
using set semantics, will return
{NULL, NULL} − {} ⇒ {NULL}

The second query will group by NULL, and
will have 2 NULL values. (count(*) = 2)
Thus, it will not return anything ⇒ { }

If there is a single NULL B, the first query will return
it since the second part of the except clause will return
no values. As will the second query.
If there are multiple NULL B's, the first query will return one
null value,

Problem 3: SQL Query (10 points)

Consider the Parent(parent, child) table that has parent-child relationship. For example, the tuple (John, Vanessa) in the table means that John is a parent of Vanessa. Assume that a person's name is unique.

Someone's descendants are the people in later generations who are born from them (directly or indirectly). Write a SQL query that returns the names of the people who have more children descendants than Vanessa. Please write your query neatly and concisely. Significant points may get deducted if your solution is far more complicated than necessary, or if we cannot understand your solution.

```
WITH RECURSIVE Ancestor(child, ancestor) as (
    (SELECT child, parent FROM Parent)
    UNION
    (
        SELECT A.child, P.parent
        FROM Parent P, Ancestor A
        WHERE P.child = A.ancestor
    )
)

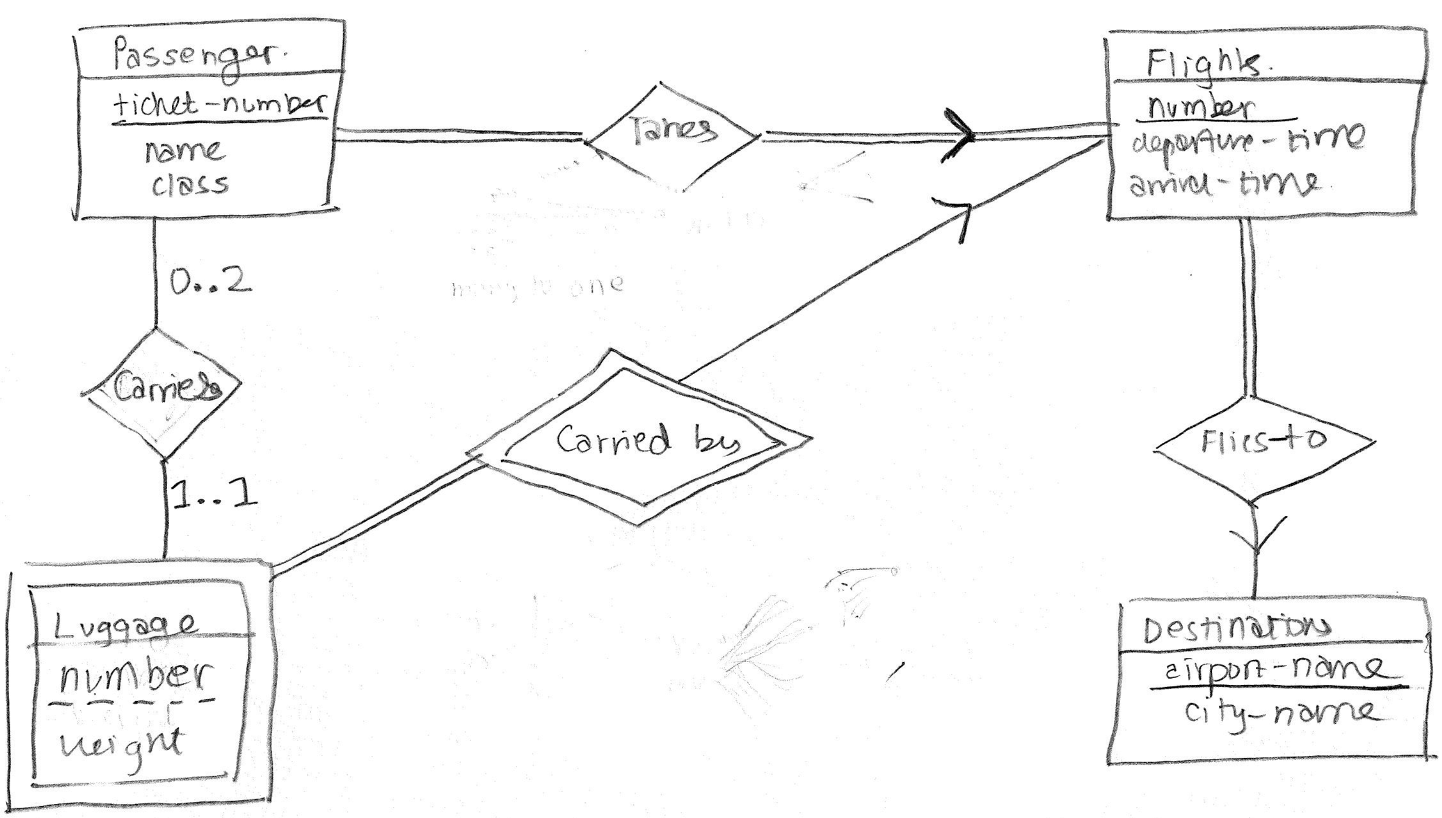
SELECT ancestor
FROM Ancestor
GROUP BY ancestor
HAVING COUNT(*) > (
    SELECT COUNT(*)
    FROM Ancestor
    WHERE ancestor = "Vanessa"
);
```

Problem 4: ER Model (20 Points)

Consider the following entity sets and their relationships:

- There are four entity sets: Passengers, Flights, Luggage and Destinations.
- Passengers have the attributes ticket-number, name, and class. Passenger ticket-numbers are unique.
- Flights have the attributes number, departure-time and arrival-time. Flight numbers are unique.
- Luggage have the attributes number and weight. Luggage numbers are numbers assigned to each piece of luggage within its corresponding flight, thus they are unique inside the flight, but can be the same across flights.
- Destinations have the attributes city-name and airport-name. Airport names are unique.
- Every passenger takes exactly one flight, and every flight contains at least one passenger.
- Every passenger carries 0, 1 or 2 pieces of luggage, and every luggage belongs to exactly one passenger and is carried by exactly one flight.
- Every flight has exactly one destination, and multiple flights can have the same destination.

1. Draw an E-R diagram using for the above scenario. In drawing the diagram use *only binary relationship sets*. Specify the cardinality and participation of relationship sets either using arrows and double lines or the general cardinality notation (like 1..3) whenever necessary. Note that some cardinality constraints may not be expressible using arrows. Also indicate whether an entity set is strong or weak, and the key or discriminator of every entity set. You can use any reasonable name for relationship sets and their attributes. (10 points)



2. Convert the ER diagram into tables and write the schema of all tables. Underline the key of each table, like Movie(<u>mid</u>, title, year). (10 points)

Passenger (<u>ticket-number</u>, name, class)

Flights (<u>number</u>, departure-time, arrival-time)

Destinations(<u>airport-name</u>, city-name)

Luggage (<u>Luggage-number</u>, <u>Flights.number</u>, weight)

Takes (<u>ticket-number</u>, number)

Carries (ticket-number, <u>number</u>)

Flies-to (<u>number</u>, airport-name)

Problem 5: Relational Design Theory (30 points)

1. Consider the relation $R(A, B, C, D)$, on which functional dependencies $A \rightarrow B$, $B \rightarrow C$, and $C \rightarrow D$ hold. If $|R| = 10$, $|\pi_B(R)| = 5$, and $|\pi_D(R)| = 4$, what are the possible value(s) of $|\pi_A(R) \times \pi_C(R)|$? Here, the notation $|R|$ means the number of tuples in R. Note that we are dealing with relational algebra operators that are based on the set semantics. (10 points)

F: {A→B, B→C, C→D}

F+: {A→BCD, B→D, B→C, C→D} ie. A is a key.

$\pi_A(R) \times \pi_C(R)$ will contain all tuples (A, C) that are unique.

since A→C,

if two tuples have the same A, values, they must have the same C values

Thus, the maximum number of unique (A, C) tuples are the total number of combinations − total initial values

2. Suppose that we decompose the table $R(A, B, C, D, E)$ into $R1(A, B, C)$ and $R2(A, D, E)$. Assume the following functional dependencies hold on R.

$A \rightarrow B$, $CD \rightarrow E$, $E \rightarrow A$.

Is the decomposition lossless? YES/NO

If your answer is YES, please explain why. If your answer is NO, please provide a simplest functional dependency (that involves the least number of attributes) whose addition will make the decomposition lossless. (10 points)

a1 c1
a1 c2 } a1c1
a1 c3

a1 c1
a1 c2
a1 c3
a2 c1
a2 c2
a2 c3
a3 c1
a3 c2
a3 c3

R(A, B, C, D, E) → R1 (A, B, C)
R2 (A, D, E)

The shared attribute is A.

for R1, A+ = {A, B} ≠ all attributes

for R2, A+ = {A} ≠ all attributes

Thus, A isn't a key, and the decomposition isn't lossless

If we add B→C, then A+ = {A, B, C} and A is a key for R1 for R1, ie. lossless decomposition

3. Consider a relation $R(A, B, C, D, E)$ and a set of functional dependencies:

F: $A \to B$, $CD \to E$, $E \to D$. F+ = {A → B, CD → E, E → D}.

Is the table in BCNF? If yes, explain why the functional dependencies do not violate the BCNF definition. If no, normalize it into BCNF. If you normalize the table, underline the key in the normalized tables. (10 points)

for A → B

A+ = {A, B} clearly, A is not a key.

we can decompose into

R2(A, B), R3(A, C, D, E)

R2 in BCNF since it only has two attributes,

For R3 (A, C, D, E).

for E → D,

E+ = {E, D} clearly, E is not a key.

we can decompose R3 into

R4(E, D) and R5(E, A, C)

R4 in BCNF since it only has two attributes.

for R5(E, A, C),

there are no non-trivial functional dependencies.

Thus, R5 is in BCNF

Our final answer is

R2 (A, B)

R4 (E, D)

R5 (E, A, C)