# CS145 Final Examination

Thursday, December 10, 1998, 8:30AM—11:30AM

## Directions

The exam is *open book/notes*; any written materials may be used. All questions about SQL refer to the SQL2 or SQL3 standard, *not* necessarily to the Oracle implementation of SQL used in the project. Several of the questions ask you to determine if some SQL code is correct or incorrect. If incorrect, there will always be some significant flaw in what the code does; it will not be something trivial like missing punctuation.

For each of the 33 questions, circle the letter (a), (b), (c), or (d) of your chosen answer. Score = 3 times number right, minus number wrong, so random guessing nets you nothing on the average. One point is offered for spelling your name correctly, so the maximum score is 100.

If you wish to explain or demonstrate your solution to a problem for partial credit, you may use page bottoms or the backs of the pages (but warn us on the front). Please use this option sparingly, e.g., if you think the question is flawed or open to multiple interpretations, because we shall only be awarding partial credit in rare situations.

Do not forget to **sign the pledge** below.

I acknowledge and accept the honor code.

______________________________

Print your name here (**1 pt.**): ____________________________________________

---

The first two questions refer to a relation $R(A, B, C, D, E)$ with the following functional dependencies: $ABC \rightarrow DE$ and $D \rightarrow AB$. (Note we are using multi-attribute right sides as a shorthand.)

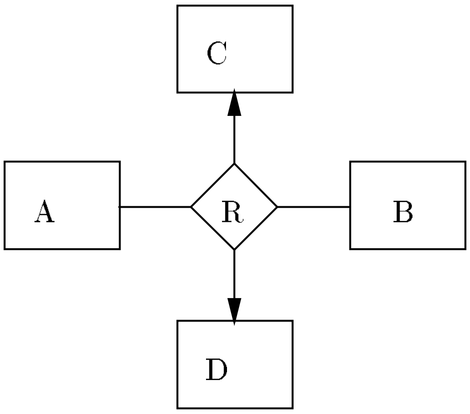
**Question 1:** Which of the following is true of $R$?

(a) $R$ is not in 3NF. (b) $R$ is in 3NF but not BCNF. (c) $R$ is in BCNF but not 4NF. (d) $R$ is in 4NF.

**Question 2:** The number of superkeys of $R$ is:

(a) 2 (b) 7 (c) 10 (d) 12

**Question 3:** Consider the following E/R diagram:




Below are three possible relationship sets for this E/R diagram:

| | $A$ | $B$ | $C$ | $D$ |
|---|---|---|---|---|
| I. | $a_1$ | $b_1$ | $c_1$ | $d_1$ |
| | $a_1$ | $b_1$ | $c_1$ | $d_2$ |

| | $A$ | $B$ | $C$ | $D$ |
|---|---|---|---|---|
| II. | $a_1$ | $b_1$ | $c_1$ | $d_1$ |
| | $a_1$ | $b_1$ | $c_2$ | $d_2$ |

| | $A$ | $B$ | $C$ | $D$ |
|---|---|---|---|---|
| III. | $a_1$ | $b_1$ | $c_1$ | $d_1$ |
| | $a_1$ | $b_2$ | $c_1$ | $d_1$ |

You may assume that different symbols stand for different values, e.g., $d_1$ is definitely not equal to $d_2$. Which of the above could *not* be the relationship set for the E/R diagram?

(a) I only (b) I and II only (c) II only (d) I, II and III.

---

**Question 4:** Consider relation $S(A, B, C)$ with dependencies $A \to B$ and $B \twoheadrightarrow C$. Of the following dependencies:

I. $B \to C$
II. $A \twoheadrightarrow C$
III. $B \twoheadrightarrow A$

which must *necessarily* (i.e., for all instances of $S$ that satisfy $A \to B$ and $B \twoheadrightarrow C$) hold in $S$?

(a) II only (b) III only (c) I and II only (d) II and III only.

The next two questions are based on the following database. Automobiles have attributes `make`, `model`, `engine` (e.g., 6-cylinder), and `length`; `make` and `model` are the key. SUV's (sport-utility vehicles) are a subclass of automobiles and have the additional attribute `drive` (two- or four-wheel drive). A design for an automobiles database would have a relation `Automobiles(make, model, engine, length)`. We could incorporate SUV's by either of the following two options:

I. ODL-style: add a relation `SUV(make, model, engine, length, drive)`. SUV's are represented only in the `SUV` relation, not in `Automobiles`, while automobiles that are not SUV's are represented only in `Automobiles`.

II. E/R style: add a relation `SUV(make, model, drive)`. All automobiles are represented in `Automobiles`, and those automobiles that are SUV's are also represented in `SUV`.

**Question 5:** Which of the following queries, if commonly asked in the database, would justify using I instead of II?

(a) Find the make and model of all automobiles with a 6-cylinder engine and 4-wheel drive.
(b) Find the make and model of all automobiles with 6-cylinder engine.
(c) Find the make and model of all automobiles with 4-wheel drive.
(d) None of the above.

**Question 6:** Which of the following queries, if commonly asked in the database, would justify using II instead of I?

(a) Find the make and model of all automobiles with a 6-cylinder engine and 4-wheel drive.
(b) Find the make and model of all automobiles with 6-cylinder engine.
(c) Find the make and model of all automobiles with 4-wheel drive.
(d) None of the above.

---

**Question 7:** One of the following four expressions of relational algebra is not equivalent to the other three. They are all based on the relations $R(A, B)$ and $S(B, C)$. Indicate which is not equivalent to the others.

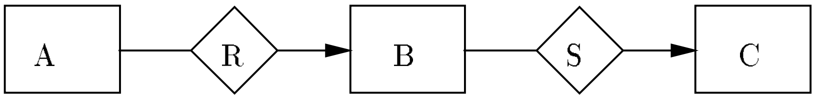
(a) $\pi_{AB}(R \bowtie S)$ (b) $R \bowtie \pi_B(S)$ (c) $R \cap \big(\pi_A(R) \times \pi_B(S)\big)$ (d) $\pi_{A,R.B}(R \times S)$.

**Question 8:** Of the following three equivalences between expressions of relational algebra, each involving relations $R(A,B)$ and $S(C,D)$ (note the schema of $S$ is different from that of the question above), which is true?

(a) $\pi_{A,B}(R \times S) = R$
(b) $R - \rho_{T(A,B)}(S) = \rho_{T(A,B)}\big(S - \rho_{U(C,D)}(R)\big)$
(c) $\pi_{A,B,D}(R \underset{B=C}{\bowtie} S) = R \bowtie \rho_{T(B,D)}(S)$
(d) none of the above (i.e., they are all false).

---

**Question 9:** Below is an E/R diagram:




Indicate which of the ODL specifications best mimics the intent of this E/R diagram. In both E/R and ODL, we have omitted mention of all attributes, which you may thus ignore.

```
(a) interface A {relation Set<B> R inverse B::R;};
    interface B {relation Set<A> R inverse A::R;
                 relation Set<C> S inverse C::S;};
    interface C {relation Set<B> S inverse B::S;};

(b) interface A {relation B R inverse B::R;};
    interface B {relation Set<A> R inverse A::R;
                 relation C S inverse C::S;};
    interface C {relation Set<B> S inverse B::S;};

(c) interface A {relation Set<B> R inverse B::R;};
    interface B {relation A R inverse A::R;
                 relation Set<C> S inverse C::S;};
    interface C {relation B S inverse B::S;};

(d) interface A {relation B R inverse B::R;};
    interface B {relation A R inverse A::R;
                 relation C S inverse C::S;};
    interface C {relation B S inverse B::S;};
```

**Question 10:** Which of the following Datalog programs describes true first cousins, i.e., individuals that have a common grandparent but not a common parent? You may assume that $P(x, y)$ is true if and only if $y$ is a parent of $x$. However, do not assume the database is constrained by popular conventions against incest, poligamy, etc.

I.
```
Cousin(x,y) <- P(x,xp) AND P(xp,z) AND P(y,yp) AND P(yp,z) AND xp≠yp
```

II.
```
Sibling(x,y) <- P(x,z) AND P(y,z)
Cousin(x,y) <- P(x,xp) AND P(xp,z) AND P(y,yp) AND P(yp,z)
               AND NOT Sibling(x,y)
```

III.
```
GrandParent(x,y) <- P(x,z) AND P(z,y)
Cousin(x,y) <- GrandParent(x,w) AND GrandParent(y,w)
```

(a) I only (b) I and II only (c) II only (d) I and III only.

---

The following 5 questions are based on a relation

`Emps(empID, ssNo, name, mgrID)`

giving for a set of employees their employee ID (assumed unique), their social-security number (also unique), the name of the employee (not necessarily unique), and the employee ID of the manager of the employee. Assume that the president is his/her own manager, so every employee has a unique manager. You may assume there are no duplicate tuples in this relation.

**Question 11:** Here are two possible ways to declare the relation `Emps`.

```
I.  CREATE TABLE Emps (
        empID INT,
        ssNo INT,
        name CHAR(50),
        mgrID INT,
        UNIQUE (empID),
        PRIMARY KEY (ssNo),
        FOREIGN KEY mgrID REFERENCES Emps(empID)
    );

II. CREATE TABLE Emps (
        empID INT PRIMARY KEY,
        ssNo INT UNIQUE,
        name CHAR(50),
        mgrID INT REFERENCES Emps(empID)
    );
```

Which, if any, of the two declarations above will correctly (in SQL2) declare the relation `Emps`?

(a) both I and II (b) I only (c) II only (d) neither I nor II.

**Question 12:** Suppose we wish to find the ID's of the employees that are managed by people who are managed by the employee with ID 123. Here are two possible queries:

```
I.  SELECT ee.empID
    FROM Emps ee, Emps ff
    WHERE ee.mgrID = ff.empID AND ff.mgrID = 123;

II. SELECT empID
    FROM Emps
    WHERE mgrID IN
        (SELECT empID FROM Emps WHERE mgrID = 123);
```

Which, if any, of the two queries above will correctly (in SQL2) get the desired set of employee ID's?

(a) both I and II (b) I only (c) II only (d) neither I nor II.

**Question 13:** Suppose we wish to find the ID's of all employees who do *not* manage any employee named "Sally." Here are two possible queries:

```
I.  SELECT mgrID
    FROM Emps
    WHERE NOT EXISTS(SELECT * FROM Emps WHERE name = 'Sally');

II. SELECT mgrID
    FROM Emps
    WHERE NOT(empID = ANY(SELECT empID FROM Emps WHERE name = 'Sally'));
```

Which, if any, of the two queries above will correctly (in SQL2) get the desired set of employee ID's?

(a) both I and II (b) I only (c) II only (d) neither I nor II.

**Question 14:** We wish to assert that no one can manage more than 10 employees. Here are two possible SQL2 assertions:

```
I.  CREATE ASSERTION I CHECK(NOT EXISTS(
        SELECT mgrID
        FROM Emps
        GROUP BY mgrID
        HAVING COUNT(*) > 10
    ));

II. CREATE VIEW mgrCounts AS
        SELECT mgrID, COUNT(*) AS cnt FROM Emps GROUP BY mgrID;

    CREATE ASSERTION II CHECK(10 >=
        (SELECT MAX(cnt) FROM mgrCounts);
```

Which of the assertions (plus view in case II) will enforce the desired requirement?

(a) both I and II (b) I only (c) II only (d) neither I nor II.

**Question 15:** Now, we would like to write an SQL3 recursion to find Joe's management chain, i.e., the ID's of his manager, his manager's manager, and so on. Joe's ID is 999. Here are two possible queries:

```
I.  WITH RECURSIVE Chain(x,y) AS
            (SELECT empID, mgrID FROM Emps)
        UNION
            (SELECT empID, y FROM Emps, Chain WHERE mgrID = x)
    SELECT y FROM CHAIN WHERE x = 999;
```

```
II. WITH RECURSIVE Chain(x,y) AS
            Emps
        UNION
            (SELECT x, mgrID FROM Chain, Emps WHERE y = empID)
    SELECT y FROM CHAIN WHERE x = 999;
```

Which, if any, of the two queries above will correctly (in SQL3) get the desired employee ID's?

(a) both I and II (b) I only (c) II only (d) neither I nor II.

---

**Question 16:** If $\cup$, $\cap$, and $-$ are given their *bag* interpretations, which of the following laws holds?

(a) $R \cup R = R$
(b) $R \cap (S \cup T) = (R \cap S) \cup (R \cap T)$
(c) $R \cup (S - T) = (R \cup S) - T$
(d) none of the above

---

**Question 17:** Suppose relation $R(A, B)$ currently has tuples $\{(1,2),\ (1,2),\ (3,4)\}$ and relation $S(B, C)$ currently has $\{(2,5),\ (2,5),\ (4,6),\ (7,8)\}$. Then the number of tuples in the result of the SQL2 query

```
SELECT *
FROM R NATURAL OUTER JOIN S;
```

is:

(a) 2 (b) 3 (c) 5 (d) 6.

The following 3 questions are based on the ODL description below, which represents sports teams and the players who play for them. In OQL queries based on this data, do not consider an answer incorrect based solely on the distinction between a thing and a structure that has that thing as the value of its lone field (i.e., assume `x` and `Struct(f:x)` are the same).

```
interface Team (extent Teams key name) {
    attribute string name;
    relationship Set<Player> players inverse Player::playsFor;
};

interface Player (extent Players key name) {
    attribute string name;
    relationship Team playsFor inverse Team::players;
};
```

**Question 18:** We wish to find all the players who play for the Giants. Here are two possible queries:

```
I.  SELECT t.players.name
    FROM Teams t
    WHERE t.name = "Giants";

II. SELECT p.name
    FROM Players p
    WHERE p.playsFor.name = "Giants";
```

Which, if any, of the two queries above will correctly find the players on the Giants?
(a) both I and II (b) I only (c) II only (d) neither I nor II.

**Question 19:** We wish to find the names of all the players who play on the same team as "Sue." Here are two possible queries:

```
I.  SELECT q.name
    FROM Players p, p.playsFor t, t.players q
    WHERE p.name = "Sue";

II. SELECT q.name
    FROM Teams t, t.players q
    WHERE EXISTS p IN t.players : p.name = "Sue";
```

Which, if any, of the two queries above will correctly find Sue and her teammates?
(a) both I and II (b) I only (c) II only (d) neither I nor II.

**Question 20:** If we convert the ODL description on the previous page to relations using the standard construction from the text, we get:

```
Teams(name)
Players(name, teamName)
```

We might decide to eliminate the `Teams` relation, because its schema is a subset of the `Players` schema. One reason we might *not* eliminate `Teams` is:

(a) In `Players`, the attribute `name` is not really the same as `name` in `Teams`.
(b) Some players might not be assigned to any team.
(c) There is a functional dependency from player to team.
(d) There might be teams currently without players.

---

**Question 21:** Suppose relation $R(A, B, C, D, E)$ has the following functional dependencies: $A \to B$, $B \to C$, $BC \to A$, $A \to D$, $E \to A$, and $D \to E$. Which of the following is *not* a key?

(a) $A$ (b) $E$ (c) $BC$ (d) $D$

---

**Question 22:** Consider the following SQL2 query on a relation $R(A, B)$ that has no `NULL`'s:

```
SELECT rr.A, rr.B, ss.A, ss.B
FROM R AS rr, R AS ss
WHERE rr.A = ss.A and rr.B = ss.B;
```

Suppose that $R$ has $n$ tuples (not necessarily all distinct). Which of the above conditions is the most restrictive, correct limitation on $m$, the number of tuples (again not necessarily distinct) in the result?

(a) $n \leq m \leq n^2$ (b) $n \leq m \leq 2n$ (c) $0 \leq m \leq n$ (d) $m = n$.

---

**Question 23:** Suppose now that $R(A, B)$ and $S(A, B)$ are two relations with $r$ and $s$ tuples, respectively (again, not necessarily distinct). If $m$ is the number of (not necessarily distinct) tuples in the result of the SQL2 query:

```
R INTERSECT S;
```

then which of the following is the most restrictive, correct condition on the value of $m$?

(a) $m = \min(r, s)$
(b) $0 \leq m \leq r + s$
(c) $\min(r, s) \leq m \leq \max(r, s)$
(d) $0 \leq m \leq \min(r, s)$.

**Question 24:** Suppose we have a relation `Emps(empID, dept, salary)` on which there is the following trigger. *Note*: we shall use the familiar Oracle syntax for triggers, although in fact, the following would not compile as an Oracle trigger because the action affects the triggering relation `Emps`. You should think of the semantics of the trigger as in SQL3.

```
CREATE OR REPLACE TRIGGER Trigger1
AFTER DELETION ON Emps
FOR EACH ROW
WHEN 50000 < (SELECT AVG(salary) FROM Emps)
    BEGIN
        UPDATE Emps
        SET salary = salary + 1000
        WHERE dept = 'MATH';
    END;
```

Suppose further that the current relation `Emps` is:

| empID | dept | salary |
|---|---|---|
| 111 | CS | 65000 |
| 222 | EE | 60000 |
| 333 | EE | 62000 |
| 444 | MATH | 52000 |

Finally, suppose we issue the following SQL command:

```
DELETE FROM Emps WHERE dept = 'EE';
```

Then after this command and any triggered actions, the average salary of the remaining employees is:

(a) 59,000 (b) 59,500 (c) 59,750 (d) 60,250.

---

**Question 25:** Which of the following is a safe Datalog rule?

(a) `P(x,y) <- Q(x,z) AND NOT R(y,z)`
(b) `P(x,y) <- Q(x,z) AND R(z,y) AND x<z`
(c) `P(x,y) <- Q(x,z) AND x<y AND NOT R(y,z)`
(d) None of the above (i.e., none is safe).

**Question 26:** Suppose we have the following database schema: $P(A, B)$ and $Q(B, C)$. Then the Datalog rule

```
Ans(x,y) <- P(x,y) AND Q(y,z)
```

produces the same relation instance as which of the following expressions of relational algebra?

(a) $\pi_{AB}(P \bowtie Q)$

(b) $\rho_{Ans(x,y)}(P \underset{P.B=Q.B}{\bowtie} Q)$

(c) $\pi_{AB}\big(P \bowtie \sigma_{C=z}(Q)\big)$

(d) There is no such relational-algebra expression, because the rule is not safe.

---

**Question 27:** Suppose we have the following database schema: $P(A, B)$, $Q(A, C)$, and $R(C, B)$. Then the relational algebra expression $P \cup \big(\pi_{AB}(Q \bowtie R)\big)$ produces the same value as the relation `Ans` in which of the following Datalog programs?

```
I.   T(x,y) <- Q(x,z) AND R(z,y)
     Ans(x,y) <- P(x,y) AND T(x,y)

II.  T(x,y) <- Q(x,z) AND R(z,y)
     Ans(x,y) <- P(x,y)
     Ans(x,y) <- T(x,y)

III. Ans(x,y) <- Q(x,z) AND R(z,y)
     Ans(x,y) <- P(x,y)
```

(a) I only (b) II and III only (c) III only (d) I, II, and III.

---

**Question 28:** Suppose we have a relation $R(A, B, C, D, E)$ with functional dependencies $A \to C$, $C \to E$, and $BC \to D$. Which of the following decompositions of $R$ has a lossless join (i.e., if any relation instance $R$ that satisfies the functional dependencies is projected onto the schemas of the decomposition, then the join of the projected relations is exactly the instance of $R$ you started with)?

(a) $ABC$, $CD$, $DE$
(b) $AC$, $CDE$, $AE$
(c) $AC$, $CE$, $BCD$
(d) $AB$, $BCD$, $ACE$

The next 2 questions are based on the following relation:

```
Family(parent, child, childDOB)
```

The intent is that a tuple $(p, c, d)$ means that parent $p$ has child $c$, who was born on date $d$. You may assume that parents do not have two children of the same name, and that there are no twins; i.e., no parent has two or more children born on the same day. Here are three queries we might ask about this data:

I. Find for each parent, the youngest child, i.e., the set of $(p, c)$ such that $p$ has child $c$, and no other child of $p$ has a smaller date of birth than $c$ does.

II. Find the set of great grandparents of "Amy."

III. Find all the descendants of "Mike."

**Question 29:** Which of the above queries are expressible in Datalog?
(a) II only (b) III only (c) II and III only (d) I, II, and III.

**Question 30:** Which of the above queries can be expressed in relational algebra?
(a) I only (b) I and II only (c) III only (d) I, II, and III.

---

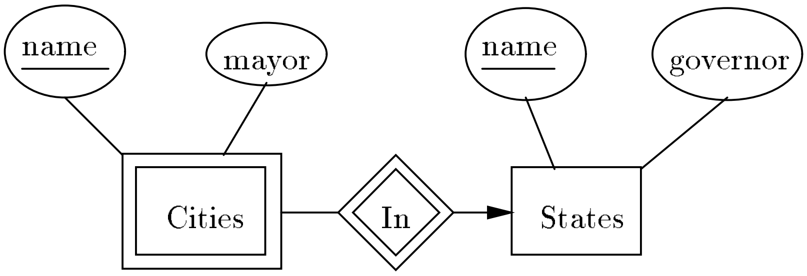
**Question 31:** Consider the following relation instance:

| $A$ | $B$ | $C$ |
|---|---|---|
| $a_1$ | $b_2$ | $c_1$ |
| $a_1$ | $b_2$ | $c_2$ |
| $a_1$ | $b_2$ | $c_3$ |
| $a_2$ | $b_3$ | $c_1$ |

The above relation instance (with different symbols representing definitely different values) illustrates the *invalidity* of which of the following rules about functional dependencies?

(a) If $AB \to C$, then $A \to C$
(b) If $AC \to B$, then $A \to B$
(c) If $BC \to A$, then $C \to A$
(d) none of the above.

The next two questions are based on the following E/R diagram representing cities and the states they are in.




A city's name is unique within a state, but there are often cities with the same name in different states. Thus, *Cities* is a weak entity set taking its key from its own name and the name of the state it is in.

**Question 32:** If we follow the rules given in the text for converting this E/R diagram to relations, we get:

(a) `Cities(name, mayor, stateName)` and `States(name, governor)`
(b) `Cities(name, mayor, governor)` and `States(name, governor)`
(c) `Cities(name, mayor, stateName, governor)`
(d) `Cities(name, mayor)` and `States(name, governor)`

**Question 33:** If we convert this E/R diagram to an ODL description with classes `City` and `State`, how should we best handle the relationship `In`?

(a) `In` would be represented by a relationship of type `State` in `City` and a relationship of type `Set<City>` in State.
(b) `In` would be represented by a third class, with many-one relationships between it and `City` and between it and `State`.
(c) `In` would not be represented, because there is no need for keys in ODL.
(d) `In` would be represented as a `state` attribute of class `City`.