CS2102: Database Systems (AY2022-2023 – Sem 1)

Final Exam

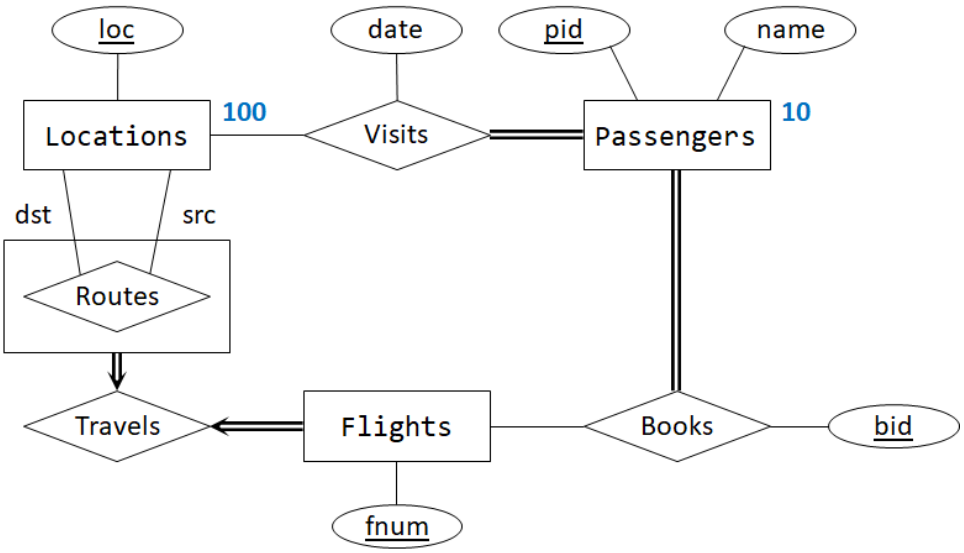
Instructions

1. Please read **ALL** instructions carefully.
2. This assessment contains **TWENTY TWO (22)** questions:
   (a) There are 5 Multiple Response Questions (MRQ): 1.1 - 1.2, 2.1 - 2.3
   (b) There are 6 Fill in the Blank Questions (FITB): 1.3, 2.4, 4.1 - 4.4
   (c) There is 10 Multiple Choice Question (MCQ): 3.1 - 3.10
   (d) There is 1 Short Essay Question: 3.11
3. All the assessment is be done using Examplify:
   (a) This is a secure assessment:
      i. Your Internet connection will be blocked.
      ii. You will not be able to access any other software besides Examplify.
   (b) This is a closed-book exam
   (c) The choices on Examplify may be in a different order
4. No additional time will be given to submit.
5. Use the question number shown on Examplify when asking question.
   - If the answer is clear from the question pdf/Examplify, we will reply with "No Comment".
6. Failure to follow each of the instructions above may result in deduction of your marks.

Good Luck!

1 Design

Consider the following ER diagram:



The number beside `Locations` and `Passengers` are the number of entries in the respective entity sets.

1.1 Constraints (4 marks). Select ALL constraints that are enforced by the ER diagram. Consider each constraint separately from other constraints.

(A) There can be no two bookings with the same `bid` for the same flight but for different passengers

(B) There can be no two bookings with the same `bid` for the same passenger but for different flights

(C) All flights must travel from an origin `src` to a destination `dst` but they may be the same location

(D) There can be no two travels with the same flight number `fnum` from the same origin `src` but to a different destination `dst`

(E) There can be no two travels with the same flight number `fnum` to the same destination `dst` but from a different origin `src`

(F) There can be two travels with the same `src` and `dst` but different `fnum`

(G) A passenger can only be recorded to visit the same location at most once

(H) A location can be recorded to be visited multiple times but only by different passengers

(I) *None of the above*

1.2 Schema (4 marks). Using the *fewest* number of relations, we want to translate the ER diagram to a schema. This translation should enforce as many *functional dependency* constraints in the ER diagram as possible without adding additional constraints. **Only primary key constraints can be used to enforce functional dependency.** Assume that foreign key constraints are automatically and *correctly* added.

We add foreign key constraints when the name of the attribute matches the referencing relation's attributes (*e.g, we add foreign key from* `pid` *of* `Visits` *to* `pid` *of* `Passengers`) or matches the label on the line connecting the entity set to the relationship set (*e.g., we add foreign key from attribute* `src` *to attribute* `loc`).

The following relations are already created:

- `Passengers(pid, name)`
- `Locations(loc)`
- `Visits(pid, loc)`

Select ALL the remaining relations in the resulting schema.

(A) `Flights(fnum)`

(B) `Flights(fnum, src, dst)`

(C) `Flights(fnum, src, dst)`

(D) `Routes(fnum, src, dst)`

(E) `Routes(fnum, src, dst)`

(F) `Books(bid, pid, fnum)`

(G) `Books(bid, pid, fnum)`

(H) `Books(bid, pid, fnum)`

1.3 Cardinality (4 marks). If there are 100 entries in `Locations` and 10 entries in `Passengers`, deduce the minimum and maximum number of entries on `Visits`, `Routes`, `Flights`, and `Books`.

Your answer ***MUST*** consists only of digits with a single exception: if the maximum is *unbounded*, then your answer should be -1. Any other answers will be marked as **wrong**. If the answer is nine thousand and nine hundred, you should write 9900. Any of the following will not be accepted: nine thousand and nine hundred, 9,000, 9000 + 900, 99 * 100, 990 × 10, *etc.*

(A) `Visits` : MIN = ☐ MAX = ☐

(B) `Routes` : MIN = ☐ MAX = ☐

(C) `Flights`: MIN = ☐ MAX = ☐

(D) `Books` : MIN = ☐ MAX = ☐

2 Routines

We use the following simple translation of the ER diagram from previous section. Note that this schema does not reflect the ER diagram from previous section and models some other constraints. You should treat each question separately.

```
CREATE TABLE Booking (
  pid  INT,
  bid  INT,
  src  VARCHAR(50),
  dst  VARCHAR(50),
  PRIMARY KEY (pid, bid, src, dst)
);
CREATE TABLE Visits (
  pid  INT NOT NULL,
  loc  VARCHAR(50) NOT NULL,
  PRIMARY KEY (pid, loc)
);
```

2.1 Function (4 marks) We say that a passenger has visited a certain location if the passenger `pid` has made a booking from (*i.e.,* `src`) *or* to (*i.e.,* `dst`) the given location. We want to create a procedure to find all ***distinct*** passenger IDs (*i.e.,* `pid`) that have visited both location A (*i.e.,* `locA`) and location B (*i.e.,* `locB`) using:

```
CREATE OR REPLACE FUNCTION hasVisited(
  locA VARCHAR(50), locB VARCHAR(50)
) RETURNS TABLE (pid INT) AS $$

  /* Implementation here ... */

$$ LANGUAGE sql;
```

Select ALL correct implementations.

You may assume that the tables satisfies all the table/column constraints. No other assumptions can be made on the table.

(A)
```
SELECT DISTINCT pid FROM Booking
WHERE  (src = locA AND dst = locB)
   OR  (src = locB AND dst = locA);
```

(B)
```
SELECT DISTINCT pid FROM Booking
WHERE  (src = locA OR src = locB)
  AND  (dst = locA OR dst = locB);
```

(C)
```
(SELECT pid FROM Booking WHERE src = locA)
UNION
(SELECT pid FROM Booking WHERE dst  = locB);
```

(D)
```
(SELECT pid FROM Booking WHERE src = locA)
INTERSECT
(SELECT pid FROM Booking WHERE dst = locB);
```

(E)
```
(SELECT DISTINCT pid FROM Booking
 WHERE  src = locA OR src = locB)
INTERSECT ALL
(SELECT DISTINCT pid FROM Booking
 WHERE  dst = locA OR dst = locB);
```

(F) *None of the above*

2.2 Procedure (4 marks) Recall that procedures are treated like a transaction. Consider the following procedure to insert return trips (*i.e., a trip comprising of two flights from a source to destination and the opposite flight back*):

```
CREATE OR REPLACE PROCEDURE returnTrip(
  id INT, book INT, source VARCHAR(50), dest VARCHAR(50)
) AS $$
BEGIN
  INSERT INTO Visits VALUES (id, source);
  INSERT INTO Visits VALUES (id, dest);
  INSERT INTO Booking VALUES (id, book, source, dest);
  INSERT INTO Booking VALUES (id, book, dest, source);
END;
$$ LANGUAGE plpgsql;
```

Select ALL statements that are **true** about the procedure. You may assume that no `NULL` values will be given as inputs to the procedures.

(A) The procedure may insert 2 new rows on `Booking`

(B) The procedure may insert 1 new row on `Booking`

(C) The procedure may insert 0 new row on `Booking`

(D) The procedure may insert 2 new rows on `Visits`

(E) The procedure may insert 1 new row on `Visits`

(F) The procedure may insert 0 new row on `Visits`

(G) The procedure may insert into `Booking` without inserting into `Visits`

(H) The procedure may insert into `Visits` without inserting into `Booking`

(I) *None of the above*

2.3 Trigger (4 marks) Consider the following trigger function and trigger *independently* of other questions:

```
CREATE OR REPLACE FUNCTION func1()
RETURNS TRIGGER AS $$
DECLARE
  cur CURSOR FOR ( SELECT dst FROM Booking
                   WHERE  pid = NEW.pid AND  bid = NEW.bid );
  rec RECORD;  loc VARCHAR(50);
BEGIN
  OPEN cur;  loc := NULL;
  LOOP
    FETCH cur INTO rec;
    EXIT WHEN NOT FOUND;
    loc := rec.dst;
    IF loc = NEW.src THEN
      CLOSE cur;
      RETURN NEW;
    END IF;
  END LOOP;
  CLOSE cur;
  IF loc IS NULL THEN
    RETURN NEW;
  ELSE
    RETURN NULL;
  END IF;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trig1
BEFORE INSERT ON Booking
FOR EACH ROW EXECUTE FUNCTION func1();
```

Starting from an ***empty*** database, select ALL sequence of insert statements such that both statements will insert new rows (*i.e., at the end, two rows are inserted*). Each choice starts from an ***empty*** database.

(A)
```
INSERT INTO Booking VALUES (1, 10, 'A', 'B');
INSERT INTO Booking VALUES (2, 10, 'A', 'B');
```

(B)
```
INSERT INTO Booking VALUES (1, 10, 'A', 'B');
INSERT INTO Booking VALUES (1, 10, 'A', 'B');
```

(C)
```
INSERT INTO Booking VALUES (1, 10, 'A', 'B');
INSERT INTO Booking VALUES (1, 10, 'C', 'D');
```

(D)
```
INSERT INTO Booking VALUES (1, 10, 'A', 'B');
INSERT INTO Booking VALUES (1, 10, 'B', 'A');
```

(E) *None of the above*

2.4 Trigger (6 marks) We want to create an after statement-level trigger such that every insertion to `Booking` will also insert to `Visits` if the location is not already recorded. *The trigger should not raise any error.* Fill in the blanks below to complete the trigger. Each blank should be filled with a single word keyword, operation, attribute, or table name. There should **NOT** be any nested queries inside the blanks.

```
CREATE OR REPLACE FUNCTION func2()
RETURNS TRIGGER AS $$
DECLARE
  cur CURSOR FOR (
    (SELECT pid, src FROM Booking)
    [          ]
    (SELECT pid, [          ] FROM [          ])
    [          ]
    (SELECT pid, [          ] FROM [          ])
  );
  rec RECORD;
BEGIN
  OPEN cur;
  LOOP
    FETCH cur INTO rec;
    EXIT WHEN NOT FOUND;
    INSERT INTO Visits VALUES (rec.pid, rec.src);
  END LOOP;
  CLOSE cur;
  RETURN NULL;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trig2
AFTER INSERT ON Booking
FOR EACH STATEMENT EXECUTE FUNCTION func2();
```

3 Normal Forms

Consider the relation $R(A, B, C, D)$ with the set of functional dependencies:

$$\Sigma = \{\{A, B\} \to \{C\}, \{B, C\} \to \{C, D\}, \{C, D\} \to \{C, D\}, \{D\} \to \{A\}\}$$

3.1 Trivial (2 marks) Which of the following dependencies in Σ is trivial?

(A) $\{B, C\} \to \{C, D\}$

(B) $\{C, D\} \to \{C, D\}$

(C) $\{C, D\} \to \{C\}$

(D) *All of the above*

(E) *None of the above*

3.2 Attribute Closure (2 marks) Which of the following sets of attributes is the attribute closure of $\{B, D\}$ in R with Σ?

(A) $\{B, D\}$

(B) $\{A, B, D\}$

(C) $\{A, B, C, D\}$

(D) *All of the above*

(E) *None of the above*

3.3 Functional Dependency Closure (2 marks) Which of the following functional dependencies is in Σ^+?

(A) $\{A\} \to \{C, D\}$

(B) $\{C, D\} \to \{A\}$

(C) $\{A, D\} \to \{C\}$

(D) *All of the above*

(E) *None of the above*

3.4 Non Trivial (2 marks) Which of the following functional dependencies in Σ^+ is non trivial but not completely non trivial?

(A) $\{B, C\} \to \{C, D\}$

(B) $\{C, D\} \to \{C, D\}$

(C) $\{C, D\} \to \{C\}$

(D) *All of the above*

(E) *None of the above*

3.5 Completely Non Trivial (2 marks) Which of the following functional dependencies in Σ^+ is completely non trivial?

(A) $\{B, C\} \to \{A\}$

(B) $\{C, D\} \to \{C, D\}$

(C) $\{A, B, C\} \to \{C, D\}$

(D) *All of the above*

(E) *None of the above*

3.6 Superkey (2 marks) Which of the following sets of attributes is a superkey of R with Σ?

(A) $\{A, C\}$

(B) $\{A, D\}$

(C) $\{A, C, D\}$

(D) *All of the above*

(E) *None of the above*

3.7 Candidate Key (2 marks) Which of the following sets of attributes is a candidate key of R with Σ?

(A) $\{A, B\}$

(B) $\{B, C\}$

(C) $\{B, D\}$

(D) *All of the above*

(E) *None of the above*

3.8 Minimal Cover (2 marks) Which of the following sets of functional dependencies is a minimal cover Σ?

(A) $\Sigma_1 = \{\{A, B\} \to \{C\}, \{B, C\} \to \{C\}, \{D\} \to \{A\}\}$

(B) $\Sigma_1 = \{\{A, B\} \to \{C\}, \{B, C\} \to \{D\}, \{D\} \to \{A\}\}$

(C) $\Sigma_1 = \{\{B, D\} \to \{C\}, \{B, C\} \to \{A\}, \{D\} \to \{A\}\}$

(D) *All of the above*

(E) *None of the above*

3.9 3NF (2 marks) Which of the following decomposition of R is a lossless dependency-preserving decomposition of R with Σ in third normal form?

(A) $\{R_1(A, D), R_2(B, C, D)\}$

(B) $\{R_1(B, C, D), R_2(A, D), R_3(A, C)\}$

(C) $\{R_1(A, B, C), R_2(B, C, D), R_3(A, D)\}$

(D) *All of the above*

(E) *None of the above*

3.10 BCNF (2 marks) Which of the following decomposition of R is a lossless decomposition of R with Σ in Boyce-Codd normal form?

(A) $\{R_1(A, D), R_2(B, C, D)\}$

(B) $\{R_1(B, C, D), R_2(A, D), R_3(A, C)\}$

(C) $\{R_1(A, B, C), R_2(B, C, D), R_3(A, D)\}$

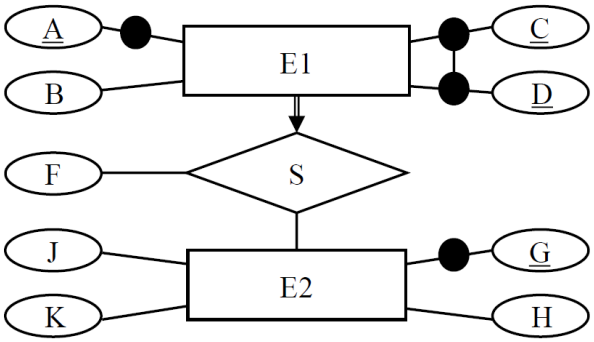
(D) *All of the above*

(E) *None of the above*

3.11 Armstrong Axioms (8 marks) Prove that $\{A, B\}$ is a superkey of R with Σ using the three Armstrong axioms only.

4 ER to Refinement

Consider the ER diagram below:



Consider that we translate the ER diagram above into a single table:

$$R(A, B, C, D, F, G, H, J, K)$$

Note that in this ER diagram representation, black dots represent prime attributes. A collection of black dots *connected* to other black dots represents a single candidate key. As such, there can be multiple candidate keys represented for a single entity set in this ER diagram representation.

4.1 Compact Cover (6 marks) Find a compact cover of the set of functional dependencies, Σ, that exist according to the ER diagram above. Fill in the set of attributes for each blank below separated by a comma (*i.e.,* **,**). If there are fewer than 6 functional dependencies, you may leave the spot blank.

The compact cover, $\Sigma' = \{$

$\{$ ☐ $\} \rightarrow \{$ ☐ $\}$ $\{$ ☐ $\} \rightarrow \{$ ☐ $\}$

$\{$ ☐ $\} \rightarrow \{$ ☐ $\}$ $\{$ ☐ $\} \rightarrow \{$ ☐ $\}$

$\{$ ☐ $\} \rightarrow \{$ ☐ $\}$ $\{$ ☐ $\} \rightarrow \{$ ☐ $\}$

$\}$

4.2 Candidate Keys (4 marks) Find all the candidate keys of R with Σ. Fill in the set of attributes for each blank below separated by a comma (*i.e.,* **,**). If there are fewer than 4 candidate keys, you may leave the spot blank.

The keys are $\{$ $\{$ ☐ $\}$, $\{$ ☐ $\}$, $\{$ ☐ $\}$, $\{$ ☐ $\}$ $\}$

4.3 Violation (4 marks) Argue that this table R with the set of functional dependencies Σ is neither in Boyce-Codd normal form nor in third normal form. Find a functional dependency that violates both Boyce-Codd normal and third normal form of R with Σ. Fill in the set of attributes for each blank below separated by a comma (*i.e.,* **,**).

$\{$ ☐ $\} \rightarrow \{$ ☐ $\}$

4.4 3NF (8 marks) Which new functional dependencies (*only one*) can be added to Σ such that the new set of functional dependencies is a compact cover and R with the new set of functional dependencies is in third normal form but not in Boyce-Codd normal form. Fill in the set of attributes for each blank below separated by a comma (*i.e.,* **,**).

$\{$ $\square$ $\}$ $\rightarrow$ $\{$ $\square$ $\}$