McMaster University
CS 3DB3, SE 4DB3/6DB3 Winter 2015
Assignment 3 Solution Sketch

April 12, 2015

I. Database Design (55 marks)

Question 1 (12 marks)

Consider a relation schema R with attributes $\mathcal{U} = \{A, B, C, D, E\}$, and the set of functional dependencies

$$\begin{aligned}\mathcal{F} = \{&A \rightarrow BC\\ &CD \rightarrow E\\ &B \rightarrow D\\ &E \rightarrow A\}\end{aligned}$$

Find all candidate keys (i.e., minimal keys) of relation R. Show all the steps you took to derive each key, and clearly state which of Armstrong's axioms are used in each step.

Answer:

Show that each key leads to the entire relation. Keys are: A, BC, CD, and E.

Sample steps to derive key E:

$E \rightarrow A$ and $A \rightarrow BC$ (Given)
$E \rightarrow BC$ (Transitivity)
$B \rightarrow D$ (Given)
$BC \rightarrow CD$ (Augmentation with C)
$E \rightarrow CD$ (Transitivity)
$E \rightarrow ABCD$ (Union)
$E \rightarrow ABCDE$ (Augmentation with D)

Question 2 (18 marks)

Consider the relation R(A,B,C,D,E,F), and the decomposition of R into R1(ABCD) and R2(DEF).

a) (4 marks) Give a set of functional dependencies (FDs) such that the decomposition into R1 and R2 is lossless join and dependency preserving. Justify why your FDs satisfy the criteria.

Sample solution:
Functional dependencies $A \rightarrow BCD, D \rightarrow A, DF \rightarrow E$.

b) (4 marks) Give a set of functional dependencies such that the decomposition into R1 and R2 is lossless join, but not dependency preserving. Justify why your FDs satisfy the criteria.

Sample solution:
Functional dependencies $A \rightarrow BCD, D \rightarrow A, C \rightarrow E$.

c) (4 marks) Give a set of functional dependencies such that the decomposition into R1 and R2 is not lossless join, but dependency preserving. Justify why your FDs satisfy the criteria.

Sample solution:
Functional dependencies $A \rightarrow BCD, C \rightarrow A, F \rightarrow ED$.

d) (6 marks) Consider a relation S(A,B,C,D,E,F), and functional dependency $F_1 : A \rightarrow BCDEF$, that holds over S. Define two other functional dependencies F_2 and F_3, which satisfy the following three properties:

i. Neither F_2 nor F_3 can be inferred from F_1 using Armstrong's axioms.

ii. Relation S with functional dependencies F_1 and F_2 is in BCNF.

iii. Relation S with functional dependencies F_1, F_2, and F_3 is in 3NF but not in BCNF.

State your two FDs F_2 and F_3, and justify how your FDs satisfy the above properties.

Sample solution:
Functional dependencies $F_2 : BD \rightarrow A$ and $F_3 : C \rightarrow B$.
Note: F_2 must have superkey LHS; F_3 must have prime RHS and can't have superkey LHS.

Question 3 (10 marks)

a) (5 marks) Consider the schema R(A,B,C,D,E,F), and the following functional dependencies: $A \rightarrow BCD, BC \rightarrow DE, B \rightarrow D, D \rightarrow A$.

Using Armstrong's Axioms (using only the axioms presented in class), show that AF is a superkey. Show the steps of your proof, and indicate which of Armstrong's axioms is applied in each step.

Answer:
$A \rightarrow BCD$ (Given)
$A \rightarrow ABCD$ (reflexive)
$BC \rightarrow DE$ (Given)
$ABCD \rightarrow ABCDE$ (Augmentation with ABCD)
$A \rightarrow ABCDE$ (Transitivity)
$AF \rightarrow ABCDEF$ (Augmentation with F)

b) (5 marks) Given the relational schema $T(A, B, C, D)$ and the FDs: $\{ABC \rightarrow D$, $CD \rightarrow A$, and $CA \rightarrow B$, $AD \rightarrow C$, $CD \rightarrow B\}$. Compute the minimal basis for T's functional dependencies.

Answer:

1^{st} Step:

Done since all FDs have signletons RHS.
Step outcome $\Rightarrow H = \{ABC \rightarrow D, CD \rightarrow A, CA \rightarrow B, AD \rightarrow C, CD \rightarrow B\}$.

2^{nd} Step:

$J = H - \{ABC \rightarrow D\} = \{CD \rightarrow A, CA \rightarrow B, AD \rightarrow C, CD \rightarrow B\}$
$ABC \rightarrow D$ can't be removed, $ABC^+ = \{ABC\}$.

$J = H - \{CD \rightarrow A\} = \{ABC \rightarrow D, CA \rightarrow B, AD \rightarrow C, CD \rightarrow B\}$
$CD \rightarrow A$ can't be removed, $CD^+ = \{BCD\}$.

$J = H - \{CA \rightarrow B\} = \{ABC \rightarrow D, CD \rightarrow A, AD \rightarrow C, CD \rightarrow B\}$
$CA \rightarrow B$ can't be removed, $CA^+ = \{CA\}$.

$J = H - \{AD \rightarrow C\} = \{ABC \rightarrow D, CD \rightarrow A, CA \rightarrow B, CD \rightarrow B\}$
$AD \rightarrow C$ can't be removed, $AD^+ = \{AD\}$.

$J = H - \{CD \rightarrow B\} = \{ABC \rightarrow D, CD \rightarrow A, CA \rightarrow B, AD \rightarrow C\}$
$CD \rightarrow B$ can be removed, $CD^+ = \{CDAB\}$.

Step outcome $\Rightarrow H = \{ABC \rightarrow D, CD \rightarrow A, CA \rightarrow B, AD \rightarrow C\}$.

3^{rd} Step:

$H = \{AC \rightarrow D, CD \rightarrow A, CA \rightarrow B, AD \rightarrow C\}$
$AC^+ = \{ACB \ldots\}$, since B can be derived, it is redundant.

$H = \{A \rightarrow D, CD \rightarrow A, CA \rightarrow B, AD \rightarrow C\}$
$A^+ = \{ADC \ldots\}$, since C can be derived, it is redundant.

$H = \{A \rightarrow D, D \rightarrow A, CA \rightarrow B, AD \rightarrow C\}$
$D^+ = \{DAC \ldots\}$, since C can be derived, it is redundant.

$H = \{A \rightarrow D, D \rightarrow A, A \rightarrow B, AD \rightarrow C\}$
$A^+ = \{ADC \ldots\}$, since C can be derived, it is redundant.

$H = \{A \rightarrow D, D \rightarrow A, A \rightarrow B, A \rightarrow C\}$
$A^+ = \{AD \ldots\}$, since D can be derived, it is redundant.

Step outcome $\Rightarrow H = \{A \rightarrow D, D \rightarrow A, A \rightarrow B, A \rightarrow C\}$.

4^{th} Step:

H doesn't change.

Minimal cover:

$A \rightarrow BCD$
$D \rightarrow A$

Note: The minimal basis is not unique.

Question 4 (15 marks)

The University has recruited you to design a better schema for its current `Timetable` relation, as given below.

- `Timetable (course, day, time, room, professor, ta)`
  These represent, respectively, the course code, the day the course is offered, the time of the course offering, the room course is held in, the professor name, and the lead TA name.

You must represent the following additional information:

a) Two professors cannot share the same room/day/time combination.

b) Two courses cannot share the same room/day/time combination.

c) There is at most one lead TA per course.

d) No two rooms have the same professor/day/time combination.

e) Two distinct courses cannot have the same professor teaching at the same day and time.

f) A course/day/time combination uniquely determines the professor.

g) No two rooms can have the same course, day, and time assignment.

Answer the following questions:

1. What are the functional dependencies that can be inferred from the additional information? List all of them.
2. Your client (the University) will be satisfied if your design is a good one, i.e., the schema satisfies either the BCNF or the 3NF.
   Is the design of your schema with the functional dependencies from part (1) above a good one? Justify your answer. If the design is not a good one, provide a better one, using one of the decomposition algorithms discussed in class.

Solution:

Part 1)

a) $RDT \rightarrow P$

b) $RDT \rightarrow C$

c) $C \rightarrow A$

d) $PDT \rightarrow R$

e) $PDT \rightarrow C$

f) $CDT \rightarrow P$

g) $CDT \rightarrow R$

Part 2) Keys are: RDT, PDT, CDT. Timetable() is not in BCNF. $C \rightarrow A$ violates BCNF.
R1(A,C), R2(C,D,T,R,P) along with the projected FDs.

II. Transactions and Concurrency (22 marks)

Question 5 (8 marks)

a) (4 marks) Give an example of a transaction schedule that is conflict-serializable, but which is not possible using 2PL. In other words, the schedule should be conflict-serializable, but there should be no way to acquire and release read/write locks in a way that is consistent with the 2PL protocol.

Answer: Example: W1(A),R2(A),R1(A);this is conflict-serializable, because we can exchange R2(A),R1(A) without any effect to the final result. However, it's not 2PL, because W1(A) has an exclusive lock on A, if R2(A) wants to read A, T1 should release the exclusive lock. R1(A) then requests to read A, and T1 requests to lock A again, which violates the 2PL principle: A transaction cannot request additional locks once it releases any lock.

b) (4 marks) Is every schedule that is allowed under the Strict 2PL locking protocol also allowed under the 2PL locking protocol? Or vice versa? Give an example schedule that is allowed in one of the two locking protocols, but not in the other locking protocol.

Answer: Yes, every schedule that complies with strict 2PL always complies with 2PL.
No, a schedule that complies with 2PL doesn't guarantee it complies with strict 2PL. Example: X means exclusive lock, U means release lock;

X1(A),X1(B),W1(A),U1(A),X2(A),W2(A),W1(B),U1(B),T1(commit);

Notice that before T1 commits, it already releases the exclusive lock on A, as U1(A); This example is 2PL, but not strict 2PL, since in strict 2PL, all locks held by a transaction are released when the transaction completes (aborts or commits).

Question 6 (14 marks)

In this question, you will execute a few concurrent transactions against a banking table. The `accounts` table is shown in Table 1. Create the `accounts` table in DB2. You may assume `username` is of type char(10), `name` is char(20), and `balance` is of type decimal(10,2). Write the appropriate INSERT INTO statements to load the four records shown into your `accounts` table.

Table 1: `accounts`

username	**name**	**balance**
ana	Anakin	1150.00
vader	Vader	7000.00
leia	Leia	100.00
obi	Obiwan	2275.00

Setup

You will simulate concurrent database queries, using two different database connections, against the `accounts` table. You will open two terminal sessions, and execute the following commands to prepare your environments. We will call the two sessions, A and B.

In Session A

1. Invoke db2 using `db2 +c`. This command turns off the AUTO COMMIT feature in DB2. You will now be at the DB2 command line.
2. Verify AUTO COMMIT is turned OFF by running the command `list command options`. (You should see "Auto-Commit OFF" in the second entry of the table.)
3. Change the transaction isolation level to 'Read Stability' by running the command `change isolation to rs`. Transaction isolation levels define the degree of access and interaction among a set of concurrent transactions which operate against the same data. Please see the DB2 reference for more information on isolation levels in DB2.
4. Connect to your database.

In Session B

1. Invoke DB2 by typing `db2` (do not turn AUTO COMMIT off).
2. Run `change isolation to rs`
3. Connect to your database.

Transactions

Run the following commands (in the given order) and provide your answers to the stated questions.

1. In Session A, insert the record `('yoda', 'Yoda', 8000.00)`, followed by a `select * from accounts`.
2. In Session B, run `select * from accounts`. Is the output you get the same or different than in (1)? Why did this occur? What is a possible solution?
3. Return to Session A, and implement your solution from the previous step.
4. In Session B, do the SELECT * query again (to list all records). Provide your output.
5. We will update accounts from two different transactions. In Session A, change the isolation level to cursor stability (CS). In Session B, change the isolation level to uncommitted read (UR). (You will have to disconnect and reconnect to the database.)
6. In Session A, transfer $800 from Anakin's account into Vader's account.
7. Then, in Session B, list all accounts and their balances. Then issue a $50 transfer from Leia's account to Vader's account. What happens and why?

8. Commit the transaction in Session A. What is Vader's balance now?

9. Change the transaction isolation level in Session A to 'Uncommitted Read'.

10. Now, in Session A, transfer 80% of Vader's account balance to Obiwan's account. Commit the transaction.

11. In Session A, transfer 50% of Yoda's funds to Obiwan's account. In session B, list all data records. What is Obiwan's balance? Does it reflect the latest transfer from Yoda? Based on your transaction in step (7) and this step, what can you say about the allowed actions in CS and UR isolation levels? How do these compare with the RS isolation level?

12. Abort the Yoda to Obiwan transfer transaction in Session A, by executing the command `rollback`.

13. In session A and B, list all data records. What are the final balances for each user?

Solution Highlights:

- Session in B hangs because Yoda's uncommitted record cannot be viewed until after a COMMIT from session A.
- In point 4, the SELECT * should now include all the original records + Yoda's newly inserted record.
- Only reads are allowed under the CS option in Session A, not writes.
- Uncommitted reads are allowed in CS and UR isolation levels. However, updates are not allowed in session B if there is a pending transaction from session A. It's different than RS because in RS we cannot view uncommitted reads, only committed data updates.