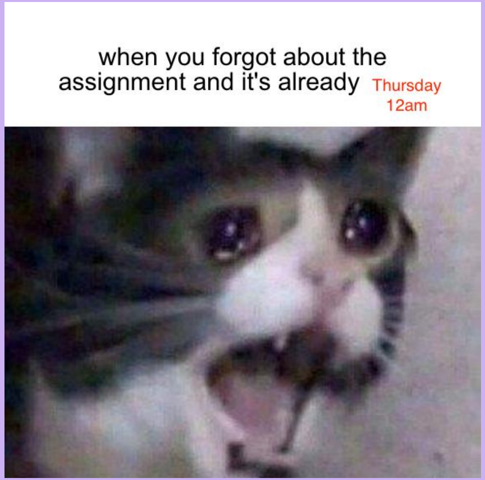
COMP3311 Week 9



By Manhua Lu
Adapted from Kyu-Sang Kim

Announcements

- NO quiz this week!! 🥳
- Assignment 2
    - Due Wednesday 15 November 2023 @11:59pm
    - There will be 4 help sessions being run
        - Mon 12-2, Tue 2-4, Thu 10-12, Fri 10-12 at K17 Ground Floor
- Exam session times out:
    - Monday 4th December
        - 10:15am - 1:30pm
        - 1:55pm - 5:10pm
    - Form to register preferences will be out soon. First-come-first-serve!

Learning Objectives

01 Functional Dependencies (Q1)

02 Closure (Q2)

03 Normalisation: (Q3)
- Boyce-Codd Normal Form (BCNF)
- Third Normal Form (3NF)

04 Minimal Cover (Q13)

05 Normalisation Decomposition (Q7)

01

Functional Dependencies

01 Functional Dependencies

- When we see something like X → Y
  - i.e. X determines Y (or Y is functionally dependent on X)
  - Every X value has exactly ONE corresponding Y value
- Example: Position → Salary
  - Position determines salary
  - For every unique position in our database, the salary for that given position is the same
  - Manager → $50
  - Intern → $30

01 Functional Dependencies

Armstrong's Rules (General Rules of Inferences on FDs):

F1. Reflexivity e.g. $X \rightarrow X$

- a formal statement of *trivial dependencies*; useful for derivations

F2. Augmentation e.g. $X \rightarrow Y \Rightarrow XZ \rightarrow YZ$

- if a dependency holds, then we can freely expand its left hand side

F3. Transitivity e.g. $X \rightarrow Y, Y \rightarrow Z \Rightarrow X \rightarrow Z$

- the "most powerful" inference rule; useful in multi-step derivations

---

Inference Rules (cont)

While Armstrong's rules are complete, other useful rules exist:

F4. Additivity e.g. $X \rightarrow Y, X \rightarrow Z \Rightarrow X \rightarrow YZ$

- useful for constructing new right hand sides of *fd*s (also called union)

F5. Projectivity e.g. $X \rightarrow YZ \Rightarrow X \rightarrow Y, X \rightarrow Z$

- useful for reducing right hand sides of *fd*s (also called decomposition)

F6. Pseudotransitivity e.g. $X \rightarrow Y, YZ \rightarrow W \Rightarrow XZ \rightarrow W$

- shorthand for a common transitivity derivation

01 Functional Dependencies: Candidate Key

- An attribute or combination of attributes that can be used to infer all the attributes in the entire schema using the given FDs.

A **super key** is also an attribute or combination that can be used to infer the entire schema. It **can** be reduced.

Candidate keys **cannot be** further reduced

- Thus they are minimal super keys

All super keys can't be candidate keys

But all candidate keys are super keys

01 Functional Dependencies: Candidate Key

- To find out whether something is a candidate key, compute a **closure** and see if all the attributes in the schema can be inferred.

e.g.

The candidate key of FD = { A -> B, BC -> D } is AC.

e.g.

R(A,B,C,D)

FD: {A->BCD}

Candidate Key: A

Super Keys: A, AB, AC, AD, ABC, ABCD

e.g.

X' subset of X where X' -> R and X -> R

-> X is not a candidate key

-> X is a super key

01 Functional Dependencies Question

1. Functional dependencies.

    a. What functional dependencies are implied if we know that a set of attributes *X* is a candidate key for a relation *R*?

    b. What functional dependencies can we infer *do not hold* by inspection of the following relation?

A	B	C
a	1	x
b	2	y
c	1	z
d	2	x
a	1	y
b	2	z

    c. Suppose that we have a relation schema *R(A,B,C)* representing a relationship between two entity sets *E* and *F* with keys *A* and *B* respectively, and suppose that *R* has (at least) the functional dependencies $A \rightarrow B$ and $B \rightarrow A$. Explain what this tells us about the relationship between *E* and *F*.

02

Closure

02 Closure

- Given a set of FDs, we can derive new ones
- The closure of F is the largest collection of dependencies that can be derived from a set F of FDs. It is denoted by F^+.
  - Generally you'll be asked to compute the closure for a given set of **attributes** as calculating closures on a set of fds becomes quickly infeasible.
- e.g. Given a set X of attributes and a set F of fds, the closure of X (denoted X^+) is the largest set of attributes that can be derived from X using F.

02 How to Compute a Closure

Given starting attributes and a set of functional dependencies

1. Add the given attributes to a new set
2. Use the FDs to figure out what attributes can be inferred with the attributes currently in the set. Add those inferred attributes to the set.
3. Keep doing this until we are unable to add anymore attributes to the set.

02 Closure Computing Algorithm

```
Input: F (set of FDs), X (starting attributes)
Output: X+ (attribute closure)

Closure = X
while (not done) {
   OldClosure = Closure
   for each A → B such that A ⊂ Closure
      add B to Closure
   if (Closure == OldClosure) done = true
}
```

02 Closure Question

2. Consider the relation $R(A,B,C,D,E,F,G)$ and the set of functional dependencies $F = \{ A \rightarrow B,\ BC \rightarrow F,\ BD \rightarrow EG,\ AD \rightarrow C,\ D \rightarrow F,\ BEG \rightarrow FA \}$ compute the following:

a. A^+

b. $ACEG^+$

c. BD^+

02 How to get Candidate Keys from FDs?

Candidate Keys: smallest subset of R such that their closure results in all attributes of R.

Given a relation R(A,B,C,D,E), we get the candidate keys by finding a set of attributes X such that:

- Their closure results in all attributes of R
- There is no subset of X whose closure is R

In essence, we take the FDs, and keep adding attributes until the RHS has all the attributes in the schema.

02 → Closure + Candidate Keys Question

3. Consider the relation $R(A,B,C,D,E)$ and the set set of functional dependencies $F = \{ A \rightarrow B,\ BC \rightarrow E,\ ED \rightarrow A \}$

$\boxed{\text{a. List all of the candidate keys for } R.}$

b. Is R in third normal form (3NF)?

c. Is R in Boyce-Codd normal form (BCNF)?

03

Normalisation

03 Normalisation

- Removing redundancies from your schema
- e.g.
  - Student(zID, Name, Surname, Address)
  - CourseEnrolments(zID, Name, Surname, Course, Course Name)
- Storing zID, Name, Surname twice is redundant.
  - Instead, make zID in course enrolments a foreign key toward Student
  - Student(zID, Name, Surname, Address)
  - CourseEnrolments(zID (fk to Student), Course, Course Name)

03 → Normal Forms

- Normal forms tell us the schema's level of redundancy.

First, Second, Third NF — allows most redundancy

Boyce-Codd NF

Fourth NF

Fifth NF — allows least redundancy

The forms that we care about in this course (industry acceptable NFs):

- BCNF and 3NF

03 Normal Forms: Detecting BCNF

A functional dependency **is in BCNF** if it passes **any** of these conditions:

1. Is RHS a subset of LHS

    e.g. ABC -> B

2. Is the LHS a superkey?

    i.e. if you used it to compute a closure, would the closure be the entire schema?

If all FDs are in BCNF, the schema is in BCNF and we can stop!

03 → Normal Forms: Detecting 3NF

A functional dependency **is in 3NF** if it passes **any** of these conditions:

1. Is RHS a subset of LHS

    e.g. ABC -> B

2. Is the LHS a superkey?

    i.e. if you used it to compute a closure, would the closure be the entire schema?

3. **Is the RHS only 1 attribute and part of a candidate key?**

If all FDs are in 3NF, the schema is in 3NF and we can stop!

03 Normal Forms Question

3. Consider the relation $R(A,B,C,D,E)$ and the set set of functional dependencies $F = \{ A \rightarrow B,\ BC \rightarrow E,\ ED \rightarrow A \}$

a. List all of the candidate keys for R.

b. Is R in third normal form (3NF)?

c. Is R in Boyce-Codd normal form (BCNF)?

04

Minimal Cover

04 Minimal Cover

- Smallest set of functional dependencies that covers the entire FD set
- Computing the Minimal Cover:
  1. Convert FDs into canonical form i.e. RHS has 1 attribute
  2. For LHS that are multi-attribute, remove any redundant attributes
  3. Eliminate redundant FDs that are implied by other ones (including ones made as canonical cover)

04 Minimal Cover Question

13. Compute a minimal cover for:

$$F = \{B \to A,\ D \to A,\ AB \to D\}$$

05

Normalisation Decomposition

05

Normalisation Decomposition: 3NF Decomposition

1. Compute the minimal cover and ‘flatten’ it into relations i.e. remove all arrows
   e.g. A→B, A→C, A→D becomes AB, AC, AD.
2. If the resulting set has no relation with the candidate key that you’ve chosen, then add a new relation with that candidate key to the set. This will act as a linking relation.

05 Normalisation Decomposition: 3NF Decomposition Question

7. For each of the sets of dependencies in question 4:

i. if R is not already in 3NF, decompose it into a set of 3NF relations

ii. if R is not already in BCNF, decompose it into a set of BCNF relations

a. $C \rightarrow D, \quad C \rightarrow A, \quad B \rightarrow C$

b. $B \rightarrow C, \quad D \rightarrow A$

c. $ABC \rightarrow D, \quad D \rightarrow A$

d. $A \rightarrow B, \quad BC \rightarrow D, \quad A \rightarrow C$

e. $AB \rightarrow C, \quad AB \rightarrow D, \quad C \rightarrow A, \quad D \rightarrow B$

f. $A \rightarrow BCD$

05 → Normalisation Decomposition: BCNF Decomposition

1. Get all FDs that pertain to the current schema
   i.e. the FD's letters are a subset of the schema.
   e.g. A→B pertains to ABC, but B→D does not pertain to ABC, as D is not part of the schema.
2. If all FDs are in BCNF, this schema is fine as is. Skip to step 5.
3. If any FD is not in BCNF, subtract the RHS from the schema, and add another schema that is the LHS + RHS.
   e.g. for schema ABCD, if B→D is not in BCNF, then replace this schema with (itself - RHS), (LHS + RHS).
   In this case, it becomes (ABCD - D), (B + D) = ABC, BD
4. Move on to the next schema in the set.
5. Repeat from step 2 until there are no more changes to make i.e. no FD that does not satisfy BCNF.

05 Normalisation Decomposition: BCNF Decomposition Question

7. For each of the sets of dependencies in question 4:

i. if R is not already in 3NF, decompose it into a set of 3NF relations

ii. if R is not already in BCNF, decompose it into a set of BCNF relations

a. $C \rightarrow D, \quad C \rightarrow A, \quad B \rightarrow C$

b. $B \rightarrow C, \quad D \rightarrow A$

c. $ABC \rightarrow D, \quad D \rightarrow A$

d. $A \rightarrow B, \quad BC \rightarrow D, \quad A \rightarrow C$

e. $AB \rightarrow C, \quad AB \rightarrow D, \quad C \rightarrow A, \quad D \rightarrow B$

f. $A \rightarrow BCD$