# Relational Database Design Theory (I)

Spring, 2024

# Course overview

## Relational databases

- Relational data model ✓
- Relational algebra ✓
- Structured query language ✓
- Relational database design theory

## DBMS internals

- Database storage
- Indexing
- Query processing and optimization
- Concurrency control
- Crash recovery

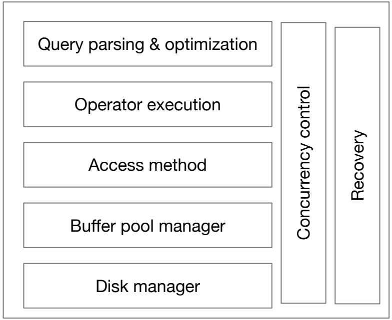


Figure: Classical DBMS architecture

Other topics (TBD): (i) graph database, (ii) parallel query processing

# A bad design

| sid | cid | cname | room | grade |
|---|---|---|---|---|
| 123 | AI-3613 | Database | 1-108 | A+ |
| 223 | AI-3613 | Database | 1-108 | B+ |
| 123 | CS-101 | CS Intro. | 3-325 | A |
| 334 | CS-101 | CS Intro. | 3-325 | A- |
| 345 | ICE-1404P | Database | 2-203 | A |

Table: R(sid, cid, cname, room, grade)

- Data redundancy: information for the same course is recorded multiple times
- Update/insertion/deletion anomalies

# Anomalies in a bad design

| sid | cid | cname | room | grade |
|---|---|---|---|---|
| 123 | AI-3613 | Database | 1-108 | A+ |
| 223 | AI-3613 | Database | 1-108 | B+ |
| 123 | CS-101 | CS Intro. | 3-325 | A |
| 334 | CS-101 | CS Intro. | 3-325 | A- |
| 345 | ICE-1404P | Database | 2-203 | A |

Table: R(sid, cid, cname, room, grade)

- Insertion anomaly: Cannot add data to db due to the absence of other data.
  - What happens if we want to add a new course CS2950?
- Deletion anomaly: Lose unintended information as a side effect when deleting tuples.
  - What happens if the student with sid 345 quit the course ICE-1404?
- Update anomaly: To update info of one tuple, we may have to update others as well.
  - What happens if the room of AI-3613 is changed?

# A good design

Decompose $R$ into two smaller tables $R_1$ and $R_2$.

| sid | cid | cname | room | grade |
|---|---|---|---|---|
| 123 | AI-3613 | Database | 1-108 | A+ |
| 223 | AI-3613 | Database | 1-108 | B+ |
| 123 | CS-101 | CS Intro. | 3-325 | A |
| 334 | CS-101 | CS Intro. | 3-325 | A- |
| 345 | ICE-1404P | Database | 2-203 | A |

Table: $R$(sid, cid, cname, room, grade)

| sid | cid | grade |
|---|---|---|
| 123 | AI-3613 | A+ |
| 223 | AI-3613 | B+ |
| 123 | CS-101 | A |
| 334 | CS-101 | A- |
| 345 | ICE-1404P | A |

Table: $R_1$(sid, cid, grade)

- The decomposition is lossless since

$$R = R_1 \bowtie R_2.$$

  That is, all tuples are preserved.

- Redundancy and anomalies are eliminated.

| cid | cname | room |
|---|---|---|
| AI-3613 | Database | 1-108 |
| CS-101 | CS Intro. | 3-325 |
| ICE-1404P | Database | 2-203 |

Table: $R_2$(cid, cname, room)

# Database design theory

- Decide whether a particular relation schema $R$ is in “good” from.

- In the case that $R$ is not in “good” form, decompose $R$ into a set of relation schemas $\{R_1, R_2, \ldots, R_n\}$ such that each $R_i$ is in good form (normal form).

- The resulting decomposition is lossless and helps avoid anomalies.

# Agenda

- Functional dependency theory (this lecture)
- NF's and decomposition algorithms (next lecture)

# Functional Dependency Theory

# Functional dependencies

Let $X = \{A_1, \ldots, A_n\}$ and $Y = \{B_1, \ldots, B_m\}$ be sets of attributes.

**Definition** [Functional dependency]

A functional dependency (FD) is of the form

$$X \to Y$$

that requires the attributes of $X$ functionally determining the attributes $Y$.

In particular, a relation $R$ satisfies $X \to Y$ if for every two tuples $t_1$ and $t_2$ of $R$

$$\wedge_{i=1}^{n} t_1[A_i] = t_2[A_i] \to \wedge_{j=1}^{m} t_1[B_j] = t_2[B_j].$$

- FD's are unique-value constraints.
- A FD $X \to Y$ holds on a relational schema $R$ if every instance of $R$ satisfies $X \to Y$.
- If $Y \subseteq X$, then $X \to Y$ is trivial.

# Notation convention

- $A_1 \ldots A_n$ represents $\{A_1, \ldots, A_n\}$.
- Attributes: $A$, B, C, D, E
- Sets of attributes: X, Y, Z
- XY represents $X \cup Y$

# FD example

| sid | cid | cname | room | grade |
|---|---|---|---|---|
| 123 | AI-3613 | Database | 1-108 | A+ |
| 223 | AI-3613 | Database | 1-108 | B+ |
| 123 | CS-101 | CS Intro. | 3-325 | A |
| 334 | CS-101 | CS Intro. | 3-325 | A- |
| 345 | ICE-1404P | Database | 2-203 | A |

Table: R(sid, cid, cname, room, grade)

- cid $\rightarrow$ cname
- cid $\rightarrow$ room
- cid $\rightarrow$ {cname, room}
- sid, cid $\rightarrow$ grade

# Keys and FD's

**Definition**

Given a relation $R$, a set $X$ of attributes is a candidate key if

- $X$ functionally determines all other attributes of $R$, i.e., $X$ is a superkey.
- No proper subset of $X$ functionally determines all other attributes of $R$.
  –That is, $X$ is minimal.

# Reasoning about FD's

## Definition

- A set F of FD's logically implies a set G of FD's if every relation instance that satisfies all the FD's in F also satisfies all the FD's in G.
- F and G are equivalent if (i) F logically implies G and (ii) G logically implies F.

## Example

- $\{A \rightarrow B\}$ logically implies $\{AC \rightarrow BC\}$.
- $\{A \rightarrow B, B \rightarrow C\}$ logically implies $\{A \rightarrow C\}$.
- $\{A \rightarrow B, B \rightarrow C\}$ is equivalent to $\{A \rightarrow B, B \rightarrow C, A \rightarrow C\}$.
- $\{A_1A_2 \rightarrow B_1B_2B_3\}$ is equivalent to $\{A_1A_2 \rightarrow B_1, A_1A_2 \rightarrow B_2, A_1A_2 \rightarrow B_3\}$.

# Closure of attributes

**Definition** [Attribute closure]

Let $X$ be a set of attributes and $F$ be a set of FD's. The closure of $X$ under $F$, written as $X_F^+$, is the set of all attributes $B$ such that $F$ logically implies $X \to B$.

**Example**

Let $F = \{A \to B, A \to C, CD \to E, CD \to K, B \to E\}$. Then

- $\{A\}_F^+ = \{A, B, C\}$, $\{C, D\}_F^+ = \{C, D, E, K\}$.
- $\{A, D\}_F^+ = \{A, B, C, D, E, K\}$.

- We omit the subscript $F$ and write $X^+$ if $F$ is clear from the context.
- To determine whether $F$ logically implies $X \to Y$ it suffices to check whether $Y \subseteq X^+$.
- To see if $X$ is a superkey of $R$, it suffices to check if $X^+$ contains all the attributes of $R$.

# Computing attribute closure

| Input: A set of attributes $X$ and a set of FD's $F$ |
|---|
| Output: $X_F^+$ |
| 1. $Z \leftarrow X$; |
| 2. repeat |
| 3. if ex. $X' \rightarrow Y'$ in $F$ s.t. $X' \subseteq Z$ and $Y' \setminus Z \neq \emptyset$ |
| 4. then $Z \leftarrow Z \cup Y'$; |
| 5. until ($Z$ no longer changes); |
| 6. return $Z$; |

Figure: Computing attribute closure

- $F = \{A \rightarrow B, A \rightarrow C, CD \rightarrow E, CD \rightarrow K, B \rightarrow E\}$
- What is $\{A, D\}_F^+$?
- Is $\{A, D\}$ a superkey/candidate key?

# Algorithm correctness (I)

Correctness. $\widehat{X_F^+} = X_F^+$, where $\widehat{X_F^+}$ the algorithm output.

- $\widehat{X_F^+} \subseteq X_F^+$. $X \subseteq X_F^+$ and by I.H. every new element introduced in line 4 is also in $X_F^+$.
- $X_F^+ \subseteq \widehat{X_F^+}$. Let B be an attribute not in $\widehat{X_F^+}$. It suffices to show that F cannot imply $X \to B$. That is, there is a table R s.t.(i) R satisfies F, and (ii) R does not satisfy $X \to B$.

  Let $\widehat{X_F^+} = \{A_1, A_2, \ldots, A_n\}$ and $\overline{\widehat{X_F^+}} = \{B_1, B_2, \ldots, B_m\}$. We define R as

| $A_1$ | $A_2$ | $\ldots$ | $A_n$ | $B_1$ | $B_2$ | $\ldots$ | $B_m$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | $\ldots$ | 1 | 1 | 1 | $\ldots$ | 1 |
| 1 | 1 | $\ldots$ | 1 | 0 | 0 | $\ldots$ | 0 |

  It should be clear that R does not satisfy $X \to B$. It remains to verify that R satisfies F.

Claim. R satisfies F.

# Algorithm correctness (II)

---

Input: A set of attributes $X$ and a set of FD's $F$
Output: $X_F^+$

---

1. $Z \leftarrow X$;
2. repeat
3.     if ex. $X' \rightarrow Y'$ in $F$ s.t. $X' \subseteq Z$ and $Y' \setminus Z \neq \emptyset$
4.     then $Z \leftarrow Z \cup Y'$;
5. until ($Z$ no longer changes);
6. return $Z$;

---

Figure: Computing attribute closure

Claim. $R$ satisfies $F$.

Proof. We prove it by contraction.

Let $X' \rightarrow Y'$ be an FD in $F$ that $R$ does not satisfy. By construction, we must have $X' \subseteq \{A_1, A_2, \ldots, A_n\}$ and $Y' \cap \{B_1, B_2, \ldots, B_m\} \neq \emptyset$.

It follows that all the attributes in $Y'$ should also be included in $\widehat{X}_F^+$ (lines 3-4).

This contradicts to that fact that $Y' \cap \{B_1, \ldots, B_m\} \neq \emptyset$. □

# Closure of FD's

**Definition**

The closure of $F$, denoted by $F^+$, is the set of all FD's logically implied by $F$.

Question. Given a set of FD's $F$, how to decide whether $X \to Y \in F^+$?

- Approach 1: compute $X^+$ and check whether $Y \subseteq X^+$.
- Approach 2: use Armstrong's axioms.

# Armstrong's axioms

- Reflexivity: If $Y \subseteq X$, then $X \rightarrow Y$.
- Augmentation: If $X \rightarrow Y$, then $XZ \rightarrow YZ$.
- Transitivity: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$.

---

Theorem (Armstrong '74). The Armstrong's axioms are both sound and complete.

- Soundness: Only correct FD's are derived.
- Completeness: Every FD in $F^+$ can be derived by using the axioms.

# Motivation for canonical cover

- A set of FD's $F$ defines a set of unique-value constraints.
- We want a minimal set $F'$ of FD's to reduce constraint checking cost.
- $F'$ should be equivalent to $F$ to ensure correctness.

A canonical cover $F_c$ of $F$ is a minimal set of FD's equivalent to $F$.

# Extraneous attributes

An attribute of a FD $X \to Y$ in FD is extraneous if we can remove it without changing $F^+$.

- An attribute $A \in X$ is extraneous and can be removed from the LHS of $X \to Y$ if

$$F \text{ logically implies } (F \setminus \{X \to Y\}) \cup \{(X \setminus \{A\}) \to Y\}.$$

- Example. $F = \{AB \to C, A \to D, D \to C\}$

- An attribute $B \in Y$ is extraneous and can be removed from the RHS of $X \to Y$ if

$$(F \setminus \{X \to Y\}) \cup \{X \to (Y \setminus \{B\})\} \text{ logically implies } F.$$

- Example. $F = \{A \to CD, D \to C\}$

**Lemma 1**

1. $A \in X$ is extraneous in $X \to Y$ iff $Y \subseteq (X \setminus \{A\})^+_F$.
2. $B \in Y$ is extraneous in $X \to Y$ iff $B \in X^+_{F'}$, where $F' = (F \setminus \{X \to Y\}) \cup \{X \to (Y \setminus \{B\})\}$.

# Canonical cover

**Definition**

A canonical cover $F_c$ for $F$ is a set of FD's equivalent to $F$ such that

- No FD in $F_c$ contains an extraneous attribute.
- Each LHS of a FD in $F_c$ is unique.

# Computing canonical cover

**Input:** A set $F$ of FD's
**Output:** A canonical cover $F_c$ of $F$

$F_c \leftarrow F$;
**repeat**
    **for each** pair of FD's $X \rightarrow Y_1$ and $X \rightarrow Y_2$ in $F_c$ **do**
        replace them with $X \rightarrow Y_1Y_2$;
    **if** ex. a FD in $F_c$ with an extraneous attribute **then**
        remove the extraneous attribute and update $F_c$;
**until** ($F_c$ no longer changes)
**return** $F_c$;

Figure: Computing canonical cover

# Canonical cover examples

Let $F = \{A \to BC, B \to C, A \to B, AB \to C\}$.

- $F_c^0 = \{A \to BC, B \to C, AB \to C\}$
- $F_c^1 = \{A \to B, B \to C, AB \to C\}$
- $F_c^2 = \{A \to B, B \to C\}$

Let $F = \{A \to BC, B \to AC, C \to AB\}$.

- $F_c = \{A \to B, B \to C, C \to A\}$.
- $F_c = \{A \to C, C \to B, B \to A\}$.
- $F_c = \{A \to C, B \to C, C \to AB\}$.

# Recap

- A function dependency $X \rightarrow Y$ is a unique-value constraint. It means that

  whenever two tuples agree on all attributes in $X$, they must also agree on all attributes in $Y$.
- $X_F^+$: the closure of $X$ under $F$ is the set of all attributes functionally determined by $X$.
- A canonical cover $F_c$ of $F$ is a minimal set of FD's equivalent to $F$.
- Two simple algorithms to compute $X_F^+$ and $F_c$.

$\Rightarrow$ We will use FD as a tool to design normalization algorithms.