UNAM FACULTAD DE INGENIERIA

BASES DE DATOS

Tarea 05

Martínez García José Eduardo

AXIOMAS DE ARMSTRONG

The term Armstrong Axioms refers to the sound and complete set of inference rules or axioms, introduced by William W. Armstrong, that is used to test the logical implication of functional dependencies [1]. Armstrong Axioms are a set of references (or, more precisely, inference rules) for inferring all of a relational database's functional dependencies [2].

If F is a set of functional dependencies then the closure of F, denoted as F+, is the set of all functional dependencies logically implied by F

$$F^+ \rightarrow \{...\}$$

AXIOMS

- **Axiom of Reflexivity:**

If **A** is a set of attributes and **B** is a subset of **A**, then **A** holds **B**. If **B⊆A** then **A→B**. This property is trivial property[1].

$$if\ B \subseteq A$$

$$A \rightarrow B$$

Example: A university student profile contains attributes: *{Student_ID, Name, Major}.*

- Since ***{Student_ID, Name}*** is a subset of *{Student_ID, Name, Major},* we can say: *{StudentID,Name,Major}→{Name}*

- This means that if we know all three attributes, we automatically know "Name," which is trivial [3].

- **Axiom of Augmentation:**

If **A→B** holds and Y is the attribute set, then **AY→BY** also holds. That is adding attributes to dependencies, does not change the basic dependencies. If **A→B**, then **AC→BC** for any **C** [1].

$$if\ A \rightarrow B$$

$$AC \rightarrow BC$$

Example: Consider an employee database where:

- Employee_ID → Employee_Name (An employee ID determines their name).
- If we introduce another attribute, "Department," the dependency still holds:

  {EmployeeID,Department}→{EmployeeName,Department}
- Adding "Department" doesn't affect the basic dependency [3].

- **Axiom of Transitivity:**

Same as the transitive rule in algebra, if **A→B** holds and **B→C** holds, then **A→C** also holds. **A→B** is called A functionally which determines B. If **X→Y** and **Y→Z**, then **X→Z** [1].

$$If\ X \rightarrow Y\ and\ Y \rightarrow Z$$

$$X \rightarrow Z$$

Example: In an airline booking system:

- Passenger_ID → Seat_Number (A passenger ID determines a seat number).
- Seat_Number → Flight_Number (A seat number is associated with a flight).
- Therefore, Passenger_ID → Flight_Number (A passenger ID ultimately determines the flight they are on) [3].

SECONDARY RULES

- **Union:**

If A→B holds and A→C holds, then A→BC holds. If X→Y and X→Z then X→YZ [1].

Example: In an online shopping system:

- Order_ID → Customer_ID (An order ID determines the customer).
- Order_ID → Order_Date (An order ID also determines the order date).
- So, Order_ID → {Customer_ID, Order_Date} (An order ID determines both customer and order date) [3].

- **Composition:**

If A→B and X→Y hold, then AX→BY holds [1].

Example: In a university system:

- Student_ID → Student_Name (A student ID determines their name).
- Course_Code → Course_Title (A course code determines the course title).
- So, {Student_ID, Course_Code} → {Student_Name, Course_Title} (A combination of student ID and course code determines both student name and course title) [3].

- **Decomposition:**

If A→BC holds then A→B and A→C hold. If X→YZ then X→Y and X→Z [1].

Example: In a vehicle registration system:

- License_Plate → {Car_Model, Car_Owner} (A license plate determines both the car model and owner).
- By decomposition:

$$LicensePlate \rightarrow CarModelLicense_{Plate}$$

$$LicensePlate \rightarrow CarOwnerLicense_{Plate}$$

- Each piece of information can be determined separately [3].

- **Pseudo Transitivity:**

If A→B holds and BC→D holds, then AC→D holds. If X→Y and YZ→W then XZ→W [1].

Example: In an HR system

- Employee_ID → Department (An employee ID determines the department).
- {Department, Manager_ID} → Budget (A department and manager together determine the budget).
- Therefore, {Employee_ID, Manager_ID} → Budget (An employee ID and manager ID together determine the budget) [3].

- **Self Determination:**

It is similar to the Axiom of Reflexivity, i.e. A→A for any A [1].

Example: This is trivial in any system. For example, a Social Security Number (SSN) determines itself [3]:

$$SSN \rightarrow SSNSSN$$

- **Extensivity:**

Extensivity is a case of augmentation. If AC→A, and A→B, then AC→B.
Similarly, AC→ABC and ABC→BC. This leads to AC→BC [1].

Example: In an academic grading system:

- {Exam_ID, Student_ID} → Exam_ID (Knowing both doesn't change that we know the exam ID).
- Exam_ID → Exam_Schedule (An exam ID determines the schedule).
- By extensivity: {Exam_ID, Student_ID} → Exam_Schedule (Knowing both still determines the exam schedule) [3].

Referencias

- [1]GeeksforGeeks. “Armstrong's Axioms in Functional Dependency in DBMS - GeeksforGeeks”. GeeksforGeeks. Accedido el 1 de abril de 2025. [En línea]. Disponible: https://www.geeksforgeeks.org/armstrongs-axioms-in-functional-dependency-in-dbms/

- [2]Accedido el 1 de abril de 2025. [En línea]. Disponible: https://www.naukri.com/code360/library/armstrong-axioms

- [3]“Types of Functional Dependencies in DBMS + 4 Examples”. monday.com Blog. Accedido el 1 de abril de 2025. [En línea]. Disponible: https://monday.com/blog/project-management/functional-dependencies-2/