The Relational Data Model

Constraints

SWEN304 / SWEN435

Trimester 1, 2024

Lecturer: Kevin Shedlock

Engineering and Computer Science

VICTORIA UNIVERSITY OF WELLINGTON TE HERENGA WAKA

Outline

- Introduction to the RDM Constraints
  - Domain constraints, Attribute constraints
  - Key constraint, Unique constraint
  - Interrelation constraints
  - Constraint violation: database updates

- RDM Operations
  - insert,
  - delete, and
  - modify

Quick Review - Definition for RDM Schema:

- Given a relation schema $N(A_1{:}D_1, A_2{:}D_2, \ldots\ldots, A_n{:}D_n)$
  - N is the name of the **relation**
  - $A_1, A_2, \ldots\ldots, A_n$ are the **attributes** of the relation
  - D_i is the **domain** of attribute A_i: $dom(A_i) = D_i$
- For convenience we sometimes omit the domain assignment from a relation schema
- Relation $r(N)$: a specific **state** (or "value" or "population") of N as a *set of tuples* (rows)
  - $r(N) = \{t_1, t_2, \ldots, t_n\}$ where each t_i is an n-tuple
  - $t_i = <v_1, v_2, \ldots, v_n>$ where each v_j is an element of $dom(A_j)$
- $r(R) \subset dom(A_1) \times dom(A_2) \times \ldots . \times dom(A_n)$

***Proper* Subset** **Cartesian Product**

Introduction to Constraints

- Introduction to the RDM Constraints
  - Definition, Rules, Constrain Types
  - Domain constraints, Attribute constraints
  - Key constraint, Unique constraint
  - Interrelation constraints

- Constraint violation: database updates

Definition of RDM Constraints

- A **condition** that must hold on **all** valid relation states
- Fundamental to databases
- Real world has constraints on what is possible
- A database is an abstraction of the real world
  $\Rightarrow$ should reflect these constraints
- We cannot ensure that the database is always correct, but we can ensure that it is meaningful
- Constraints are derived from the semantics of the TTUPA (rules of behaviour, business rules)

Relational Constraints Rules

- The constraint rules in a relational database model help maintain the integrity and consistency of data.

- These rules include primary key constraints, unique constraints, foreign key constraints, check constraints, default constraints, not null constraints, multi-column constraints…

Types of Constraints on the Schema

- The basic relation schema constraints C are:
- Domain constraint
- Attribute constraint
- Key constraint, entity integrity constraint
- Referential integrity constraint
- Unique constraint

- Some other relational data model constraints, like data dependencies, e.g. functional dependencies, *will be covered later in the course*

Use Case Modelling Constraints

You are given a relation schema $N(R, C)$ and an instance $r(N)$. Suppose C does not contain any key specification

- Inferring keys from instances is very hard if even possible, since there are so many of them
- By analysing instances and constraint rules, you may conclude which subsets of R cannot be a key (not unique)
- Also, from instances you may infer which key constraints are not violated by instances.

Updates and Constraints

- No update operation should leave a database in an inconsistent state (with violated constraints)

- A DBMS must take the actions necessary to prevent a constraint violation:
  - **reject**: do not allow the operation
  - **cascade**: propagate the operation by making necessary consequential changes
  - **set null**, or **set default**: reset other values to maintain consistency

Modify and Constraint Violations

- Modifying/updating the values of attributes in a tuple may violate constraints
  - Attribute/domain constraints
    Response: **reject** (like insert)
  - Key constraints (if attribute is part of a key)
    Response: treat as a delete followed by an insert
  - Referential integrity constraints (if attribute is part of a foreign key).
    Response: **reject** (like insert), or **cascade**, or **set null,** or **set default** (like delete)

Inserts and Constraint Violations

- Inserting a new tuple could violate:
  - Attribute/domain constraints
    (a value is not of the right type or in the required range)
  - Uniqueness constraints
    (the values of the key attributes duplicate another tuple)
  - Not Null constraints
    (an attribute has the value null when it shouldn't)
  - Referential Integrity constraints
    (the values of the attributes of a foreign key do not match any tuple in the other relation)

- Response:
  - **Reject** the operation – there is no change that the DBMS system could safely make to resolve the inconsistency

RDM Violations

Given a relation

STUDENT			
Id	**Lname**	**Fname**	**Major**
300111	Smith	Susan	COMP
300121	Bond	James	MATH
300132	Smith	Susan	COMP
300135	John	Cecil	MATH

What is r(STUDENT) [Lname, Major]?

a)

Lname	Major
Smith	COMP
Bond	MATH
Smith	COMP
John	MATH

b)

Lname	Major
Bond	MATH
Smith	COMP
John	MATH

c)

Lname	Major
Smith	COMP
Bond	MATH
John	MATH

Attribute Constraint Examples

- Relation schema: STUDENT
  - Attribute: Fname
  - *Dom* (STUDENT, Fname) = *STRING*
  - *Range* (STUDENT, Fname) = *none*
  - *Null* (STUDENT, Fname) = *not null* user requires name

- Relation schema: GRADES
  - Attribute: Grade
  - *Dom* (GRADES, Grade) = *STRING*
  - *Range* (GRADES, Grade): one of {'A+', 'A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'C-', 'D', 'E', 'K', 'P', 'F', 'G', 'J', 'L', 'Z'}
  - *Null* (GRADES, Grade) = Y //*yes, null value allowed*

Restrictions

- Let $R = \{A_1, \ldots, A_n\}$ be the set of attributes of a relation schema N and $r(N) = \{t_1, \ldots, t_n\}$

- Restriction of a tuple t onto $\{A_k, \ldots, A_m\} \subseteq \{A_1, \ldots, A_n\}$, denoted as $t[A_k, \ldots, A_m]$, refers to a **sublist** of values $(v_k, \ldots, v_m)$ in $t = (v_1, \ldots, v_n)$, for $1 \leq k$ and $m \leq n$

- Example: STUDENT = {id, Lname, Fname, Major}

  t = (300121, Bond, Jame, MATH)

  t [*Lname*] = <Bond>,

  t [Fname, Major] = <James, Math>

- Restriction of a relation r onto a set of attributes $\{A_k, \ldots, A_m\}$, is denoted by:

  $r(N)[A_k, \ldots, A_m] = \{t[A_k, \ldots, A_m] \mid t \in r\}$

Attribute Constraint

- Domain constraints restrict the attribute values, but may not be sufficient
- Attribute constraints can further restrict attribute values
- Generally, the attribute constraint of an attribute A within a relation schema N is defined as:

$$(Dom(N, A), Range(N, A), Null(N, A))$$

  - $Dom(N, A)$ associates attribute A in N with a domain via domain name D
  - $Range(N, A)$ is used to further restrict the range of allowable attribute A values in the relations over N
  - $Null(N, A)$ specifies whether attribute A may or may not have a null value in any instance over N

Relation Schema Key

- A relation is a **set** of tuples, hence all tuples must be distinct
- Let $N(A_1{:}D_1, \ldots, A_n : D_n)$ be a relation schema,
  $X = \{A_k, \ldots, A_m\} \subseteq \{A_1, \ldots, A_n\}$,
  X is a relation schema key of N, if:

$1^o\ (\forall r(N))(\forall u, v \in r(N))(u[X] = v[X] \Rightarrow u = v)$ (unique)

$2^o\ (\forall Y \subset X)(\neg 1^o)$ (minimal)

$3^o\ (\forall r(N))(\forall t \in r(N))(\forall A \in X)(t[A] \neq \omega)$ (not null)

- A relation schema key is **not** allowed to have a null value as the key value uniquely identifies the individual tuples
- A relation schema key is also called a minimal key

Primary Key and Entity Integrity

- A primary key, also called a primary keyword, is a column in a relational database table that's distinctive for each record.

- It's a unique identifier, such as a student number, health index number, staff number.

- Examples:
  - STAFF(IRDNo, Staff_id, Fname, Lname, DoB)
    - K_1 = {Lname, DoB}
    - K_p = {Staff_id}
  - STUDENT(Id, Fname, Lname, Major) (primary key underlined)

Primary Key and Entity Integrity

- A relation schema may have more than one key K
- One of the relation schema keys K is designated as a primary key denoted by K_p
  - a key used in TTUPA for identification most frequently
- Entity integrity constraint: no primary key values can be null

Referential Integrity – A Formal Definition

- Referential integrity is a term used in database design to describe the relationship between two tables.
- It is important because it ensures that all data in a database remains consistent and up to date.
- It helps to prevent incorrect records from being added, deleted, or modified

Referential Integrity Constraints

- A set of referential integrity constraints forms the most important subset of the relational database schema constraints set IC

- Very often, referential integrities are the **only** interrelation constraints considered

- For example:
  - Database schema name: TTUPA
  - Has a set of relation schemas:

$$S = \{\text{STUDENT, GRADES, COURSE}\}$$

$$IC = \{\text{GRADES[Id]} \subseteq \text{STUDENT[Id]},$$
$$\text{GRADES[Course\_id]} \subseteq \text{COURSE[Course\_id]}\ \}$$

An Example – Consistent relations?

STUDENT			
Id	**Lname**	**Fname**	**Major**
300111	Smith	Susan	COMP
300121	Bond	James	MATH
300143	Bond	Jenny	MATH
300132	Smith	Susan	COMP

?

COURSE			
Course_id	**Cname**	**Points**	**Dept**
SWEN304	DB sys	15	Engineering
COMP301	softEng	20	Engineering
MATH214	DisMat	15	Mathematics

GRADES[Id] ⊆ STUDENT[Id]

GRADES[Course_id] ⊆ COURSE[Course_id]

GRADES		
Id	**Course_id**	**Grade**
300111	SWEN304	A+
300111	COMP301	A
300111	MATH214	A
300121	COMP301	B
300132	COMP301	C
300121	SWEN304	B+
300132	SWEN304	C+

$$(\forall u \in r\,(N_2))(\exists v \in r\,(N_1))(u[Y] = v[X] \vee (\exists i \in \{1,\ldots, m\})(u[B_i] = \omega))$$

An Example – Inconsistent relations?

STUDENT			
Id	**Lname**	**Fname**	**Major**
300111	Smith	Susan	COMP
300121	Bond	James	MATH
300143	Bond	Jenny	
300132	Smith	Susan	COMP

COURSE			
Course_id	**Cname**	**Points**	**Dept**
SWEN304	DB sys	15	Engineering
COMP301	softEng	20	Engineering
MATH214	DisMat	15	Mathematics

GRADES		
Id	**Course_id**	**Grade**
300111	SWEN304	A+
300111	COMP301	A
300111	MATH114	A
300121	COMP301	B
300132	COMP301	C
300121	SWEN304	B+
300138	SWEN304	C+

To avoid inconsistencies with reality we first need to observe the actual data dependencies and make them explicit (specify them)

A Common Pitfall – Foreign key

- Consider the following relation schemas:

  PERSON({Name, Birthday, Address}, {Name + Birthday})

  STUDENT({ID, Name, Birthday}, {ID})

- We define a foreign key on STUDENT:

  $\text{STUDENT[Name, Birthday]} \subseteq \text{PERSON[Name, Birthday]}$

- This is **NOT equivalent** to: (Why?)

  $\text{STUDENT[Name]} \subseteq \text{PERSON[Name]}$, and

  $\text{STUDENT[Birthday]} \subseteq \text{PERSON[Birthday]}$

PERSON		
Name	Birthday	Address
Grampa Simpson	01.01.1900	16 Park Ave
Apu Nahasapeemapetilon	29.02.1961	98 Ada St.

STUDENT		
ID	Name	Birthday
007	Apu Nahasapeemapetilon	01.01.1900
008	Grampa Simpson	29.02.1961

Other Types of Constraints

- Semantic Integrity Constraints:
  - based on application semantics and cannot be expressed by the model per se
  - Example: "the max number of courses a student can enroll in one year"

- A **constraint specification** language may have to be used to express these
- SQL-99 allows triggers and ASSERTIONS to express for some of these

Business Rules/ Requirements

- Business rules are implemented as triggers in relational databases.

- Business rules represent information about the real world and database is collection of related information.

- Business rules are statements that imposes some form of constraint on a specific aspect of the database, such as the attributes that impacts a set of tuple's in some way and its relationship.

Business Rules Example for TTUPA

As an example, Te Taupanga University of Performing Arts, students are required to participate in performance groups tas part of the examination. it's crucial to ensure that every performance has at least one artist associated with it.

This rule ensures that performances are properly credited to the student (artist) involved which is credit to grades.

- **Domains:** The RDM is represented by *domain entities* containing *artists* who are involved with *performance* items. Each performance **must** be associated with at least one artist.

- **Relationship:** One performance can involve one or more artists (One-to-Many relationship)

Business Rules Example for TTUPA

- **Attributes:** Consist of:
  - A Performance Relation with:
    - PerformanceID (Primary Key);
    - PerformanceName
    - Date
    - Time
    - VenueID (Foreign Key)

Performance

PerformanceID	PerformanceNam	Date	Time	VenueID
1	Performance1	1/01/2024	19.00	101
2	Performance2	2/01/2024	20.00	102

Business Rules Example for TTUPA

- An Artist Relation:
  - ArtistID (Primary Key)
  - ArtistName
  - Genre
  - Other relevant attributes

Artist			
ArtistID	**ArtistName**	**Genre**	**Other**
1371352	Kiana	Drame	Null
1371337	Awhi	Comedy	Null
1371240	April	Music	Null
1370718	Andrew	Kapahaka	Null

Business Rules Example for TTUPA

- In this RDM, one Performance-Artist Association Relation ensures that each performance is associated with one or more artists. And one incorrect reports the results!
- The foreign keys PerformanceID and ArtistID in each table references the primary keys of the Performance and Artist tables, respectively. This structure enforces the business rule that each performance must be associated with at least one artist.

Performance-Artisit Association	
PreformanceID	ArtistID
1	1371352
1	1371352
2	1371337
3	1371240
4	1370718

Performance-Artisit Association	
PreformanceID	ArtistID
1	1371352
1	1371352
2	1371337
2	1371240
1	1370718

Business Rules Example for TTUPA

- Business rules are statements that imposes some form of constraint on a specific aspect of the database, such as the attributes that impacts a set of tuple's in some way and its relationship.

Wrap Up Constraints

- You are given a relation schema $N(R, C)$ and an instance $r(N)$. Most times C is contained within the rules as a key specification

- Inferring keys from instances can be ambiguous and difficult, since there can be many instances

- By analysing instances and $Null(N, A)$ constraints, you can only conclude which subsets of R cannot be a key

- Also, from instances you may infer which key constraints are not violated.

Summary

- Basic concepts of the relational data model:
  - Domain (set of values) – data type
  - Attribute (property of a set of similar TTUOPA objects)
  - Relation schema

- Relation schema constraints:
  - Domain, attribute, key, and unique constraints
- A **relational** database schema – a set of relation schemas and a set of interrelation constraints

- The referential integrity is the **most important** interrelation constraint: it links tuples of two relations
- A relational database is a set of relation instances that satisfy all relational and interrelation constraints

- No update operation should leave a database in an inconsistent state (with violated constraints)