CS 405G: Introduction to Database Systems

Relational Constraints

Topics Next

- Relational Integrity Constraint
- Reading assignment:
  - Chapter 3.2

Relational Integrity Constraints

- Constraints are *conditions* that must hold on *all* valid relation instances. There are four main types of constraints:
  1. Domain constraints
     1. The value of an attribute must come from its domain
  2. Key constraints
  3. Entity integrity constraints
  4. Referential integrity constraints

Primary Key Constraints

- A set of fields is a *candidate key* for a relation if :
  1. No two distinct tuples can have same values in all key fields, and
  2. This is not true for any subset of the key.
     - Part 2 false? A *superkey*.
     - If there are >1 keys for a relation, one of the keys is chosen (by DBA) to be the *primary key*.
- E.g., given a schema Student(sid: string, name: string, gpa: float) we have:
  - *sid* is a key for Students. (What about *name*?) The set {*sid, gpa*} is a superkey.

Key Example

- CAR (licence_num: string, Engine_serial_num: string, make: string, model: string, year: integer)
  - What is the candidate key(s)
  - Which one you may use as a primary key
  - What are the super keys

Entity Integrity

- Entity Integrity: The *primary key attributes* PK of each relation schema R in S cannot have null values in any tuple of r(R).
  - Other attributes of R may be similarly constrained to disallow null values, even though they are not members of the primary key.

Foreign Keys, Referential Integrity

- *Foreign key* : Set of fields in one relation that is used to `refer' to a tuple in another relation. (Must correspond to primary key of the second relation.) Like a `logical pointer'.
- E.g. *sid* is a foreign key referring to *Students*:
  - Student(sid: string, name: string, gpa: float)
  - Enrolled(*sid*: string, *cid*: string, *grade*: string)
  - If all foreign key constraints are enforced, *referential integrity* is achieved, i.e., no dangling references.
  - Can you name a data model w/o referential integrity?
    - Links in HTML!

Foreign Keys

- Only students listed in the Students relation should be allowed to enroll for courses.

Enrolled

sid	cid	grade
53666	Carnatic101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

- Or, use NULL as the value for the foreign key in the referencing tuple when the referenced tuple does not exist

In-Class Exercise

(Taken from Exercise 5.16)

Consider the following relations for a database that keeps track of student enrollment in courses and the books adopted for each course:

STUDENT(SSN, Name, Major, Bdate)

COURSE(Course#, Cname, Dept)

ENROLL(SSN, Course#, Quarter, Grade)

BOOK_ADOPTION(Course#, Quarter, Book_ISBN)

TEXT(Book_ISBN, Book_Title, Publisher, Author)

Draw a relational schema diagram specifying the foreign keys for this schema.

In-Class Exercise

Consider the following relations for a database that keeps track of student enrollment in courses and the books adopted for each course:

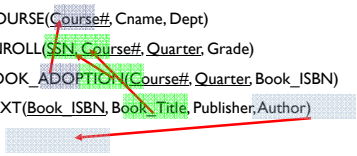
STUDENT(SSN, Name, Major, Bdate)

COURSE(Course#, Cname, Dept)

ENROLL(SSN, Course#, Quarter, Grade)

BOOK_ADOPTION(Course#, Quarter, Book_ISBN)

TEXT(Book_ISBN, Book_Title, Publisher, Author)



Draw a relational schema diagram specifying the foreign keys for this schema.

Other Types of Constraints

- Semantic Integrity Constraints:
  - based on application semantics and cannot be expressed by the model per se
  - e.g., "the max. no. of hours per employee for all projects he or she works on is 56 hrs per week"
  - A *constraint specification language* may have to be used to express these
  - SQL-99 allows triggers and ASSERTIONS to allow for some of these

Update Operations on Relations

- Update operations
  - INSERT a tuple.
  - DELETE a tuple.
  - MODIFY a tuple.
- Constraints should not be violated in updates

Example

- We have the following relational schemas
  - Student(sid: string, name: string, gpa: float)
  - Course(cid: string, department: string)
  - Enrolled(sid: string, cid: string, grade: character)
- We have the following sequence of database update operations. (assume all tables are empty before we apply any operations)
- INSERT<'1234', 'John Smith', '3.5> into Student

sid	name	gpa
1234	John Smith	3.5

Example (Cont.)

- INSERT<'647', 'EECS'> into Courses
- INSERT<'1234', '647', 'B'> into Enrolled
- UPDATE the grade in the Enrolled tuple with sid = 1234 and cid = 647 to 'A'.
- DELETE the Enrolled tuple with sid 1234 and cid 647

sid	name	gpa
1234	John Smith	3.5

cid	department
647	EECS

sid	cid	grade
1234	647	A

Exercise

- INSERT<'108', 'MATH'> into Courses
- INSERT<'1234', '108', 'B'> into Enrolled
- INSERT<'1123', 'Mary Carter', '3.8'> into Student

sid	name	gpa
1234	John Smith	3.5
1123	Mary Carter	3.8

cid	department
647	EECS
108	MATH

sid	cid	grade
1234	108	B

Exercise (cont.)

- A little bit tricky
- INSERT<'1125','Bob Lee','good'> into Student
  - Fail due to domain constraint
- INSERT<'1123', NULL, 'B'> into Enrolled
  - Fail due to entity integrity
- INSERT <'1233','647', 'A'> into Enrolled
  - Failed due to referential integrity

sid	name	gpa
1234	John Smith	3.5
1123	Mary Carter	3.8

cid	department
647	EECS
108	MATH

sid	cid	grade
1234	108	B

Exercise (cont.)

- A more tricky one
- UPDATE the cid in the tuple from Course where cid = 108 to 109

sid	name	gpa
1234	John Smith	3.5
1123	Mary Carter	3.8

cid	department
647	EECS
~~108~~ 109	MATH

sid	cid	grade
1234	~~108~~ 109	B

Update Operations on Relations

- In case of integrity violation, several actions can be taken:
  - Cancel the operation that causes the violation (REJECT option)
  - Perform the operation but inform the user of the violation
  - Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)
  - Execute a user-specified error-correction routine