Tutorial 1 - Relational Data Model

CSC343 - Introduction to Databases
Fall 2008

TA: Lei Jiang

This is a modified version of the tutorial material initially created by Yuan An.

Relation

- **Question**
  - (E. 3.2) How many distinct tuples are there in a relation instance with cardinality 22?
- **The Way of Thinking**
  - A relation instance is formally defined as *a set of tuples*
  - The cardinality of a set is defined as the *number of elements of the set*
    - A set cannot contain duplicates
  - If the cardinality of the relation (i.e., the set) is 22, …
  - there must be ***22*** distinct tuples in the relation instance
- **Answer**
  - There are 22 distinct tuples.

Key

- **Question**
  - (E. 3.4) What is the difference between a candidate key and the primary key for a given relation? What is a superkey?
- **Answer**
  - A candidate key (or simply key) of a relation is a *minimal subset* of the fields of the relation that *uniquely* identifies a tuple of the relation.
    - Two tuples in a legal instance cannot have identical key values
    - No subset of the key is a unique identifier for a tuple
  - A primary key is a key chosen by the DB designer
  - A superkey is any *set* of fields that contains a key.

Key

- **Question**
  - Based on following instance, give some examples of (sets of) atttributes that are *not* keys for sure.

RegNum	Surname	FirstName	BirthDate	DegreeProg
284328	Smith	Luigi	29/04/59	Computing
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Fine Art

- **The Way of Thinking**
  - check if values of any attribute contain duplicates
    - If yes, then it is cannot be a key
  - then try a set of 2, 3, … attributes
  - if each attribute contains no duplicates, you cannot make conclusion.
- **Answer**
  - {Surname}, {FirstName}, ...
  - {FirstName, BirthDate}, {BirthDate, DegreeProg}, ...
  - {Surname, BirthDate, DegreeProg}, ...

Key

- **Question**
  - Based on following instance, give some examples of (set of) atttributes that you can deduce they *might be* keys.

RegNum	Surname	FirstName	BirthDate	DegreeProg
284328	Smith	Luigi	29/04/59	Computing
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Fine Art

- **The Way of Thinking**
  - First check if any set of single attribute is an unique identifier in the relation
    - If yes, then it might be a key
  - Then check if any set of two attributes is an unique identifier in the relation
    - and it does not contain the the possible keys are have identied previously, if yes, then it might be a key
  - Then check if any set of three attributes ...
- **Answer**
  - Possible keys:
    - {RegNum}, {Surname, FirstName},{FirstName, BirthDate, DegreeProg}

Key

- **Question**
  - Based on following instance, give some examples of (set of) atttributes that you can deduce they *are* keys.

RegNum	Surname	FirstName	BirthDate	DegreeProg
284328	Smith	Luigi	29/04/59	Computing
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Fine Art

- **The Way of Thinking**
  - You cannot determine a key of a relation given only one instance of the relation.
  - A (candidate) key, as defined here, is not something that only might be a key, but is a key all possible instances of the relation.
  - The instance shown is just one possible “snapshot” of the relation.
    - At other times, the same relation may have an instance (or snapshot) that contains a totally different set of tuples
    - we cannot make predictions about those instances based only upon the instance that we are given
- **Answer**
  - We cannot make the conclusion what is the key.

Foreign Key

- **Question**
  - (E. 3.6) What is a foreign key constraint? Why are such constraints important?

- **Answer**
  - A foreign key constraint requires that the values on a set X of attributes of a relation R1 must appear as values for *the primary key* of the *referenced relation* R2.
  - Foreign key constraints are important because they provide safeguards for insuring the *referential integrity* of data.

Foreign Key

- **Question**
  - List all the foreign key constraints among these relations.

Offences

Code	Date	Officer	Dept	Registration
143256	25/10/1992	567	75	5694 FR
987554	26/10/1992	456	75	5694 FR
987557	26/10/1992	456	75	6544 XY
630876	15/10/1992	456	47	6544 XY
539856	12/10/1992	567	47	6544 XY

Officers

RegNum	Surname	FirstName
567	Brun	Jean
456	Larue	Henri
638	Larue	Jacques

Cars

Registration	Dept	Owner	…
6544 XY	75	Cordon Edouard	…
7122 HT	75	Cordon Edouard	…
5694 FR	75	Latour Hortense	…
6544 XY	47	Mimault Bernard	…

- **Answer**
  - *Offences.Officer* should have FKC, referencing Officers.RegNum
    - Real officers must be recording offences
  - *Offences.Registration, Offences.Dept should have FKCs referencing Cars.Registration and Cars.Dept respectively.*
    - Real cars must be recorded for offences

Foreign Key

- **Question**
  - Given following database schema with proper foreign keys defined
    - *Emp (eid: integer, ename: string, age: integer, salary: real)*
    - *Works (eid: integer, did: integer, pct_time: integer)*
    - *Dept (did: integer, dname: string, budget: real, managerid: integer)*
  - when a user attempts to delete a *Dept* tuple, what are the four options a DBMS has?

- **Answer**
  - *Also delete all Works tuples* that refer to it.
  - *Disallow deletion* of the *Dept* tuple if some *Works* tuple refers to it.
  - For every *Works* tuple that refers to it, set the *did* field to some *default value*.
  - For every *Works* tuple that refers to it, set the did field to *NULL*.

SQL DDL

- **Question**
  - Create ***Employee****(RegNo, FirstName, Surname, Dept, Salary, City)* using SQL.
    - Choose the data type for attributes you feel appropriate
    - {RegNo} and {FirstName, Surname} are two candidate keys
    - {RegNo} is designated as the primary key
    - FirstName and Surname cannot be null
    - The default value for Salary is 0
    - Dept is a foreign key, referring to Department.DeptName with following referential ICs:
      - If a Department row is deleted, all Employee.Dept values referring to it are set to null
      - If a Department row is updated, all all Employee.Dept values referring to it are updated also.

- **Answer**

```
CREATE TABLE Employee (
        RegNo CHAR(6),
        FirstName CHAR(20) NOT NULL,
        Surname CHAR(20) NOT NULL,
        Dept CHAR(15),
        Salary REAL(9) DEFAULT 0,
        City CHAR(15),
        PRIMARY KEY(RegNo),
        FOREIGN KEY(Dept) REFERENCES Department(DeptName)
                ON DELECT SET NULL
                ON UPDATE CASCADE,
        UNIQUE(FirstName,Surname)
)
```

SQL DDL

- **Question**
  - (E. 3.8) Write the SQL statements required to create the following relations, including appropriate primary and foreign key integrity constraints.
    - *Emp (eid: integer, ename: string, age: integer, salary: real)*
    - *Works (eid: integer, did: integer, pct_time: integer)*
    - *Dept (did: integer, dname: string, budget: real, managerid: integer)*