Introduction to Relational Databases Part 2: How?

Claude Rubinson
University of Houston—Downtown

rubinsonc@uhd.edu
cjr@grundrisse.org

September 28, 2011

Normalized Database

People

PersonUID	Person	Age
1	Smith	46
2	Jones	38
3	John	22
4	Mary	18
5	Jane	24
6	James	24

ClassTeacher

ClassUID	PersonUID
1	1
2	1
3	2

Classes

ClassUID	Class
1	Econ 101
2	Econ 201
3	Art Hist 101
4	Soc 101

ClassStudents

ClassUID	PersonUID
1	3
1	4
1	5
2	5
3	4
3	1

What is SQL?

- Structured Query Language
- Pronounced “Ess Que El” or “Sequel”
- Standardized, English-like language for interacting with Relational Database Management Systems (RDBMS)
- Set (technically, “Bag”) based
- Declarative Language

Query Language

```
SELECT Classes.class, count(*) as N,
   People.person as Teacher
FROM Classes, ClassStudents as CS,
   ClassTeacher as CT, People
WHERE Classes.classuid = CS.classuid AND
   Classes.classuid = CT.classuid AND
   CT.personuid = People.personuid
GROUP BY Classes.class, People.person
HAVING count(CS.personuid) >= 2;
```

```
     class     | n | teacher
---------------+---+---------
 Art Hist 101 | 2 | Jones
 Econ 101     | 3 | Smith
(2 rows)
```

Data Definition Language (DDL)

```
CREATE TABLE People (
   personuid serial PRIMARY KEY,
   person text NOT NULL,
   age integer CHECK (age >= 18)
   );
                personuid | person | age
               -----------+--------+-----
               (0 rows)

DROP TABLE People;
```

Data Definition Language (DDL)

```
ALTER TABLE People
ADD COLUMN birthday date;

 personuid | person | age | birthday
-----------+--------+-----+----------
(0 rows)

ALTER TABLE People
DROP COLUMN age;

 personuid | person | birthday
-----------+--------+----------
(0 rows)
```

Constraints

- Check constraints
- Unique constraints
- Primary Key constraints
- Foreign Key constraints

Normalized Database

People

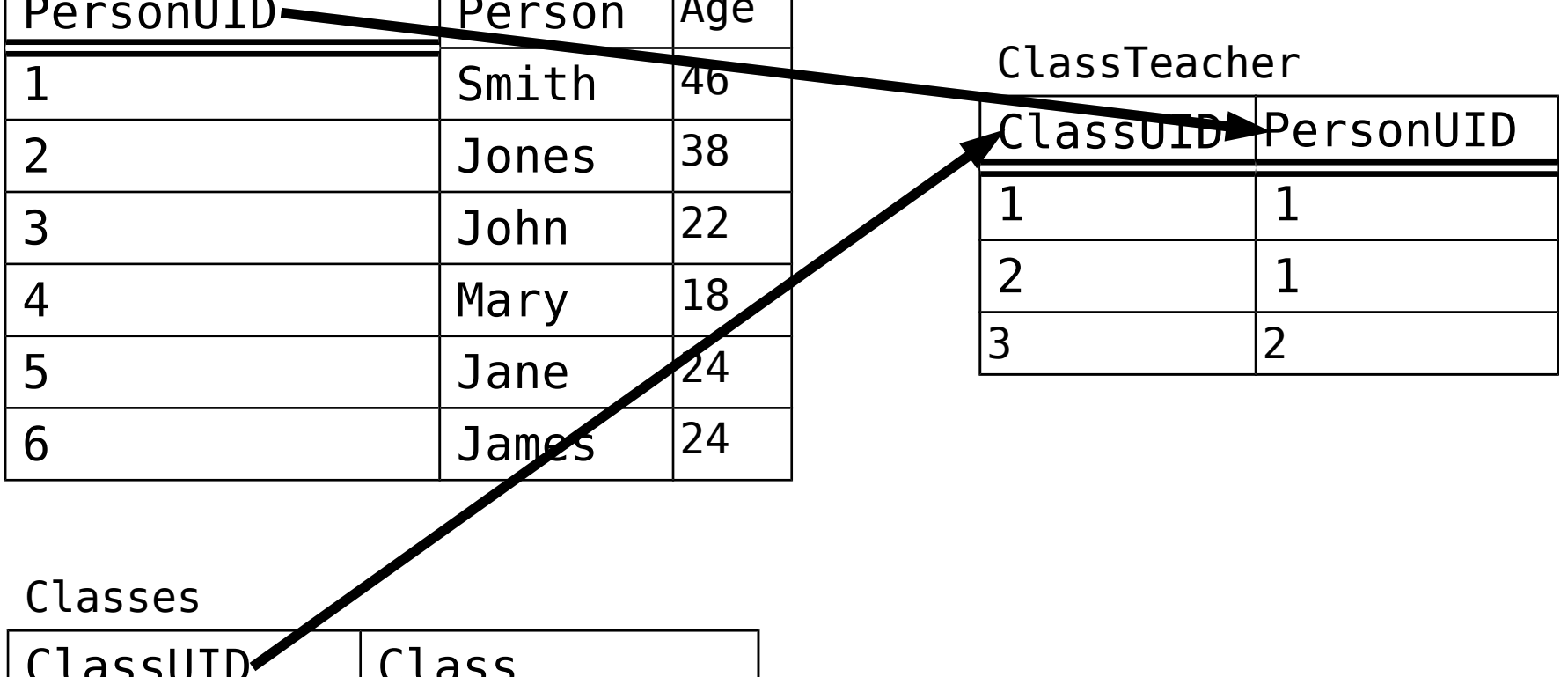
PersonUID	Person	Age
1	Smith	46
2	Jones	38
3	John	22
4	Mary	18
5	Jane	24
6	James	24

ClassTeacher

ClassUID	PersonUID
1	1
2	1
3	2

Classes

ClassUID	Class
1	Econ 101
2	Econ 201
3	Art Hist 101
4	Soc 101



Data Definition Language (DDL)

ClassTeacher

ClassUID	PersonUID
1	1
2	1
3	2

```
CREATE TABLE ClassTeacher (
classuid int REFERENCES Classes(classuid),
personuid int REFERENCES People(personuid),
PRIMARY KEY (classuid, personuid)
);
```

Data Manipulation Language (DML)

```
INSERT INTO People (person, birthday)
VALUES
   ('Smith', '1965-Jan-23'),
   ('Jones', '1973-Mar-14'),
   ('John', '1989-Aug-08');
```

Data Manipulation Language (DML)

```
INSERT INTO People (person, birthday)
VALUES
   ('Smith', '1965-Jan-23'),
   ('Jones', '1973-Mar-14'),
   ('John', '1989-Aug-08');

UPDATE People SET birthday = '1972-Oct-05'
WHERE name='Jones';
```

Data Manipulation Language (DML)

```
INSERT INTO People (person, birthday)
VALUES
   ('Smith', '1965-Jan-23'),
   ('Jones', '1973-Mar-14'),
   ('John', '1989-Aug-08');

UPDATE People SET birthday = '1972-Oct-05'
WHERE name='Jones';

DELETE FROM People
WHERE birthday >= '1985-Jan-01';
```

Pros/Cons of DDL & DML

- Relatively straightforward but bulky
- Easy to commit simple syntax errors
- Omitting DML's WHERE clause modifies the entire table

```
UPDATE People SET
   birthday = '1972-Oct-05';

DELETE FROM People;
```

- Consider using RDBMS's shell interface, ODBC connector, or other API

The Problem of Missing Data

- NULL marker indicates that data's value is “unknown,” but doesn't say why
- 2VL versus 3VL
  - Boolean (fuzzy-set) algebra
    True = 1.0,
    Unknown = 0.5
    False = 0.0
- NULL != NULL

The Problem of Missing Data

True = 1.0, Unk = 0.5, False = 0.0

The Problem of Missing Data

True = 1.0, Unk = 0.5, False = 0.0

```
SELECT True or True;
 ?column?
----------
 t
(1 row)
```

The Problem of Missing Data

True = 1.0, Unk = 0.5, False = 0.0

```
SELECT True or True;
 ?column?
----------
 t
(1 row)
```

```
SELECT True or NULL;
 ?column?
----------
 t
(1 row)
```

The Problem of Missing Data

True = 1.0, Unk = 0.5, False = 0.0

```
SELECT True or True;
 ?column?
----------
 t
(1 row)
```

```
SELECT True or NULL;
 ?column?
----------
 t
(1 row)
```

```
SELECT True or False;
 ?column?
----------
 t
(1 row)
```

The Problem of Missing Data

True = 1.0, Unk = 0.5, False = 0.0

```
SELECT True or True;
 ?column?
----------
 t
(1 row)
```

```
SELECT True or NULL;
 ?column?
----------
 t
(1 row)
```

```
SELECT True or False;
 ?column?
----------
 t
(1 row)
```

```
SELECT NULL or False;
 ?column?
----------

(1 row)
```

The Problem of Missing Data

NULL !=NULL

The Problem of Missing Data

NULL !=NULL

```
SELECT 1=1;
 ?column?
----------
 t
(1 row)
```

The Problem of Missing Data

NULL !=NULL

```
SELECT 1=1;
 ?column?
----------
 t
(1 row)
```

```
SELECT 1=2;
 ?column?
----------
 f
(1 row)
```

The Problem of Missing Data
NULL !=NULL

```
SELECT 1=1;
 ?column?
----------
 t
(1 row)
```

```
SELECT 1=2;
 ?column?
----------
 f
(1 row)
```

```
SELECT not(True)
 ?column?
----------
 f
(1 row)
```

The Problem of Missing Data

NULL !=NULL

```
SELECT not(NULL)
 ?column?
----------

(1 row)
```

The Problem of Missing Data
NULL !=NULL

```
SELECT not(NULL)
 ?column?
----------

(1 row)


SELECT NULL=NULL;
 ?column?
----------

(1 row)
```

The Problem of Missing Data
NULL !=NULL

```
SELECT not(NULL)
 ?column?
----------

(1 row)
```

```
SELECT NULL=NULL;
 ?column?
----------

(1 row)
```

```
SELECT NULL is null;
 ?column?
----------
 t
(1 row)
```

Dealing with Nulls

- Strategy 1: Avoidance
  - Set table columns non-nullable
  - Avoid outer joins
  - Normalize properly
  - But not feasible in practice

Dealing with Nulls

- Strategy 1: Avoidance
  - Set table columns non-nullable
  - Avoid outer joins
  - Normalize properly
  - But not feasible in practice
- Strategy 2: Learn SQL's 3VL