

Class 2 MySQL

Elements of Databases

Sept 3, 2021

Relational Data Model

- Database == Collection of relations
- Relation == A table with columns (attributes) and rows (tuples)
- Column properties: named, domain, unordered
- Row properties: single-valued attributes, unique, unordered

How do we enforce a unique row constraint?

- Referential integrity: Every non-null foreign key must match an existing primary key value.

Notation: `Customer(id, fname, lname, address ...)`

`Order(orderno, custid, date, channel ...)`

Simple Select Statements

```
SELECT c1, c2, c3, cn
FROM T1
WHERE c1 > 100 AND c2 = 'XYZ'
ORDER BY c3, c4;
```

```
SELECT c1, c2, c3, cn
FROM T1
WHERE c1 IS NOT NULL OR (c2 = 'XYZ' AND c3 = 'ABC')
ORDER BY c3 DESC;
```

Simple Create, Insert, Update, Delete Statements

```
CREATE TABLE T1 (c1 INT PRIMARY KEY,  
                 c2 VARCHAR(30) NOT NULL,  
                 c3 VARCHAR(30));
```

```
INSERT INTO T1 (c1, c2, c3) VALUES (1, 'Austin',  
  'TX');
```

```
UPDATE T1 SET c2 = 'New York City', c3 = 'NY'  
WHERE c1 = 1;
```

```
DELETE FROM T1 WHERE c3 IN ('NY' 'TX', 'CA');
```

MySQL Overview

- It's been around a long time
- Simple and easy-to-use
- Open-source software (commercialized by Oracle)
- Implements the relational model
- Designed for storing structured data
- Feature-rich SQL support
- Supports many languages
- Small to medium size data (< TB storage)
- Low to moderate QPS of reads and writes (10K)
- Scale reads through read replicas
- Scale writes through sharding (e.g. [Vitess](#))

Instapoll on today's setups

MySQL Guide:

<https://github.com/cs327e-fall2021/snippets/wiki/MySQL-Setup-Guide>

Jupyter Guide:

<https://github.com/cs327e-fall2021/snippets/wiki/Jupyter-Setup-Guide>

MySQL Code Lab:

- Clone [snippets](#) repo
- Open [mysql notebook](#)
- Create database
- Create tables
- Populate tables
- Check tables
- Remove header row
- Add primary keys
- Add foreign key
- Test foreign key

College Database Schema

Student		
PK	sid	CHAR
	fname	VARCHAR
	lname	VARCHAR
	dob	DATE
	status	CHAR

Class		
	sid	CHAR
	cno	CHAR
	cname	VARCHAR
	credits	INT
	grade	CHAR

Student(sid, fname, lname, dob, status)

Class(sid, cno, cname, credits, grade)

Instructor(tid, name, dept)

Teaches(tid, cno)

Instructor		
PK	tid	CHAR
	name	VARCHAR
	dept	VARCHAR

Teaches		
PK, FK	tid	CHAR
PK	cno	CHAR



Practice Problems

Who takes CS327E or CS329E?

Who takes CS327E and CS329E?

Student(sid, fname, lname, dob, status)

Class(sid, cno, cname, credits, grade)

Instructor(tid, name, dept)

Teaches(tid, cno)

Second Question

Who takes CS327E and CS329E?

Student(sid, fname, lname, dob, status)

Class(cno, cname, credits)

Instructor(tid, name, dept)

Teaches(tid, cno)

Is this query a correct implementation?

```
SELECT sid
FROM Class
WHERE cno = 'CS327E'
      AND cno = 'CS329E'
```

Relational Data Modeling

- Entity: A real-world object
- Usually a noun
- Common examples: Person, Team, Product, Order, Shipment

Analogies with OOP:

- Entity: analogous to class
- Record: analogous to objects
- Attribute: analogous to members of an object

Questions:

- How do we represent relationships between entities?
- Can entities have methods in addition to attributes?

Design Guidelines

1. A table represents a single entity type or a $m:n$ relationship.
2. The fields of a table represent the attributes of the entity type or attributes of a $m:n$ relationship.
3. Each field in a table is assigned a data type that best fits its domain of values.
4. Each table has a Primary Key (PK) constraint which is made up of one or more fields that uniquely represent each entity in that table.
5. 1:1 and 1:m relationships are represented as a Foreign Key (FK) relationship, in which the child table has a FK constraint on the field(s) that reference its parent's PK fields.

Back to our college schema:

Student		
PK	sid	CHAR
	fname	VARCHAR
	lname	VARCHAR
	dob	DATE
	status	CHAR

Class		
	sid	CHAR
	cno	CHAR
	cname	VARCHAR
	credits	INT
	grade	CHAR

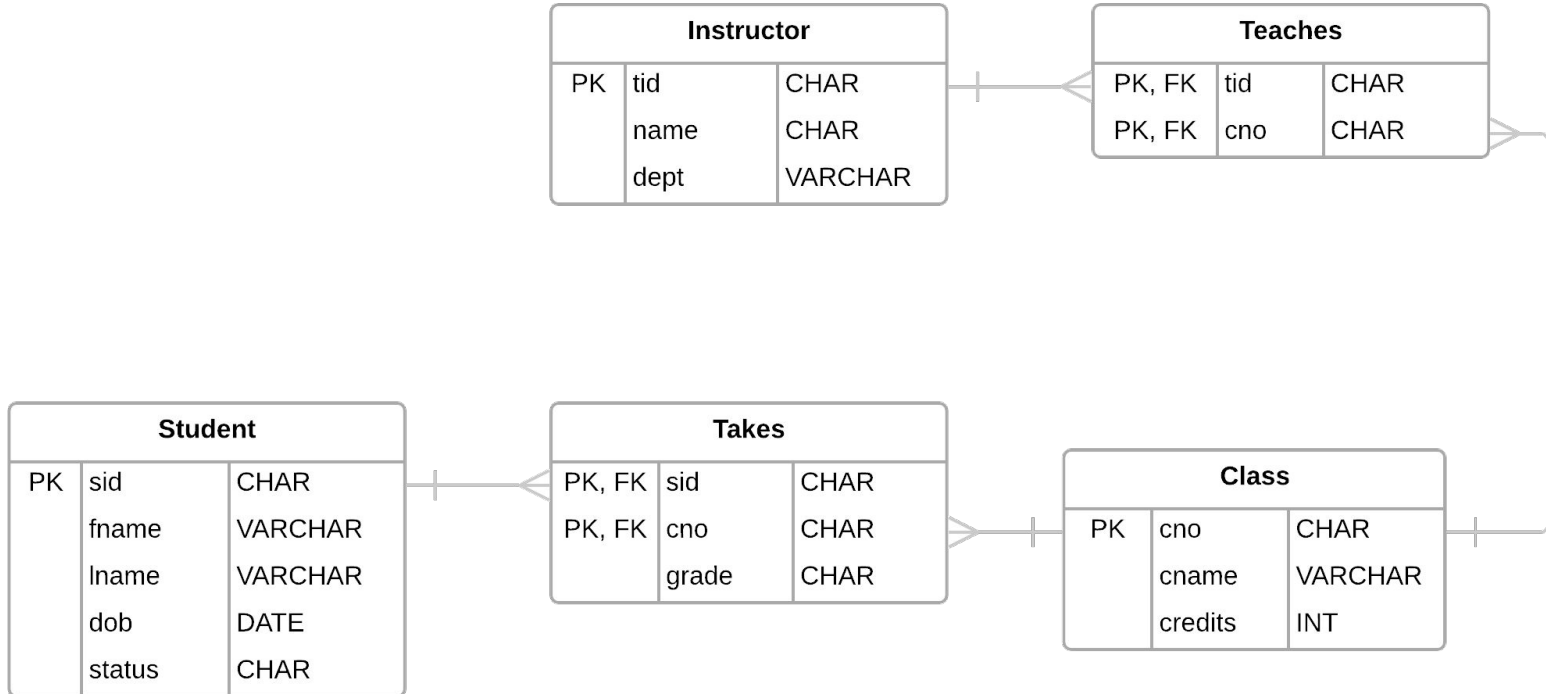
- Insert Anomaly
- Update Anomaly
- Delete Anomaly

Instructor		
PK	tid	CHAR
	name	VARCHAR
	dept	VARCHAR

Teaches		
PK, FK	tid	CHAR
PK	cno	CHAR



Remodeled college schema



Common Transforms

- `CREATE TABLE T2 AS SELECT a, b, c FROM T1;`
- `SELECT a, b, c FROM T1`
UNION [DISTINCT]
`SELECT x AS a, y AS b, z AS c FROM T2;`
- `SELECT a, b, c, 'some string' AS s FROM T1`
UNION ALL
`SELECT d, e, f, 'some string' AS s FROM T2;`

Project 1

<http://www.cs.utexas.edu/~scohen/projects/Project1.pdf>