CS143: Database Integrity

Professor Junghoo “John” Cho

What We Will Learn

- How can we ensure that data in our database is “consistent”?
  - Referential integrity constraint
  - CHECK constraint

Example Database: What is Wrong?

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	Ki#Bu!GK.$@q	19	2.1
301	Elaine	301 Wilshire	263	3.9
401	James	183 Westwood	17	-1.0
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	134	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
999	CS	143	01
401	AT	000	00
303	EE	143	01
303	CS	112	01

Addr, age, GPA, unit values are wrong
Sid is not unique
Enroll.sid $\not\subset$ Student.sid
Enroll.(dept, cnum, sec) $\not\subset$ Class.(dept, cnum, sec)

Data Integrity Enforcement in RDBMS

- Domain
  - GPA is REAL
  - NOT NULL
- Integrity constraints
  - If violated, DBMS generates an error and abort
  - e.g., Key, Referential integrity, CHECK

Key Constraint

- A set of attributes should be unique in a table

  Class(dept, cnum, sec, unit, instructor, title)
  Class(dept, cnum, sec, unit, instroctur, title)

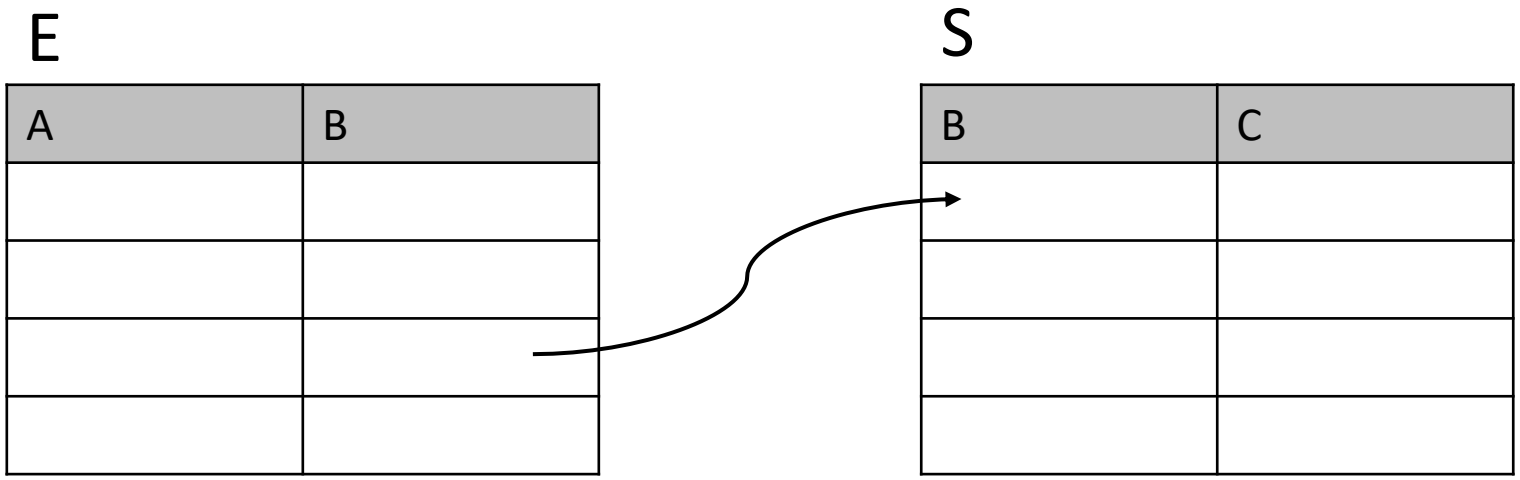
  CREATE TABLE Class(
    dept CHAR(2) NOT NULL, cnum INT NOT NULL, sec INT NOT NULL,
    unit INT, instructor VARCHAR(100), title VARCHAR(100),
    PRIMARY KEY(dept, cnum, sec),
    UNIQUE(dept, sec, title)
  );

- One PRIMARY KEY per table. Others should use UNIQUE.
  - PRIMARY KEY and UNIQUE are enforced through index (more on this later)

Referential Integrity (RI)

- Examples
  - If sid appears in Enroll, it should also appear in Student
  - If (dept, cnum, sec) appears in Enroll, it should also appear in Class
- Q: Is the reverse true?

Terminology

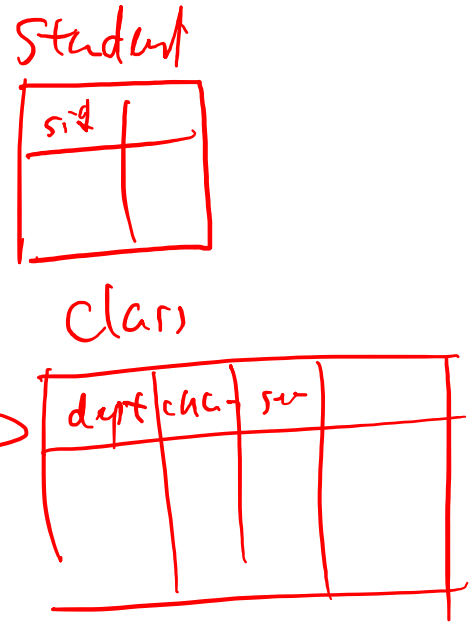


- E.B references S.B
  - E.B: foreign key (= referencing attribute)
  - S.B: referenced attribute
- Referential integrity
  - Referencing attribute should always exist in the referenced attribute
  - When foreign key is NULL, no referential integrity check is performed

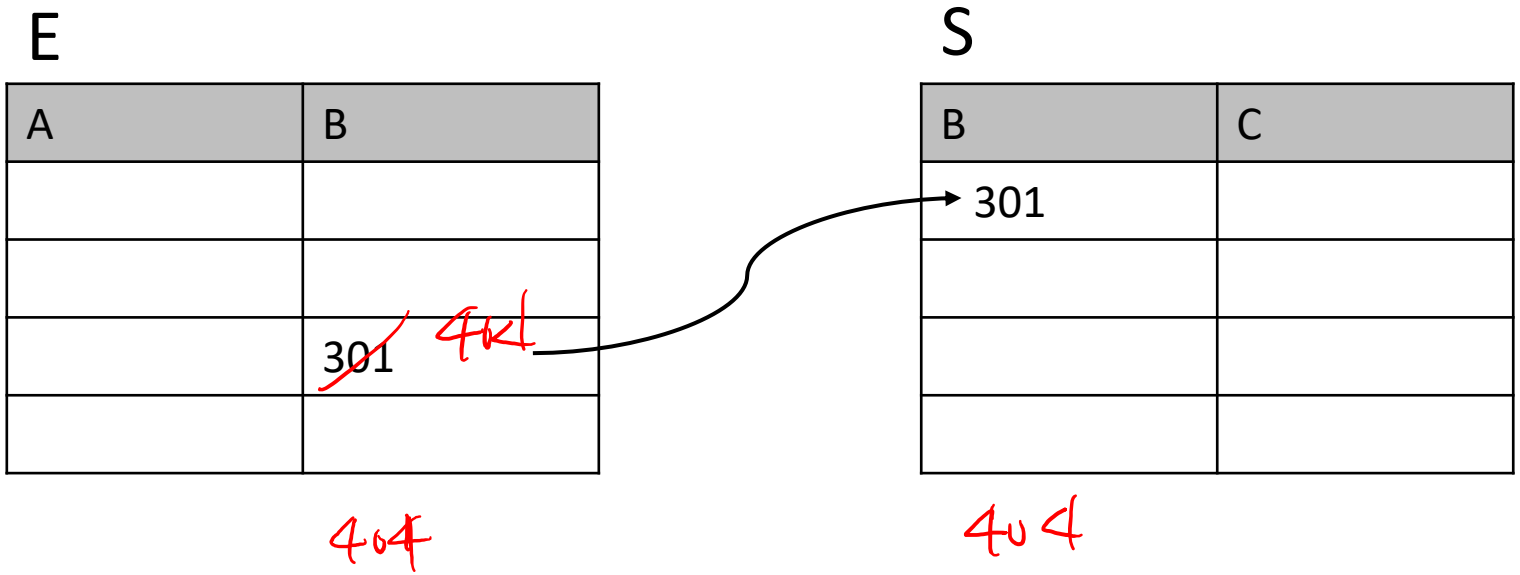
SQL Referential Integrity Syntax

- CREATE TABLE Enroll(
  sid INT,
  dept CHAR(2),
  cnum INT,
  sec INT,
  FOREIGN KEY(sid) REFRENCES Student(sid),
  FOREIGN KEY(dept,cnum,sec)
  REFERENCES Class(dept, cnum, sec)
  );
- Referenced attributes must be PRIMARY KEY or UNIQUE

  sid INT REFERENCES Student(sid)



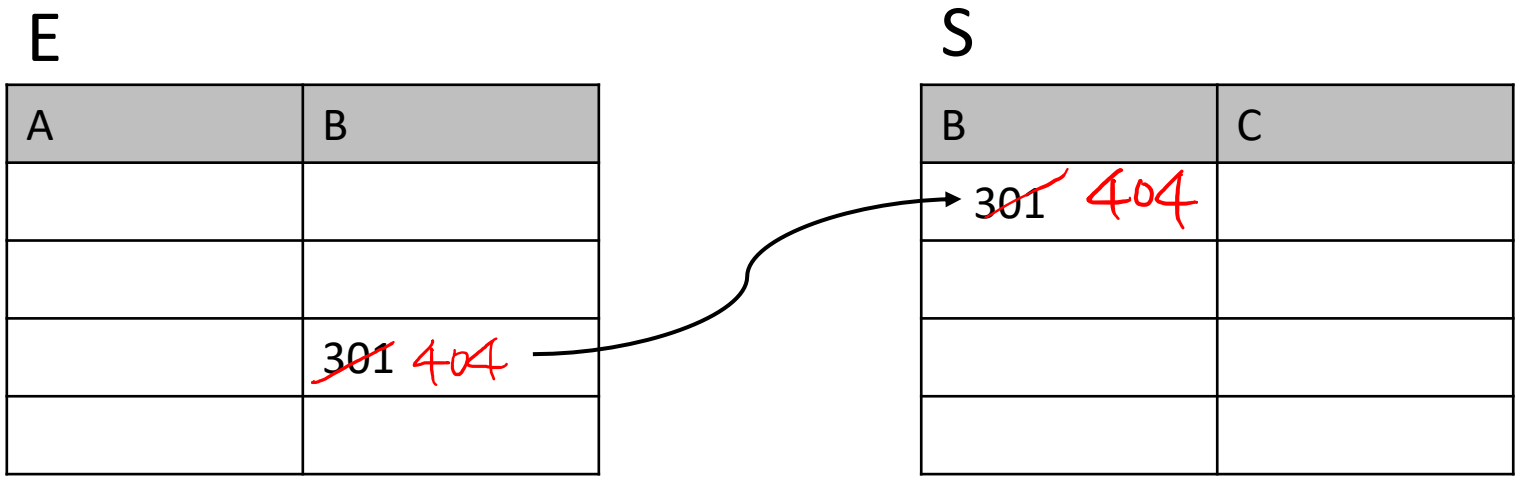
Violation of Referential Integrity



- Q: When can RI be violated?

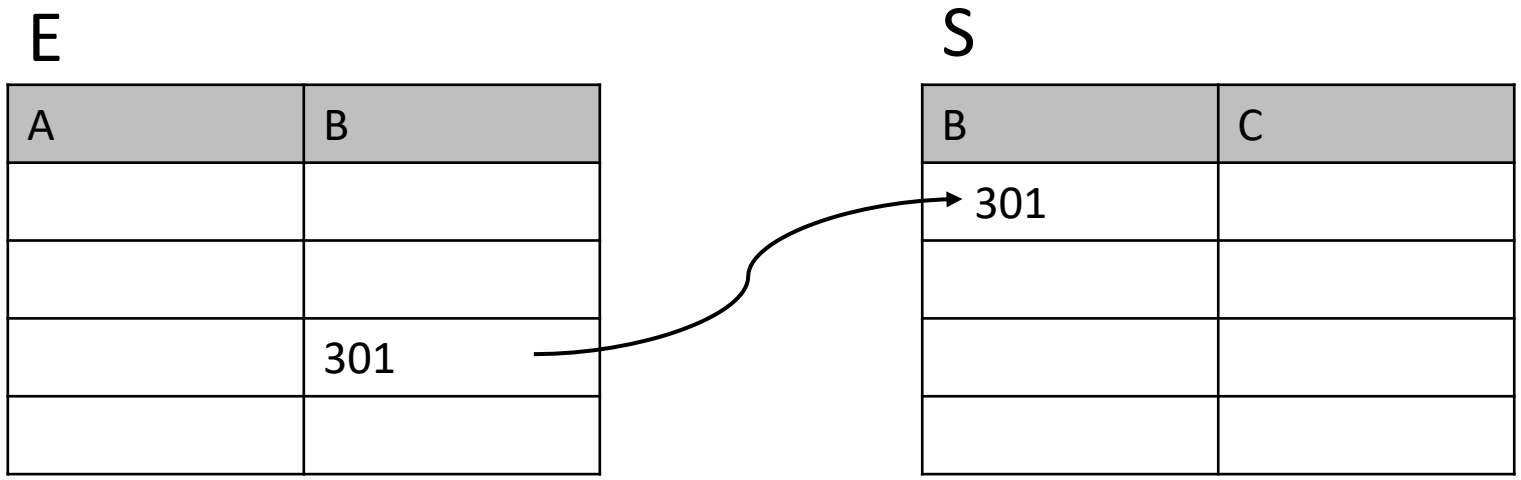
INSERT INTO E	INSERT INTO S
DELETE FROM E	DELETE FROM S
UPDATE E	UPDATE S

Violation of Referential Integrity



- RI violation from referencing table E is ***never allowed***
  - DBMS rejects the statement
- RI violation from referenced table S is not allowed by default, but we can instruct DBMS to "fix" the violation automatically
  - Q: DELETE FROM S. How to fix?
  - Q: UPDATE S. How to fix?

Specifying Automatic Fix of RI Violation

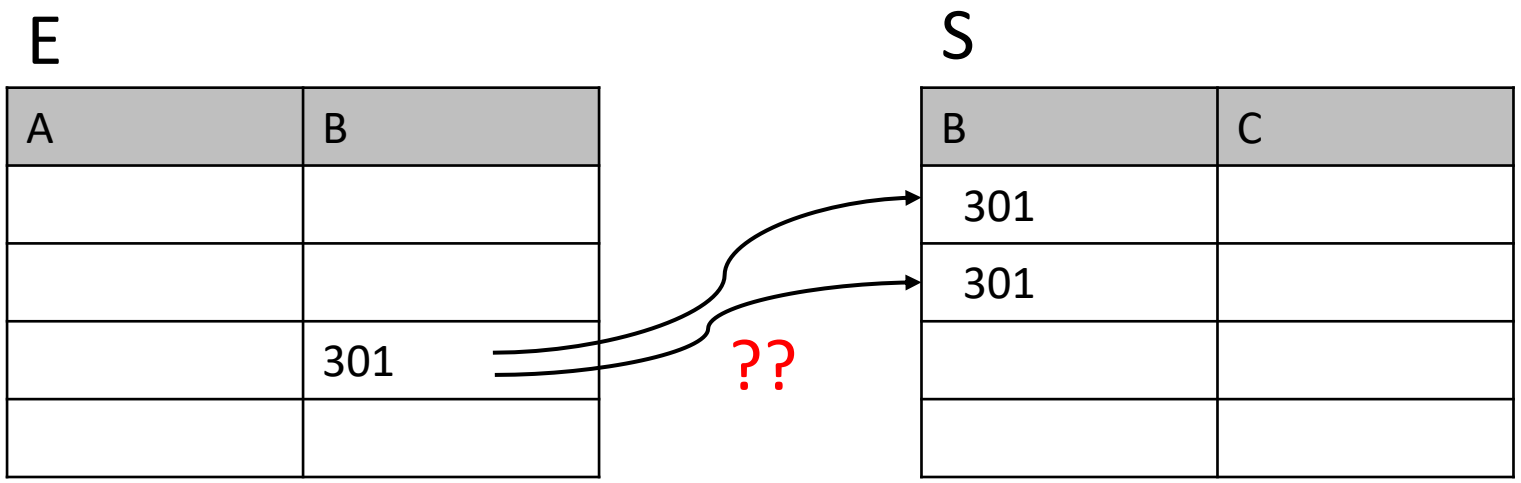


- Syntax

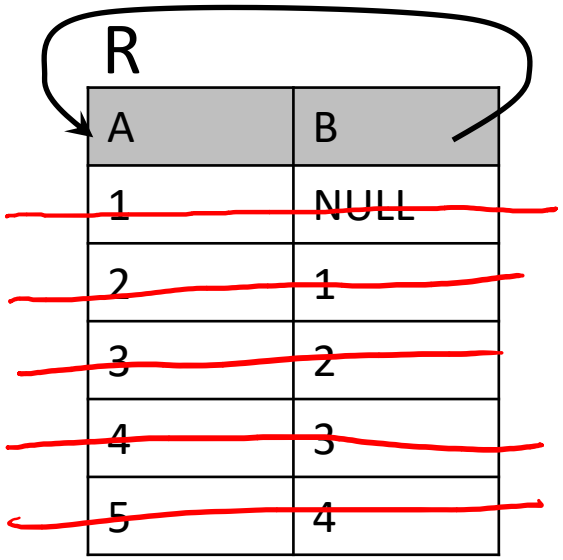
```
CREATE TABLE E(
    A INT, B INT,
    FOREIGN KEY(B) REFERENCES S(B)
    ON UPDATE { CASCADE | SET NULL | SET DEFAULT }
    ON DELETE { CASCADE | SET NULL | SET DEFAULT }
);
```

More Comments on Referential Integrity

- Referential integrity is the only SQL constraint that can “fix itself”
  - Other constraints simply reject the statement and generates an error
- Some DBMS do not support all “fixing” actions
  - Oracle supports ON DELETE but not ON UPDATE, for example
- Q: Why should referenced attributes be unique?



Self-Referencing Table



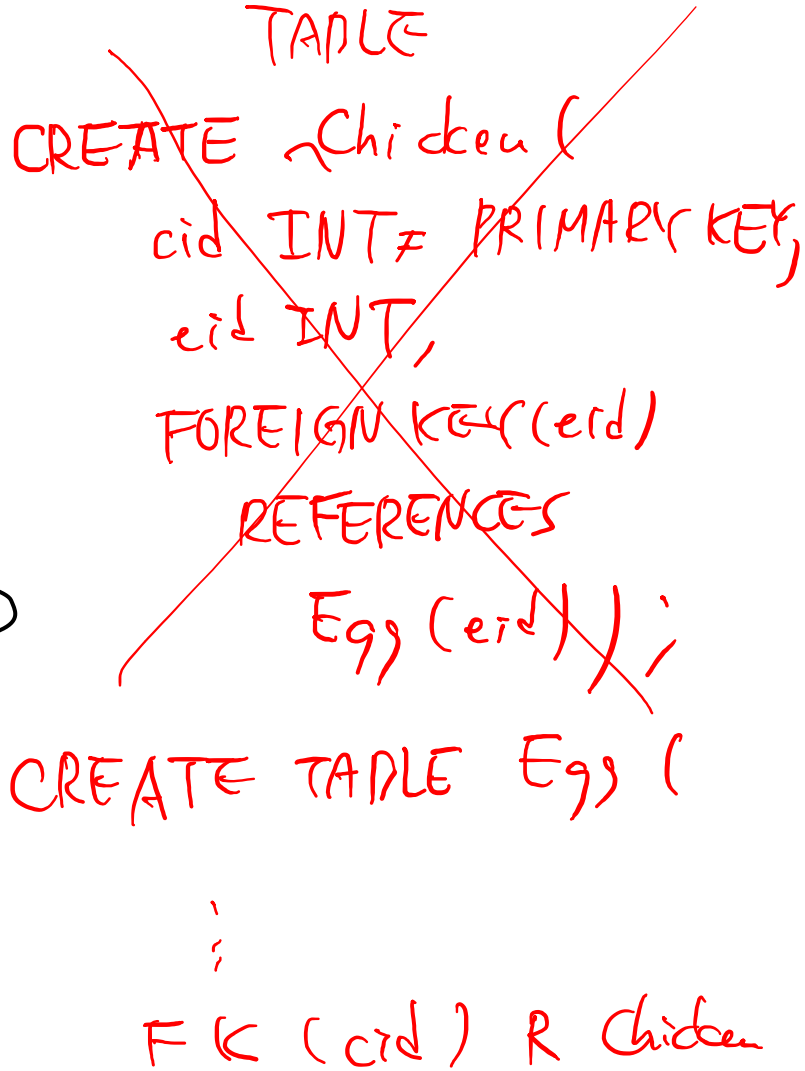
- CREATE TABLE R(
  A INT PRIMARY KEY,
  B INT,
  FOREIGN KEY(B) references R(A)
  ON DELETE CASCADE);
- Self-referencing table: A table that references itself
- Q: What will happen if we delete (1, NULL)?

DELETE FROM R WHERE A=1;

Circular Constraint

- ChickenFrom(<u>cid</u>, eid): eid became cid
  EggFrom(<u>eid</u>, cid): eid is born of cid
  (Chicken.cid ⊂ Egg.eid, Egg.cid ⊂ Chicken.cid)

- Q: Can we start with Chicken? Or Egg?
  - Q: How can we create the two tables?
  - Q: How can we insert tuples?



Cirular Constraint

- Creating Tables: ALTER TABLE

  CREATE TABLE Chicken(cid INT PRIMARY KEY, eid INT);

  CREATE TABLE Egg(eid INT PRIMARY KEY, cid INT REFERENCES Chicken);

  ALTER TABLE Chicken ADD FOREIGN KEY(eid) REFERENCES Egg(eid);

- Inserting Tuples: Two options

  1. Create the “first chicken” (or egg) that came from nowhere (= NULL)
  2. Create a chicken (and egg) that came from itself

     INSERT INTO Chicken VALUES (1, NULL);

     INSERT INTO Egg VALUES (1, 1);

     UPDATE Chicken SET eid = 1 WHERE eid IS NULL;

CHECK Constraint

- Example: GPA should be between 0.0 and 4.0

```
CREATE TABLE Student(
    sid INT, name VARCHAR(50), addr VARCHAR(50), GPA REAL,
    CHECK(GPA>=0 AND GPA<=4)
);
```

- CHECK(*condition*)
  - Condition can be any condition that may appear in WHERE clause
    - May include subqueries
- Constraint is ***attached to a particular table***
  - Constraint is checked ***when the attached table is updated***, and the statement is rejected if the condition is violated

C1: class cnum should be < 600 and class units should be < 10

- CREATE TABLE Class(
  dept CHAR(2), cnum INT, sec INT, unit INT,
  title VARCHAR(100), instructor VARCHAR(100),
  CHECK(cnum < 600 AND unit < 10)
  );

C2: The units of all CS classes should be > 3

- CREATE TABLE Class(
  dept CHAR(2), cnum INT, sec INT, unit INT,
  title VARCHAR(100), instructor VARCHAR(100),
  CHECK(dept <> 'CS' OR unit > 3)
  );

$A \rightarrow B \equiv \neg A \vee B$

dept = 'CS' → unit > 3

A B

≡ dept <> 'CS' OR unit > 3

C3: Students whose GPA < 2 cannot take CS class

- CREATE TABLE Student(
  sid INT, name VARCHAR(50), GPA REAL, …
  CHECK(GPA >= 2 OR sid NOT IN (
  SELECT sid
  FROM Enroll
  WHERE dept = 'CS'))
  );
  CREATE TABLE Enroll(
  sid INT, dept CHAR(2), cnum INT, sec INT,
  CHECK (dept <> 'CS' OR sid IN
  (SELECT sid
  FROM Student
  WHERE GPA >= 2))
  );
- Q: When will the constraint be checked?

dept = 'CS' → GPA >= 2

≡ dept <> 'CS' OR GPA >= 2

sid IN (SELECT sid
FROM Student
WHERE GPA >= 2);

$A \rightarrow B \equiv \neg A \vee B$

C4: Can we express referential integrity constraint using CHECK constraint?

- For example, can we express Enroll.sid ⊂ Student.sid?

```
CREATE TABLE Student(
  sid INT, name VARCHAR(50), GPA REAL, …



);
CREATE TABLE Enroll(
  sid INT, dept CHAR(2), cnum INT, sec INT,
  CHECK( sid IN (SELECT sid FROM Student ))

);
```

- Q: Are they really equivalent?

MySQL Support

- Domain constraint
- Key constraint
- Under InnoDB engine, referential integrity constraint
  - But not under MyISAM engine
  - Does not support single-column REFERENCES shorthand
  - Cannot omit column names even if names are the same
- No CHECK constraint support in standard MySQL
  - MariaDB 10.2.1 supports (limited) CHECK constraint
- WARNING: MySQL silently ignores constraints that it does not support
  - No warning or error messages
  - You may believe the constraint is there when it is not!
  - It is safer to use the most conservative syntax (even for lazy people like me)

What We Learned

- How to preserve database integrity
- Key constraint: PRIMARY KEY, UNIQUE
- Referential integrity constraint
  - FOREGIN KEY
  - Referenced attributes should be unique
  - Violation from referenced table may be fixed by DBMS
    - ON DELETE/UPDATE CASCADE/SET NULL
- CHECK constraint