CS 327E Class 4

Feb 17, 2020

1) An entity type is defined as _____

A. A particular occurence of an entity
B. A collection of entities that share similar characteristics
C. A property of an entity, such as a name or address
D. None of the above

2) Which of the following are **not** examples of an attribute type?

A. First Name, Middle Initial, Last Name
B. Street, City, State, Zip code
C. '123 Main Street' and '456 Avenue A'
D. Product Identifier, Product Name, Product Category

3) Which of the following concept is not included in the ER model / ERD?

A. Attribute Type
B. Entity Type
C. Relationship Type
D. None of the above

4) The relationship between the *Person* and *Movie* entity types has a cardinality of _____?

A. M:N
B. 1:M
C. 1:1
D. M:1

Person

id	name	age
1	Robert De Niro	76
2	Martin Scorsece	77
3	Greta Gerwig	36
4	Scarlett Johansson	35

Movie

id	title	year
a	The Irishman	2019
b	Jojo Rabbit	2019
c	Marriage Story	2019
d	Little Women	2019

Cast_Crew

person	movie	role
4	c	Actor
4	b	Actor
2	a	Director
1	a	Actor

5) How many joins are required in order to find the cast members from 'The Irishman' and return their names and ages?

A. 1
B. 2
C. 3

Person

id	name	age
1	Robert De Niro	76
2	Martin Scorsece	77
3	Greta Gerwig	36
4	Scarlett Johansson	35

Movie

id	title	year
a	The Irishman	2019
b	Jojo Rabbit	2019
c	Marriage Story	2019
d	Little Women	2019

Cast_Crew

person	movie	role
4	c	Actor
4	b	Actor
2	a	Director
1	a	Actor

Relational Data Model Terminology

- Entity: An object or a thing
- Usually a noun
- Common Examples: Person, Team, Product, Order, Shipment

Analogies with OOP:

- Entity: analogous to Object
- Entity Type: analogous to Class

Questions:

- What are the edges of a Entity Type?
- How do we decide what to store and not store in the database?

Relational Data Model Design Principles

P1. A table models one Entity Type and an Entity Type is modeled by one table.

P2. The set of fields of a table represent the attribute types of an entity.

P3. Each field is assigned a primitive type that best fits its domain of values.

P4. Each table has a Primary Key (PK) which is made up of one or more fields that uniquely represent each entity.

P5. A child table has a Foreign Key (FK) that points to its parent's PK.

P6. A *M:N* relationship is modeled by one junction table.

Design Principles Applied to College Dataset:

How many violations can you find?

College Staging ERD

college_staging.Classes		
	tid	String
	instructor	String
	dept	String
	cno	String
	cname	String
	credits	Integer

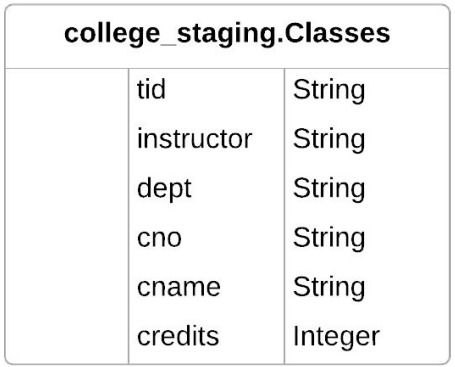
college_staging.Current_Student		
	sid	String
	fname	String
	lname	String
	dob	String
	cno	String
	cname	String
	credits	Integer
	grade	String

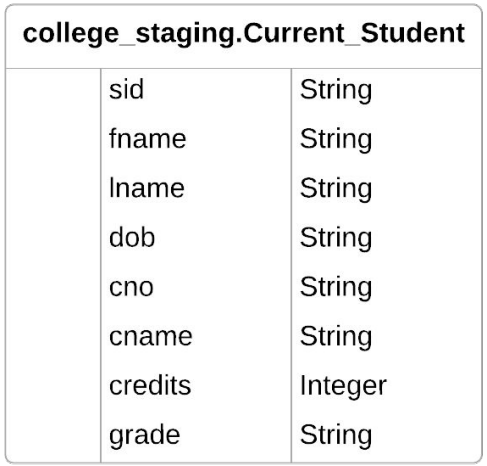
college_staging.New_Student		
PK	sid	String
	fname	String
	lname	String
	dob	Date

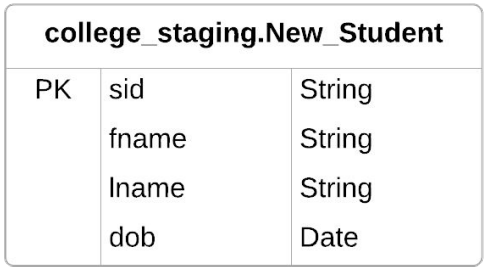
Design Principles Applied to College Dataset:

What can go wrong: data anomalies

College Staging ERD

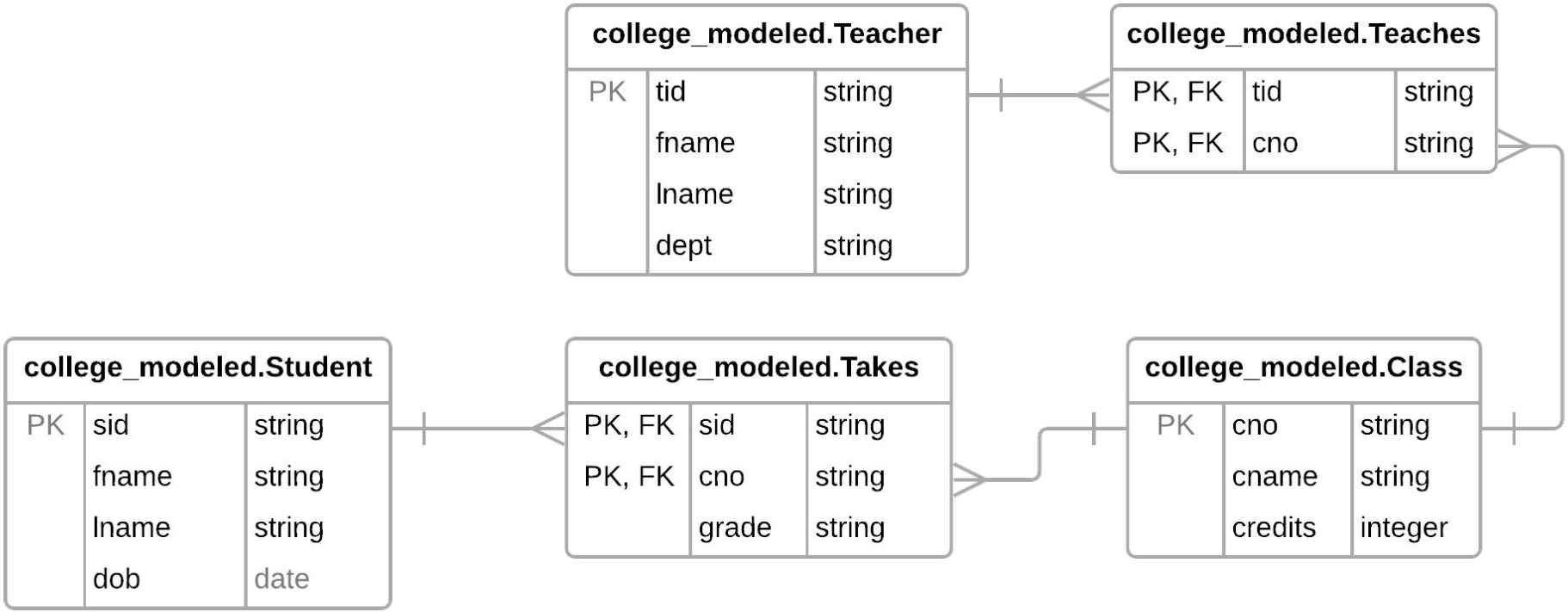






- Insert Anomaly
- Update Anomaly
- Delete Anomaly

College Modeled ERD



Common SQL Transforms

- `CREATE TABLE T2 AS SELECT a, b, c FROM T1`

- ```
  SELECT a, b, c FROM T1
  UNION ALL
  SELECT x AS a, y AS b, z AS c FROM T2
  ```

- ```
  SELECT a, b, c, 'some string' AS s FROM T1
  UNION DISTINCT
  SELECT d, e, f, 'some string' AS s FROM T2
  ```

Common SQL Transforms

- `SELECT GENERATE_UUID() AS uuid ...`
- `SELECT ROW_NUMBER() OVER() AS row_num ...`

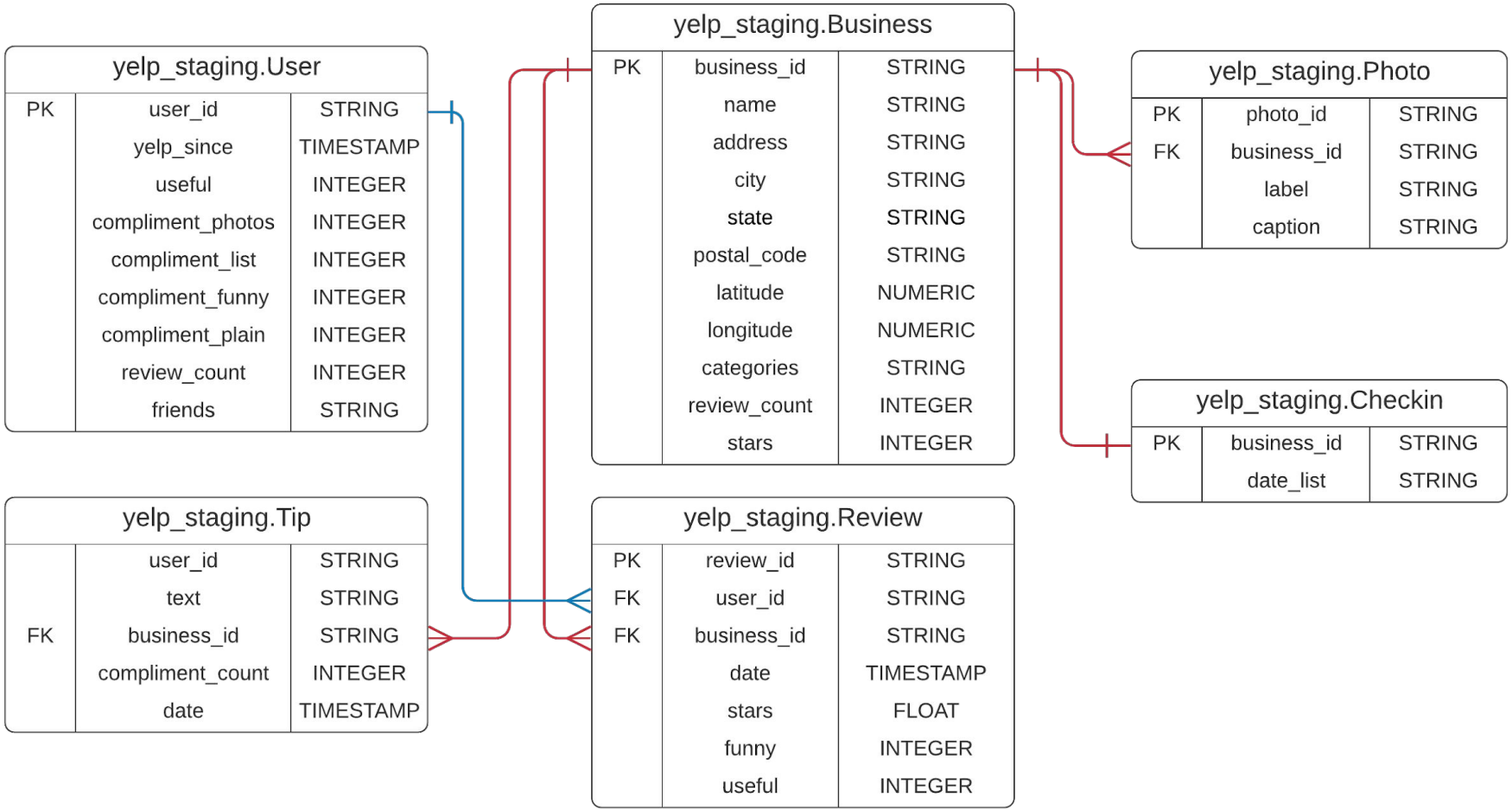
- `SELECT CAST('2020-02-17' AS DATE) AS new_date ...`
- `SELECT SAFE_CAST('02-17-2020' AS DATE) AS new_date ...`

- `SELECT CAST(SUBSTR('1000-00', 0, 4) AS INT64) AS number ...`

See documentation for more details and additional functions

Data Modeling Demo

Yelp Modeled ERD



Practice Problem

Construct a SQL query that finds all Takes records which violate referential integrity with its parent table Class.

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

Class(cno, cname, credits)

Teacher(tid, instructor, dept)

Takes(sid, cno, grade)

Teaches(tid, cno)

iClicker Question

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

Class(cno, cname, credits)

Teacher(tid, instructor, dept)

Takes(sid, cno, grade)

Teaches(tid, cno)

Construct a SQL query that finds all Takes records which violate referential integrity with its parent table Class.

What type of join is needed by this query?

A. Inner join
B. Outer join
C. Self join

Normal Forms

1NF: A database schema is in 1NF *iff* all attributes have scalar values.

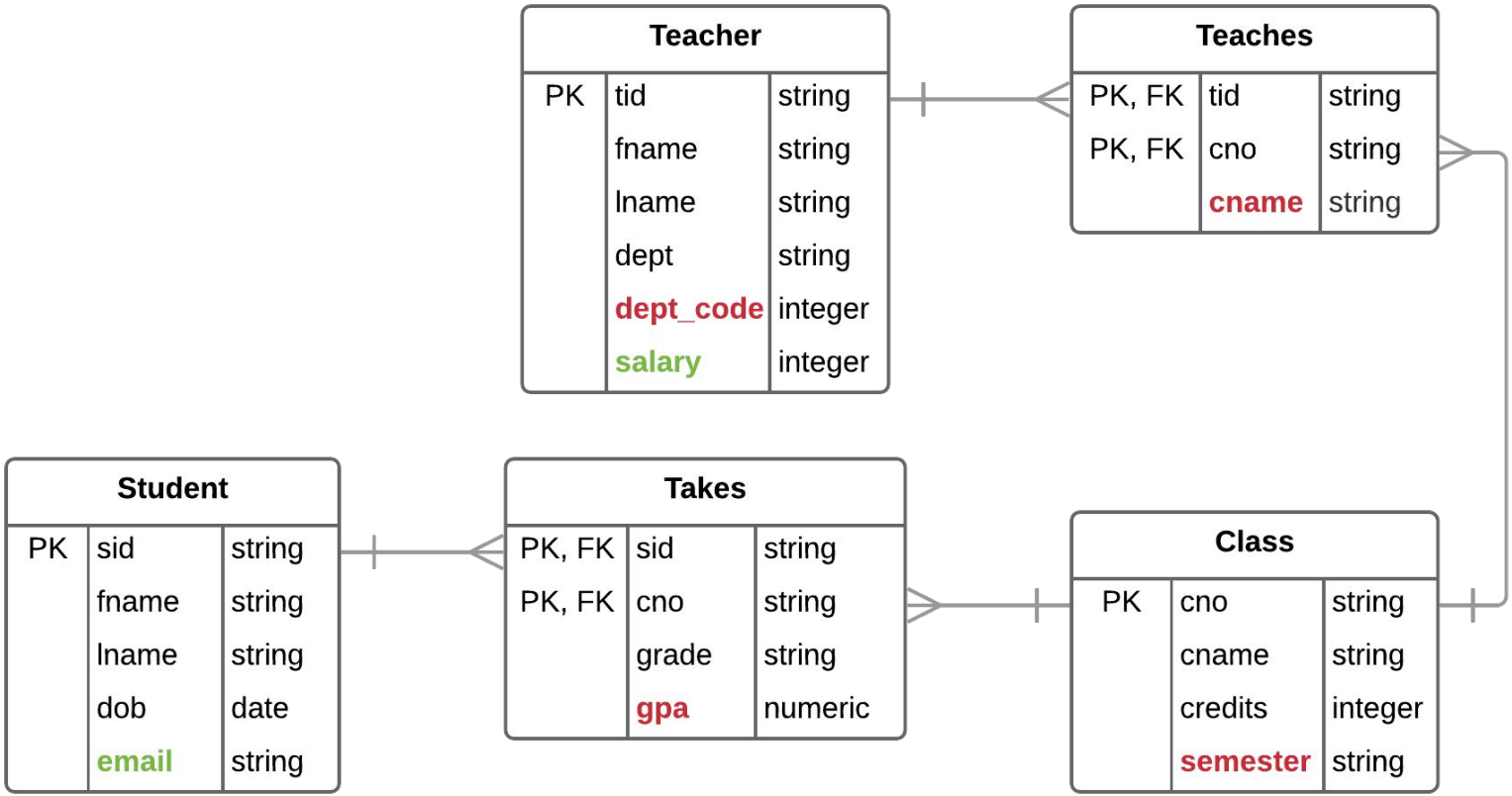
2NF: 1NF + all non-key attributes must be *functionally determined* by the *entire* primary key.

3NF: 2NF + all non-key attributes must be *functionally determined* by *only* the primary key.

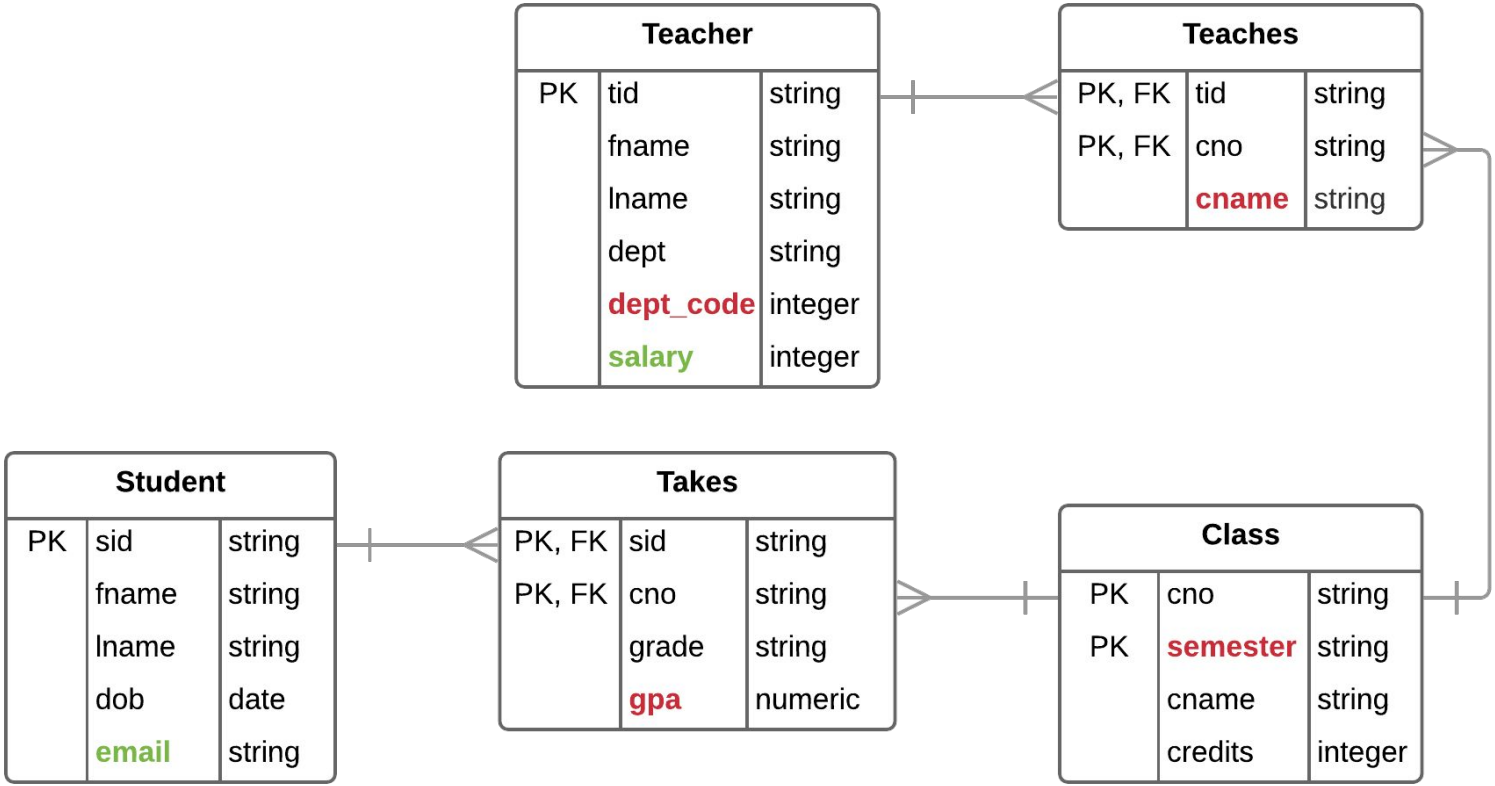
Functional Dependencies:
If two records agree on the attributes $A_1, A_2, \ldots, A_n$ then they must also agree on the attributes $B_1, B_2, \ldots, B_n$

Formally:
$A_1, A_2, \ldots, A_n \rightarrow B_1, B_2, \ldots, B_n$

Normal Form Violations

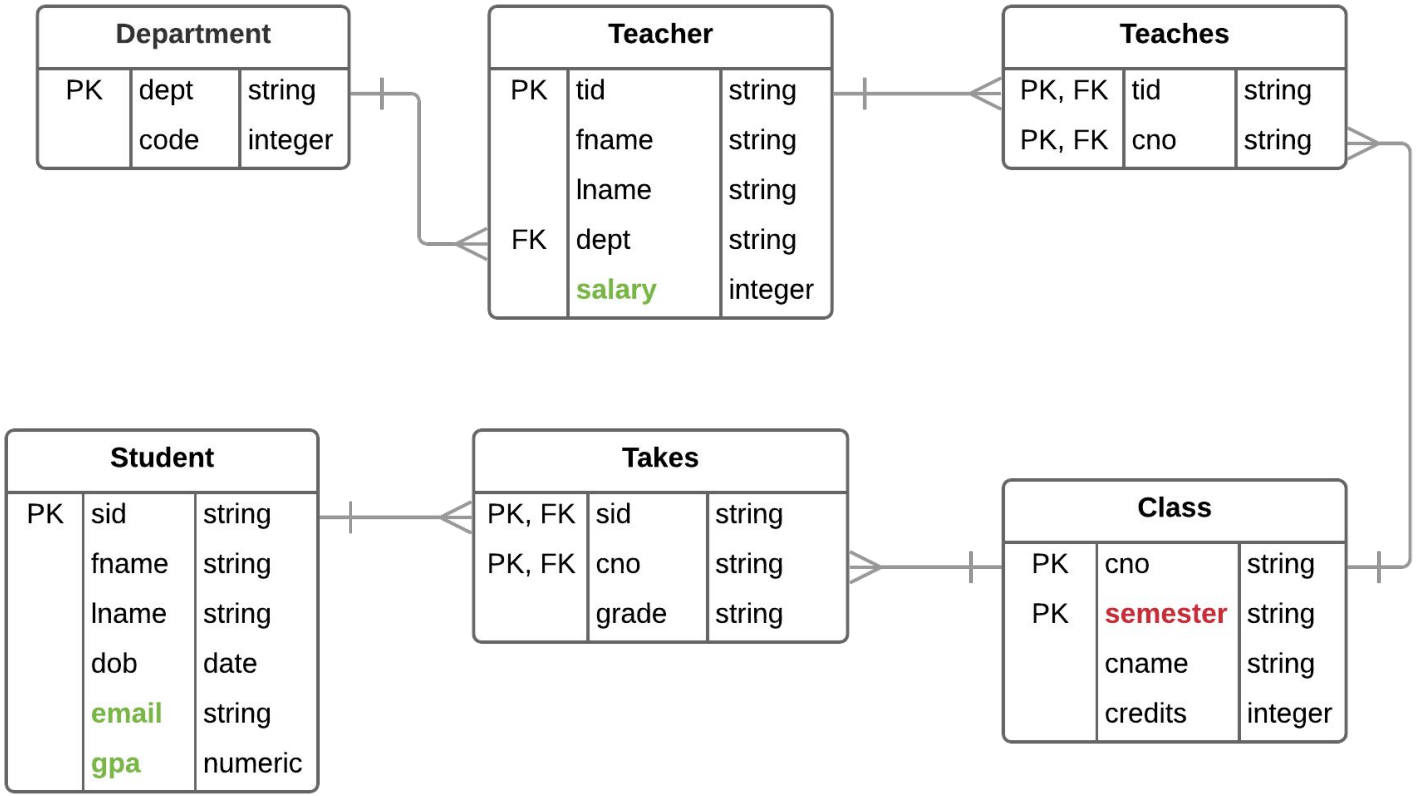


Normal Form Violations



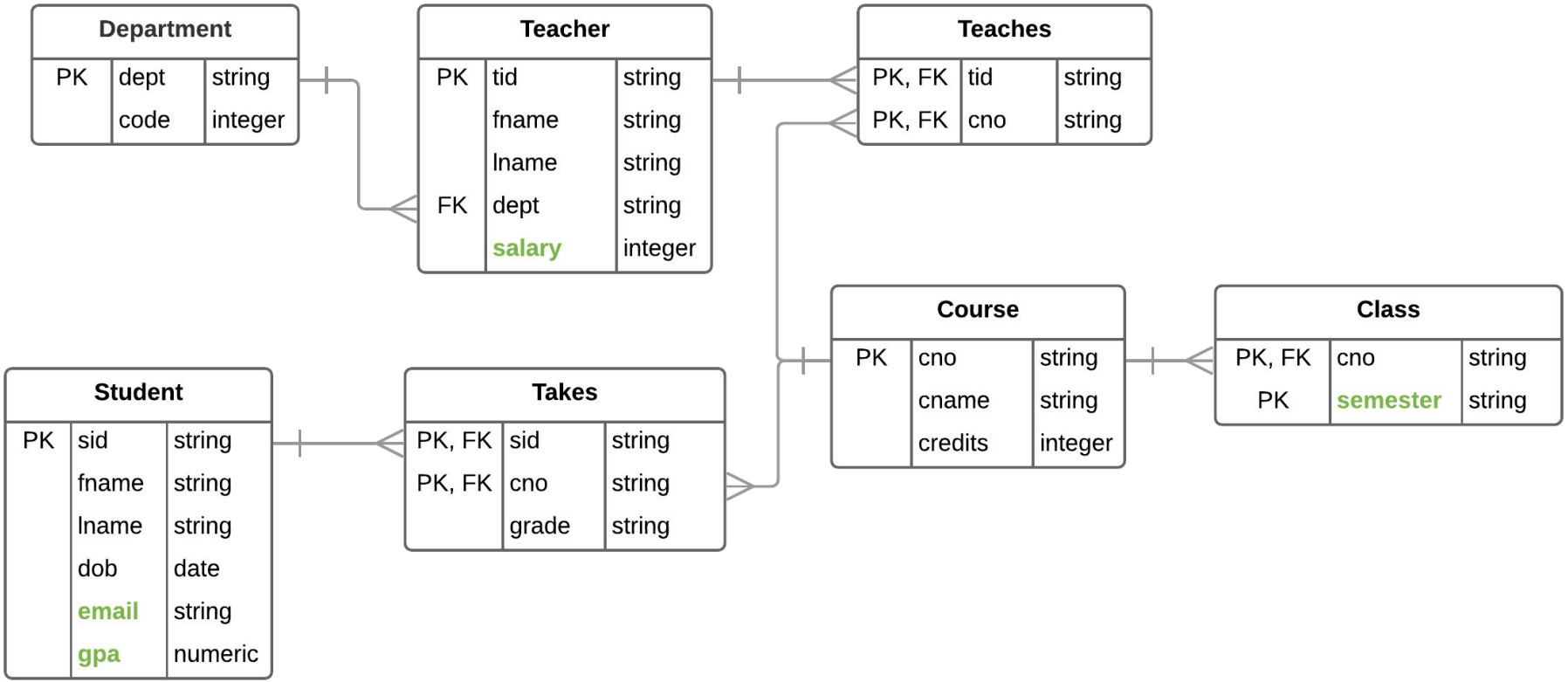
Practice Problem

Model the semester of a Class without violating normal form

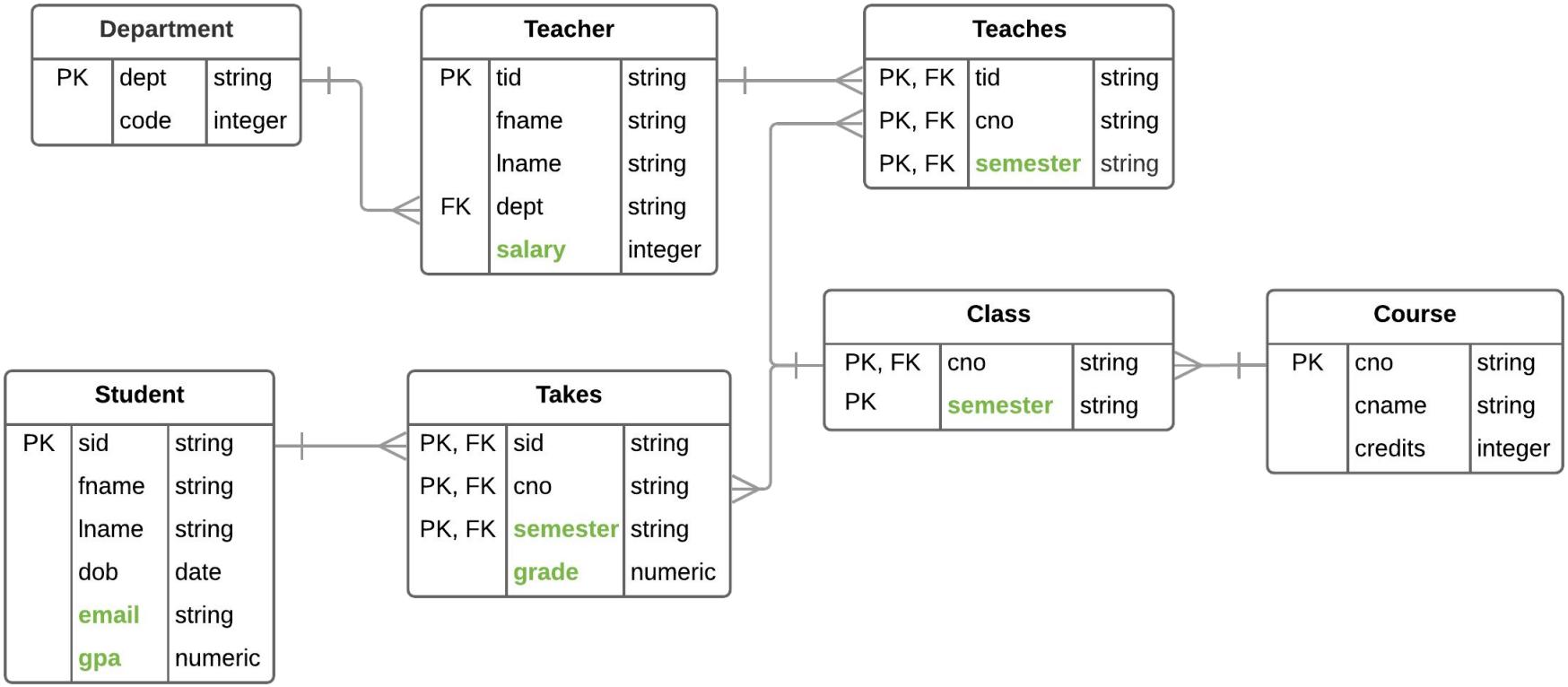


iClicker Question

Is this model correct?
A. Yes B. No



Normalized College Dataset



Milestone 4

http://www.cs.utexas.edu/~scohen/milestones/Milestone4.pdf

Appendix: H1B Case Study

Table Details: H1B_Applications_2017

Schema | Details | Preview

case_number	STRING	NULLABLE
visa_class	STRING	NULLABLE
case_status	STRING	NULLABLE
employer_name	STRING	NULLABLE
employer_business_dba	STRING	NULLABLE
employer_address	STRING	NULLABLE
employer_city	STRING	NULLABLE
employer_state	STRING	NULLABLE
employer_postal_code	STRING	NULLABLE
employer_country	STRING	NULLABLE
employer_province	STRING	NULLABLE
employer_phone	STRING	NULLABLE
employer_phone_ext	STRING	NULLABLE
naics_code	STRING	NULLABLE
soc_name	STRING	NULLABLE
soc_code	STRING	NULLABLE
job_title	STRING	NULLABLE
total_workers	INTEGER	NULLABLE
case_submitted	TIMESTAMP	NULLABLE
decision_date	TIMESTAMP	NULLABLE
employment_start_date	TIMESTAMP	NULLABLE
employment_end_date	TIMESTAMP	NULLABLE
full_time_position	BOOLEAN	NULLABLE
prevailing_wage	FLOAT	NULLABLE
pw_unit_of_pay	STRING	NULLABLE
wage_rate_of_pay_from	FLOAT	NULLABLE
wage_rate_of_pay_to	FLOAT	NULLABLE
wage_unit_of_pay	STRING	NULLABLE
worksite_city	STRING	NULLABLE
worksite_county	STRING	NULLABLE
worksite_state	STRING	NULLABLE
worksite_postal_code	STRING	NULLABLE
agent_attorney_name	STRING	NULLABLE
agent_representing_employer	BOOLEAN	NULLABLE
agent_attorney_city	STRING	NULLABLE
agent_attorney_state	STRING	NULLABLE
h1b_dependent	BOOLEAN	NULLABLE
willful_violator	BOOLEAN	NULLABLE
original_cert_date	TIMESTAMP	NULLABLE
new_employment	FLOAT	NULLABLE
continued_employment	FLOAT	NULLABLE
change_previous_employment	FLOAT	NULLABLE
new_concurrent_employment	FLOAT	NULLABLE
change_employer	FLOAT	NULLABLE
amended_petition	FLOAT	NULLABLE
pw_wage_level	STRING	NULLABLE
pw_source	STRING	NULLABLE
pw_source_year	STRING	NULLABLE
pw_source_other	STRING	NULLABLE
support_h1b	STRING	NULLABLE
labor_con_agree	BOOLEAN	NULLABLE
public_disclosure_location	STRING	NULLABLE

Step 1: load CSV files into staging area in BQ as separate tables.

Table Details:

2015 table: 241 MB size, 618,804 rows
2016 table: 233 MB size, 647,852 rows
2017 table: 253 MB size, 624,650 rows
2018 table: 283 MB size, 654,162 rows

Table Details: H1B_Applications_2017

Schema | Details | Preview

case_number	STRING	NULLABLE
visa_class	STRING	NULLABLE
case_status	STRING	NULLABLE
employer_name	STRING	NULLABLE
employer_business_dba	STRING	NULLABLE
employer_address	STRING	NULLABLE
employer_city	STRING	NULLABLE
employer_state	STRING	NULLABLE
employer_postal_code	STRING	NULLABLE
employer_country	STRING	NULLABLE
employer_province	STRING	NULLABLE
employer_phone	STRING	NULLABLE
employer_phone_ext	STRING	NULLABLE
naics_code	STRING	NULLABLE
soc_name	STRING	NULLABLE
soc_code	STRING	NULLABLE
job_title	STRING	NULLABLE
total_workers	INTEGER	NULLABLE
case_submitted	TIMESTAMP	NULLABLE
decision_date	TIMESTAMP	NULLABLE
employment_start_date	TIMESTAMP	NULLABLE
employment_end_date	TIMESTAMP	NULLABLE
full_time_position	BOOLEAN	NULLABLE
prevailing_wage	FLOAT	NULLABLE
pw_unit_of_pay	STRING	NULLABLE
wage_rate_of_pay_from	FLOAT	NULLABLE
wage_rate_of_pay_to	FLOAT	NULLABLE
wage_unit_of_pay	STRING	NULLABLE
worksite_city	STRING	NULLABLE
worksite_county	STRING	NULLABLE
worksite_state	STRING	NULLABLE
worksite_postal_code	STRING	NULLABLE
agent_attorney_name	STRING	NULLABLE
agent_representing_employer	BOOLEAN	NULLABLE
agent_attorney_city	STRING	NULLABLE
agent_attorney_state	STRING	NULLABLE
h1b_dependent	BOOLEAN	NULLABLE
willful_violator	BOOLEAN	NULLABLE
original_cert_date	TIMESTAMP	NULLABLE
new_employment	FLOAT	NULLABLE
continued_employment	FLOAT	NULLABLE
change_previous_employment	FLOAT	NULLABLE
new_concurrent_employment	FLOAT	NULLABLE
change_employer	FLOAT	NULLABLE
amended_petition	FLOAT	NULLABLE
pw_wage_level	STRING	NULLABLE
pw_source	STRING	NULLABLE
pw_source_year	STRING	NULLABLE
pw_source_other	STRING	NULLABLE
support_h1b	STRING	NULLABLE
labor_con_agree	BOOLEAN	NULLABLE
public_disclosure_location	STRING	NULLABLE

Step 2:

- read the documentation on your dataset (file descriptions and individual field descriptions).

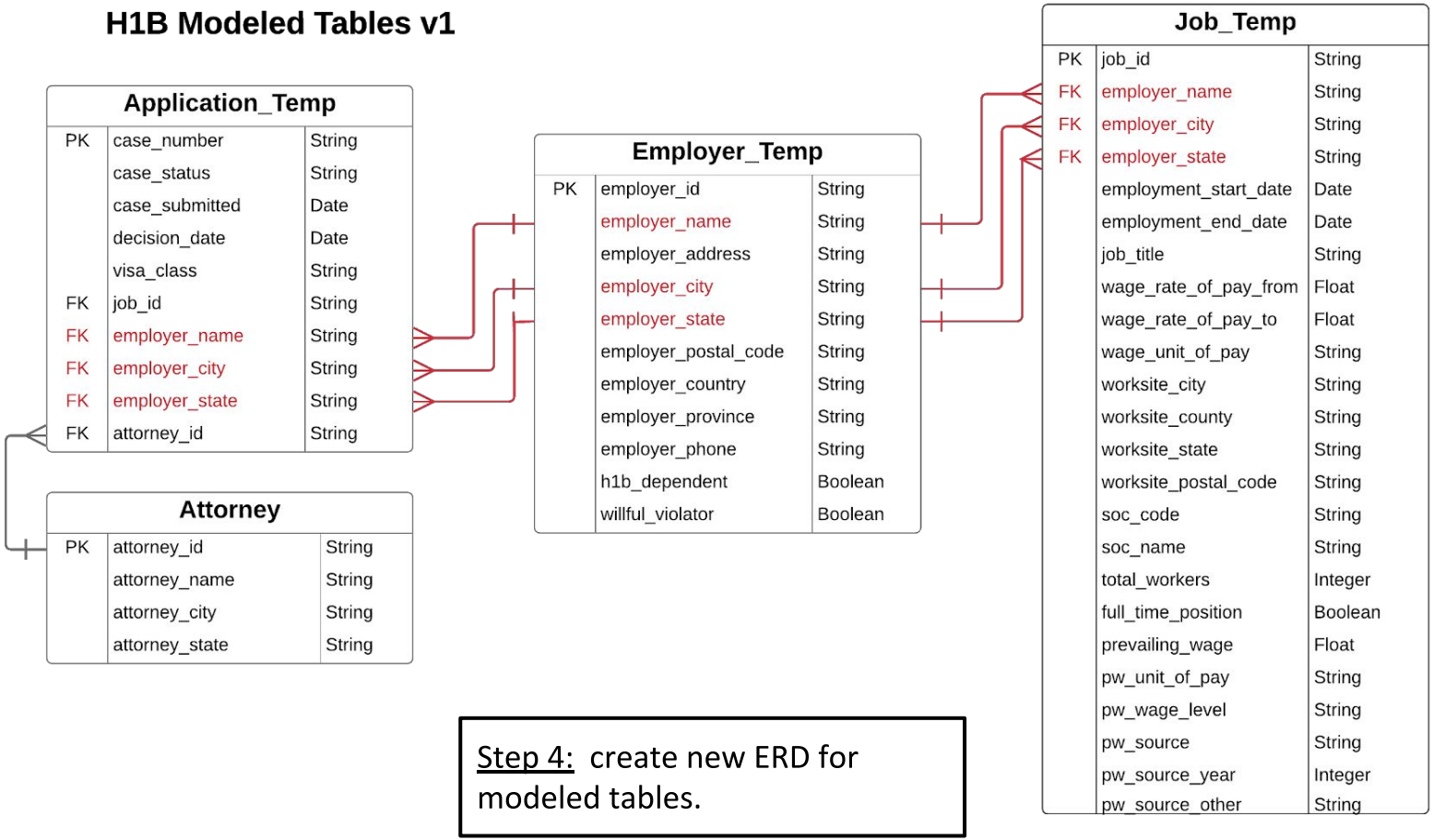
- identify the various Entity Types within and across your staging tables.

```sql
-- Create Employer_Temp tables and assign each record a unique employer_id
-- Table contains duplicate employer records
-- TO DO: remove duplicates records through Beam
CREATE TABLE h1b_modeled.Employer_Temp AS
SELECT generate_uuid() as employer_id, *
FROM
(SELECT DISTINCT employer_name, employer_address, employer_city, employer_state,
 employer_postal_code, employer_country, employer_province, CAST(employer_phone AS STRING) as employer_phone,
 CAST(CASE WHEN h1b_dependent = 'N' THEN 'False'
 WHEN h1b_dependent = 'Y' THEN 'True'
 ELSE NULL END as BOOL) AS h1b_dependent,
 willful_violator
 FROM h1b_staging.H1B_Applications_2018
 WHERE employer_name IS NOT NULL AND employer_name != '1' AND employer_city IS NOT NULL
 UNION DISTINCT
 SELECT DISTINCT employer_name, employer_address, employer_city, employer_state,
 employer_postal_code, employer_country, employer_province, employer_phone, h1b_dependent, willful_violator
 FROM h1b_staging.H1B_Applications_2017
 WHERE employer_name IS NOT NULL AND employer_name != '1' AND employer_city IS NOT NULL
 UNION DISTINCT
 SELECT DISTINCT employer_name, employer_address, employer_city, employer_state,
 employer_postal_code, employer_country, employer_province, employer_phone, h1b_dependent, willful_violator
 FROM h1b_staging.H1B_Applications_2016
 WHERE employer_name IS NOT NULL AND employer_name != '1' AND employer_city IS NOT NULL
 UNION DISTINCT
 SELECT DISTINCT employer_name, CONCAT(employer_address1, ' ', employer_address2) as employer_address,
 employer_city, employer_state, employer_postal_code, employer_country, employer_province, employer_phone,
 h1b_dependent, willful_violator
 FROM h1b_staging.H1B_Applications_2015
 WHERE employer_name IS NOT NULL AND employer_name != '1' AND employer_city IS NOT NULL
)
ORDER BY employer_name, employer_city;
```

Step 3: create new modeled tables using CTAS statements.

Referenced Code: https://github.com/shirleycohen/h1b_analytics/blob/master/h1b_ctas.sql

H1B Modeled Tables v1



Step 4: create new ERD for modeled tables.