# CS 327E Class 4

September 30, 2019

1) What type of relationship do we have between the *Actor* and *Movie* entity types as shown?

A. *1:1*
B. *1:m*
C. *m:n*

Actor

| id | name | age |
|---|---|---|
| 1 | Robert Downey Jr. | 54 |
| 2 | Lady Gaga | 33 |
| 3 | Gwyneth Paltrow | 47 |
| 4 | Bradley Cooper | 44 |

Movie

| id | title | year |
|---|---|---|
| im | Iron Man | 2007 |
| sb | A Star is Born | 2018 |
| ae | Avengers: Endgame | 2019 |
| tw | The Wife | 2017 |

Cast

| actor | movie |
|---|---|
| 1 | ae |
| 1 | im |
| 3 | ae |
| 2 | sb |

2) How many joins would we need to find the cast members who acted in 'Avengers: Endgame' and return their name and age?

Actor

| id | name | age |
|---|---|---|
| 1 | Robert Downey Jr. | 54 |
| 2 | Lady Gaga | 33 |
| 3 | Gwyneth Paltrow | 47 |
| 4 | Bradley Cooper | 44 |

Movie

| id | title | year |
|---|---|---|
| im | Iron Man | 2007 |
| sb | A Star is Born | 2018 |
| ae | Avengers: Endgame | 2019 |
| tw | The Wife | 2017 |

A. 1
B. 2
C. 3

Cast

| actor | movie |
|---|---|
| 1 | ae |
| 1 | im |
| 3 | ae |
| 2 | sb |

3) Which of the following concepts is not specified by the ER model / ERD?

A. Attribute Types
B. Key Attribute Types
C. Attribute Type Domains
D. None of the above

4) Which of the following is an example of a **generalized** entity type?

A. Customer is a generalization of Person
B. Artist is a generalization of Painter
C. Concert is a generalization of Music Event
D. None of the above

5) Which of the following is an example of a **specialized** entity type?

A. Midterm is a specialization of Exam
B. Student is a specialization of Teacher Assistant
C. Article is a specialization of Book
D. None of the above

# Review Terminology

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

Analogies with OOP:

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

Questions:

- What are the boundaries?
- How to handle hierarchies?

# Design Principles

- A table models one Entity Type and an Entity Type is modeled by one table
- Each field in a table represents an attribute of an entity
- Each field in a table is assigned a strict primitive data type
- Each table has a Primary Key (PK) which is made up of one or more fields
- Each child table has a Foreign Key (FK) that points to its parent(s)
- Each *m:n* relationship is modeled with a junction table

# Design Principles and 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 and College Dataset:

## What can go wrong: data anomalies

**College Staging ERD**

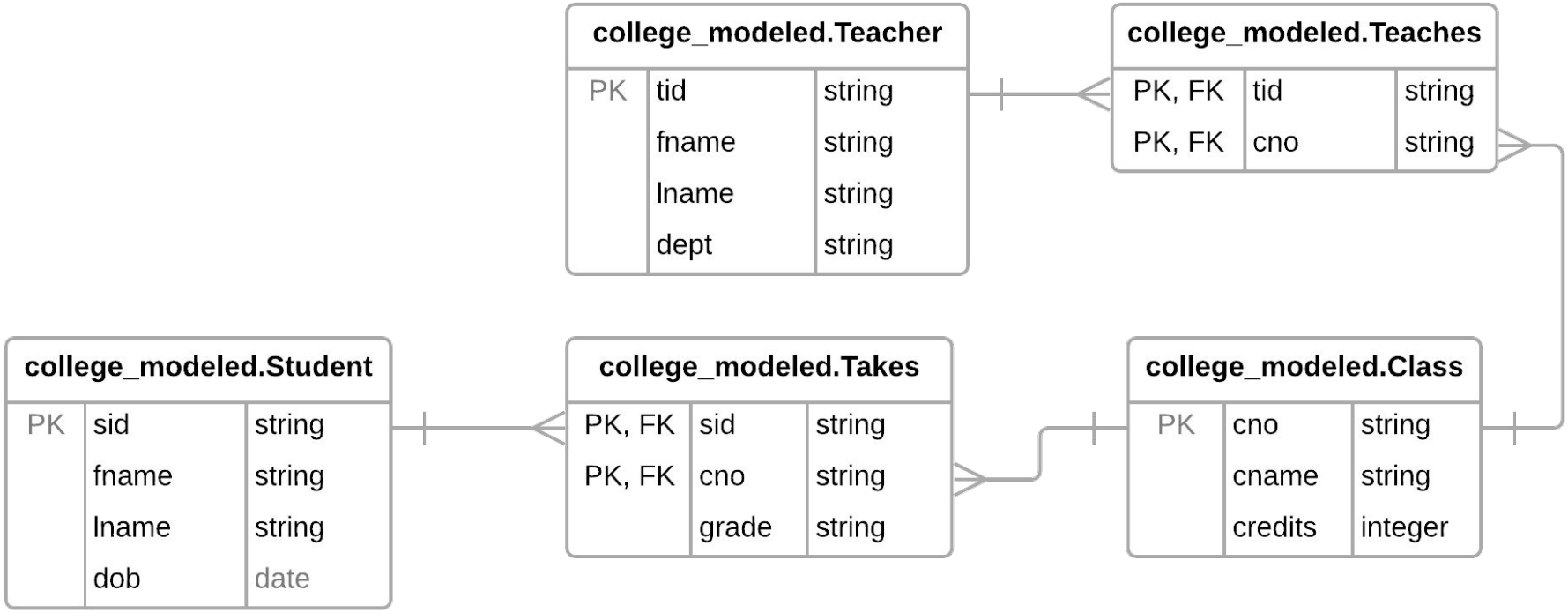
| college_staging.Classes | | |
|---|---|---|
| | tid | String |
| | instructor | String |
| | dept | String |
| | cno | String |
| | cname | String |
| | credits | Integer |

| 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 |

- Insert Anomaly
- Update Anomaly
- Delete Anomaly

# College Modeled ERD



# Common SQL Transforms

- `CREATE TABLE T2 AS SELECT …`

- ```
  SELECT a, b, c FROM T1
  UNION ALL
  SELECT d, e, f FROM T2
  ```

- ```
  SELECT a, b, c FROM T1
  UNION DISTINCT
  SELECT d, e, f FROM T2
  ```

- `SELECT CAST(xyz AS DATE) …`
- `SELECT SAFE_CAST(xyz AS DATE) …`

# Data Modeling Demo

# 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)

Class(cno, cname, credits)

Teacher(tid, instructor, dept)

Takes(sid, cno, grade)

Teaches(tid, cno)

# iClicker Question

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

Student(sid, fname, lname, dob)

Class(cno, cname, credits)

Teacher(tid, instructor, dept)

Takes(sid, cno, grade)

Teaches(tid, cno)

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.

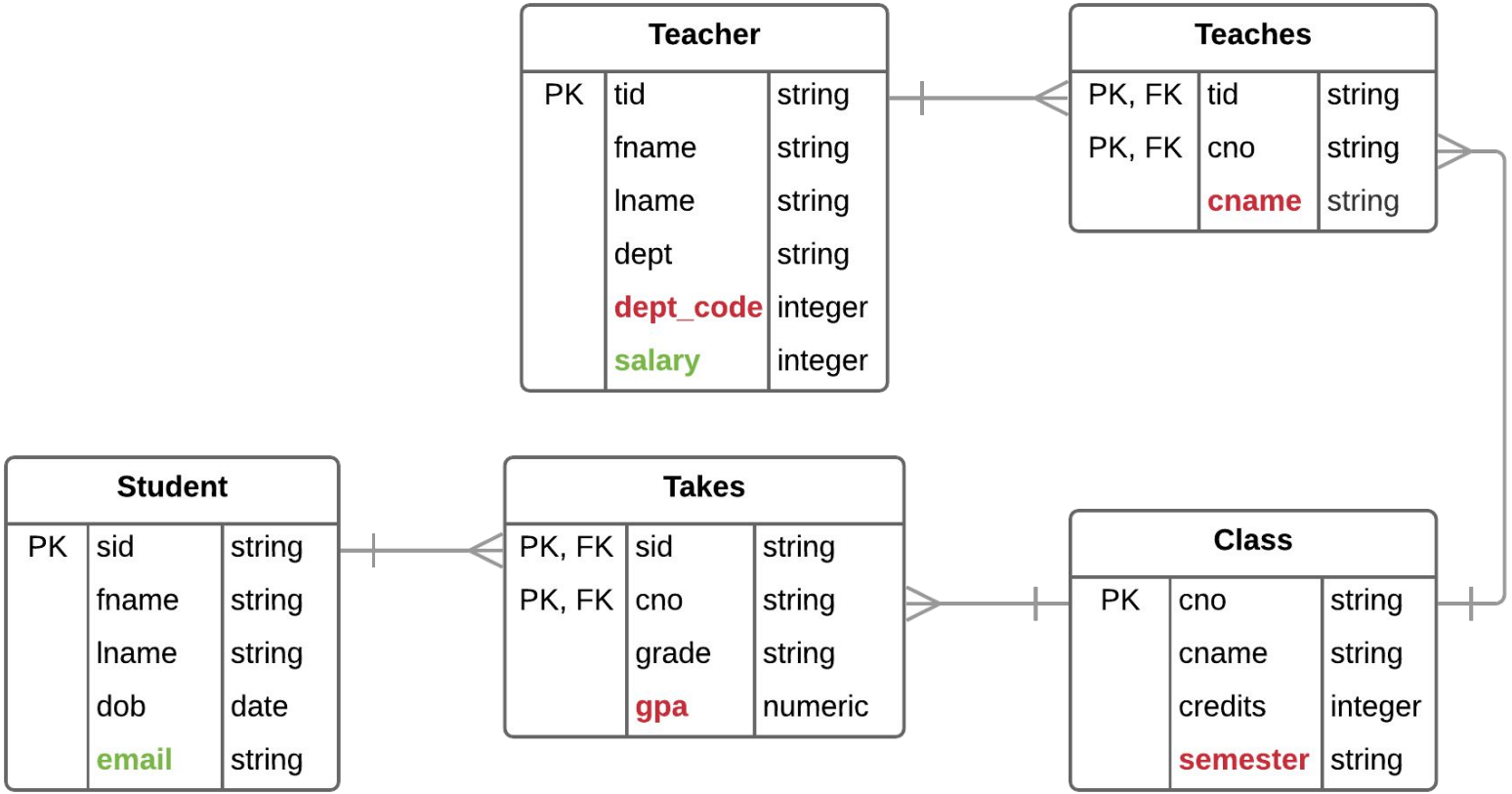
**2NF:** 1NF + all non-key attributes must be *functionally determined* by the *entire* primary key.

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

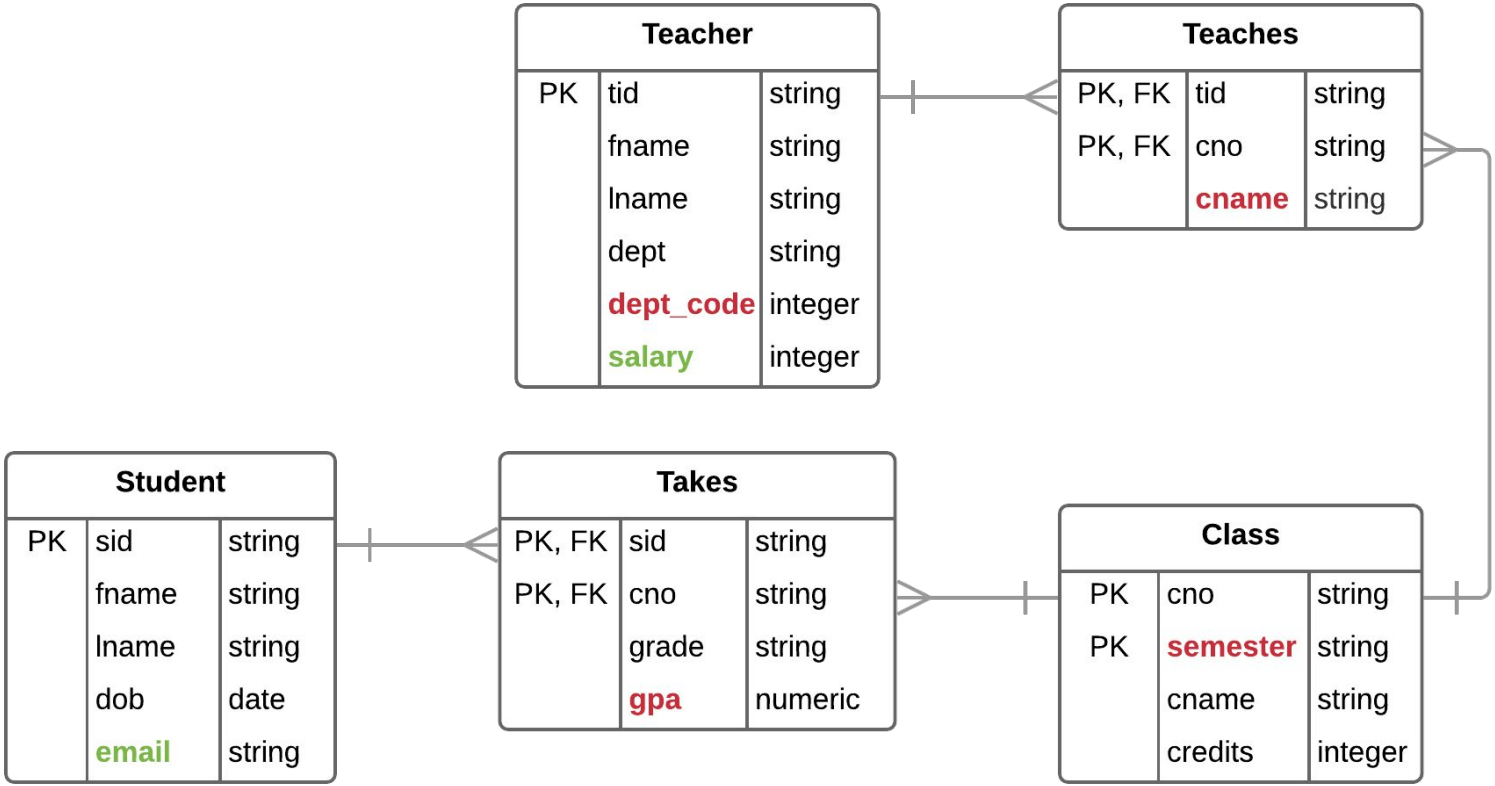
**Functional Dependencies:**
If two records agree on the attributes $A_1, A_2, \ldots, A_n$ then they must also agree on the attributes $B_1, B_2, \ldots, B_n$

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

# Normal Form Violations

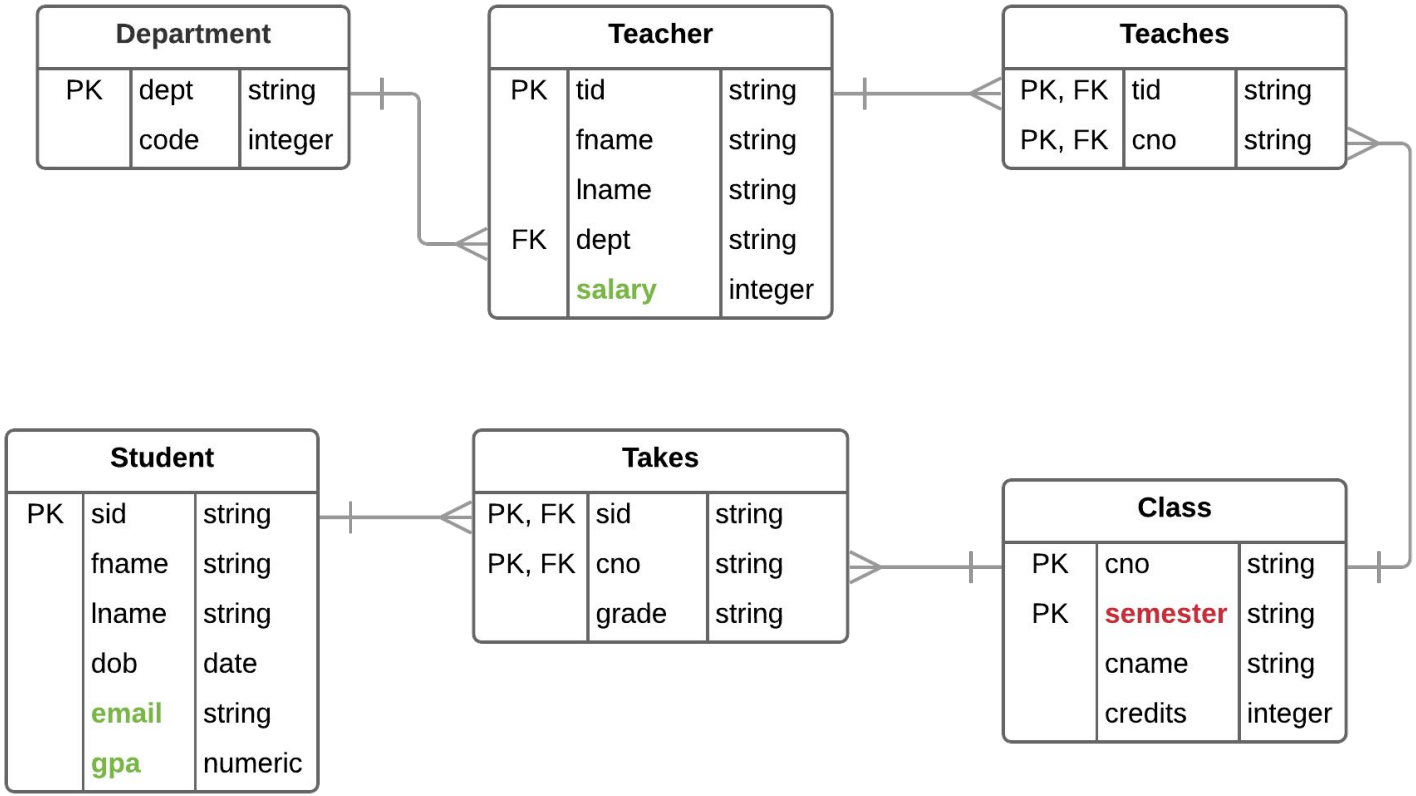


# Normal Form Violations



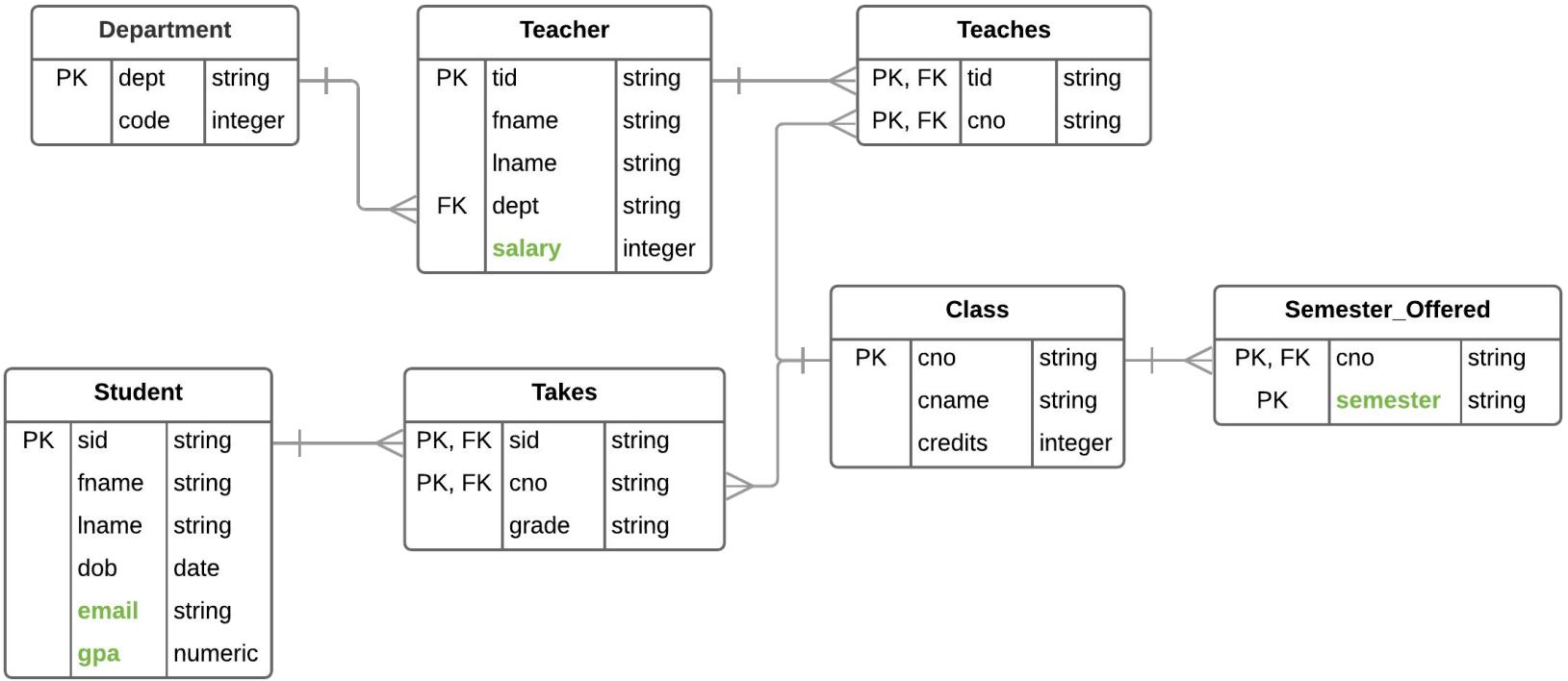
# Practice Problem

Model the semester of a Class without violating normal form



# iClicker Question

Is this a correct representation?
A. Yes B. No



# Milestone 4

1) Requirements: assignment sheet

2) Data modeling questions: sign-up sheet

## 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).

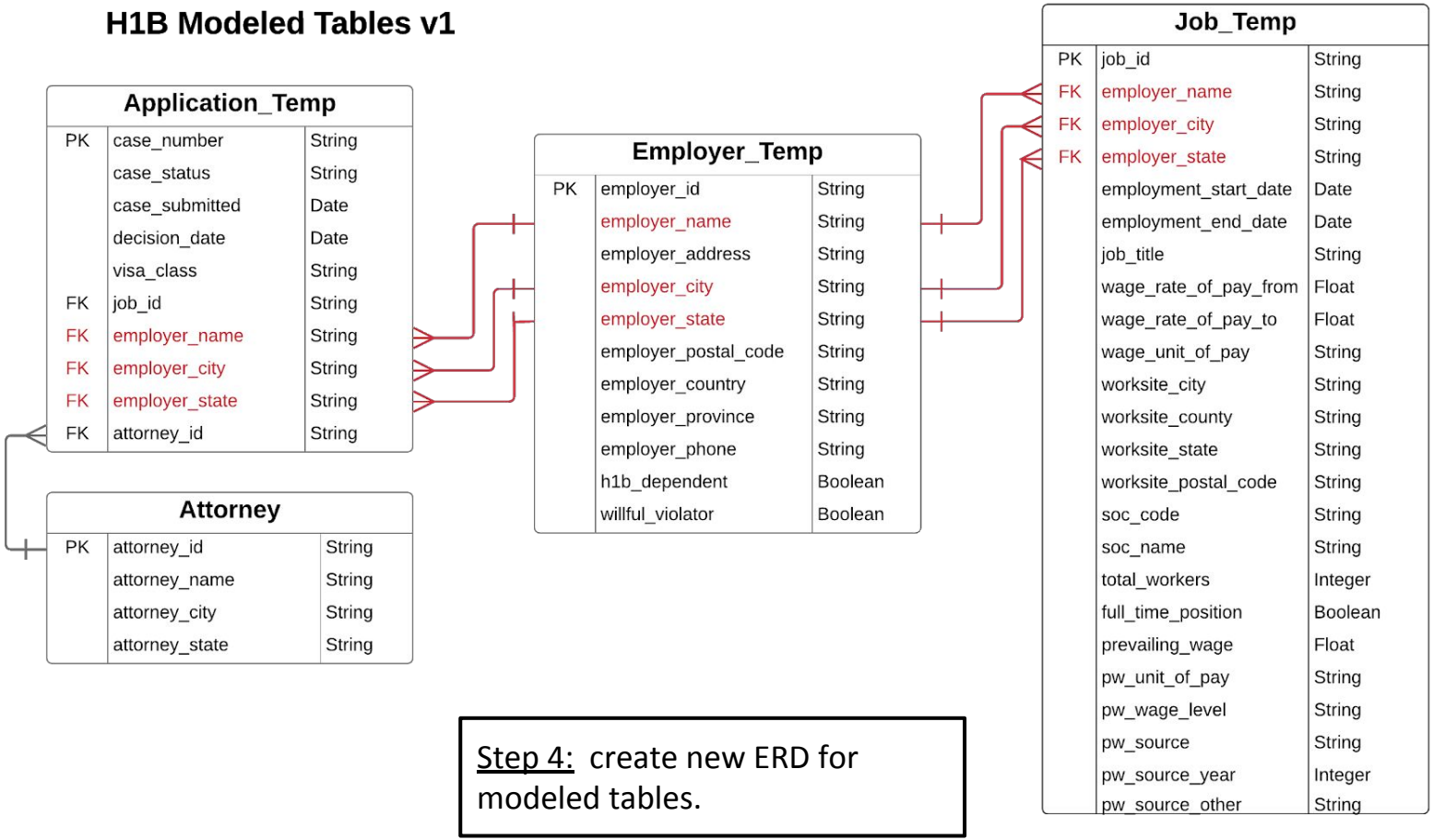
- identify the various Entity Types within and across your staging tables.

```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.