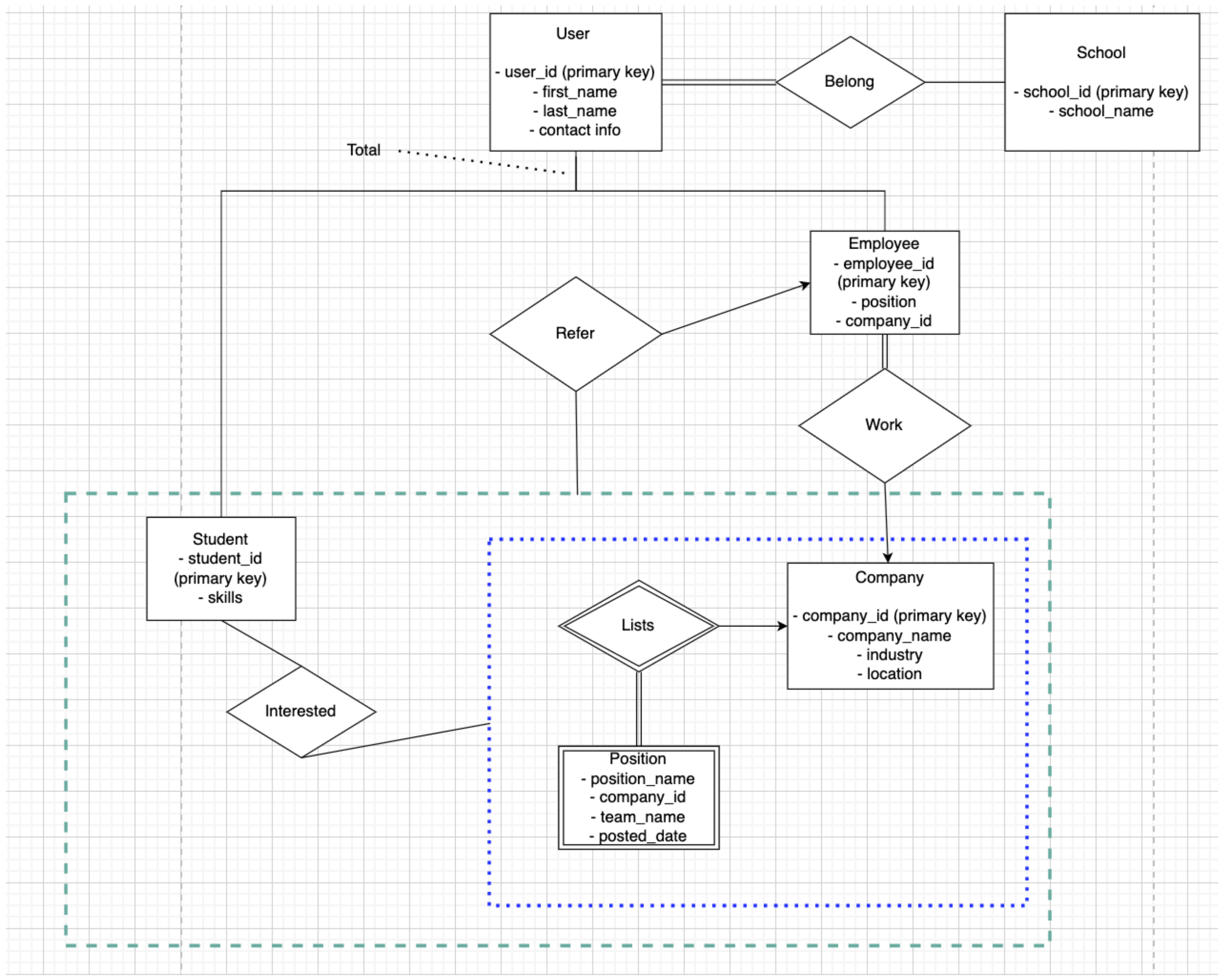
Team: Gyung Hyun Je (gj2353), Shreyans Kothari (sk4819)
Topic: Job referral web application
Date: Sep 26, 2022

Our project aims to develop a web application that connects/matches students with current employees at companies students are interested in, for job referrals. The app will have **two user bases** – current Columbia students and current employees who are Columbia alums. The student will enter their information (school at Columbia, skills, positions at companies they are interested in, etc.) on the application. They can then see a list of people working at specific companies who they could connect with/reach out to. Alternatively, current employees would fill out their information on the application (their school at Columbia, company, position, etc.) on the basis of which they will see a list of current students looking for referrals that they could connect with. Students can get referred for a specific position at a company at most once by an employee, and each employee can refer students for various positions at their company (but only once for each position in a given cycle). Current students would want to use this application because getting a referral at a company would increase their chances of getting an interview/offer for internships/full-time jobs. Employees would want to use this application because at most big companies employees get a bonus if their referral gets hired. Since each employee is only able to refer one candidate for a position at their company, the employee would try to maximize their chances of receiving the bonus by submitting a referral for the “right” candidate. Anticipated challenges of implementing the application include 1. protecting the user information and 2. data quality. Users may not want to provide their contact information visible to everyone, so there should be a measure to encrypt their information. In addition, since the application uses user submitted data/self-report, quality check is necessary. For example, the company and position names should be standardized for easier communication among users.

Relationship sets we have include:

- Relationship set between User and School (*belong* being the relationship)
- Relationship set between Employees and Company (*work* being the relationship)
- Relationship set between Company and Position (*list* being the relationship)
- Relationship set between Student and aggregated set of Company and Position (*interested* being the relationship)
- Relationship set between aggregated set of <Students interested in Position at Company> and Employee (*refer* being the relationship)

ER Diagram



Entity to Schema Translation

User <
user_id varchar PRIMARY KEY
, first_name varchar
, last_name varchar
, school_id int NOT NULL REFERENCES School
, company_id int REFERENCES Company
, student_id int REFERENCES Student
, employee_id int REFERENCES Employee
, contact_info varchar
, CHECK (not(student_id is null AND employee_id is null)) and
(not(student_id is not null AND employee_id is not null))
>

```
Student <
PRIMARY KEY student_id int
, skills varchar
, user_id varchar UNIQUE NOT NULL REFERENCES User
>

Employee <
PRIMARY KEY employee_id int
, company_id int REFERENCES Company
, position varchar
, user_id varchar UNIQUE NOT NULL REFERENCES User
>

School<
PRIMARY KEY school_id int
, school_name varchar
>

Company <
PRIMARY KEY company_id int
, company_name varchar
, industry varchar
, location varchar
>

Position <
PRIMARY KEY (pid)
, pid int serial # auto incrementing id, equivalent to PRIMARY KEY (company_id, position_title)
, position_title varchar
, team_name varchar
, posted_date date
, company_id int NOT NULL REFERENCES Company
        ON DELETE CASCADE
>

User_belongs_School < # cannot force "user has to belong to a school" (will be enforced by the
application itself as a mandatory field), but if there is a relationship then user_id and school_id is
not null
PRIMARY KEY (user_id, school_id)
, user_id varcharg NOT NULL REFERENCES User
, school_id int NOT NULL REFERENCES School
>
```

```
Work_at <
PRIMARY KEY (employee_id, company_id)
, employee_id int UNIQUE NOT NULL REFERENCES Employee
, company_id int NOT NULL REFERENCES Company
        ON DELETE NO ACTION
>

Student_interest <
PRIMARY KEY (student_id, position_at_company) as “student_position_id” # new name to reference the primary key
, student_id int NOT NULL REFERENCES Student
, position_at_company int REFERENCES Position(pid)
, company_id int NOT NULL REFERENCES Company
>

Refer <
PRIMARY KEY (student_position_id, employee_id)
, student_position_id int UNIQUE NOT NULL REFERENCES Student_interest
, employee_id int NOT NULL REFERENCES Employee
, student_id int NOT NULL REFERENCES Student
>
```