Indian Institute of Technology Delhi

COL362 Project Report

Aniket Gupta: 2019CS10327

Mohammed Jawahar Nausheen: 2019CS10371

Hardeep Kaur: 2019CS10354

Contents

Section 1

1.1 Motivation

We have designed an application called **'GrabAJob'** that works as a job portal. Job seekers can easily contact an extensive collection of job vacancies according to their profile. The employers can use it to pace up their hiring process. It would be easy for recruiters to verify job applications online than physically scan printed resumes. Hence, it helps save time and endeavour of both applicants and recruiters, which forms the motivation of this project.

1.2 ER diagram

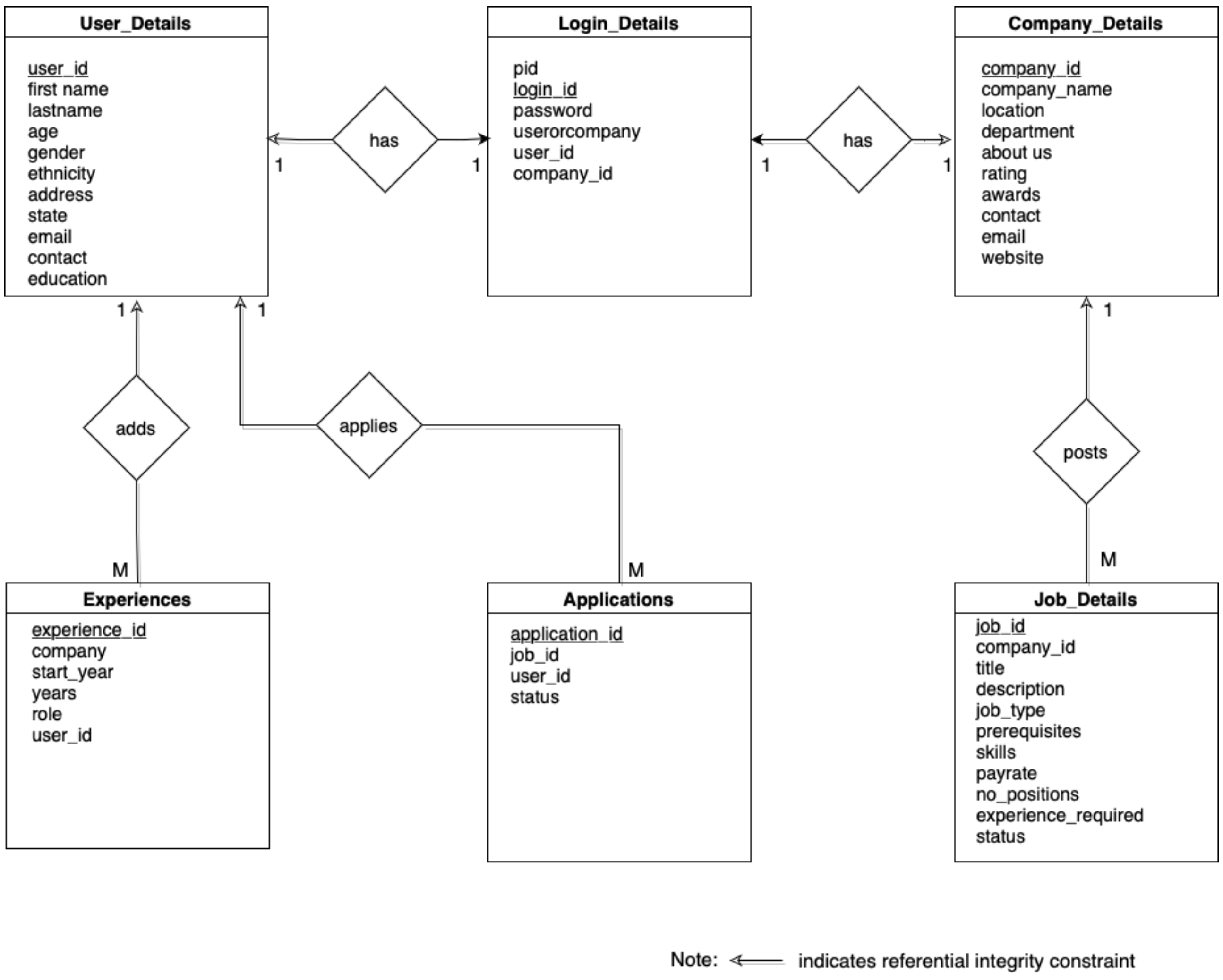


Figure 1: *ER Diagram*

1.3 Description

The job seeker, i.e., the applicant and the recruiter, i.e., the company form the main interacting agents in the portal. The portal first needs a user to sign up in either of the above roles. Then the user would be logged in to the respective version of portal. Following are the actions available for the applicant and company.

- **Applicant**
  - Add experience: An applicant can add multiple experiences which would be visible to the recruiters.
  - Search a job: An applicant can search for jobs of his interest by applying filters on location, company, job domain and type.
  - Apply for a job: An applicant can apply for multiple jobs.
- **Company**
  - Post a job: A company can post multiple jobs.
  - Delete a job: A company can delete a job after posting it. All the corresponding applications would also be deleted.
  - Accept/ Reject an application: A company can accept or reject a job after seeing the details of applicant. The same would be updated on applicant side.

Section 2

2.1 Data sources

We used the following Kaggle datasets

- Naukri job listings https://www.kaggle.com/PromptCloudHQ/jobs-on-naukricom
- Indian names dataset https://www.kaggle.com/ananysharma/indian-names-dataset
- Education dataset https://www.kaggle.com/arashnic/hr-analytics-job-change-of-data-scientists
- Indian cities dataset https://www.kaggle.com/shubhendra7/indian-cities-dataset

2.2 Download and Clean up

- The Naukri job listings dataset was the primary source of our data. It contained job listings with title, description, payrate and other details along with the details of associated company. The job details table and company details table were created using this.
- However user details were not readily available hence we had to make use of multiple datasets, namely the Indian names dataset, education dataset, Indian cities dataset, along with the experiences dataset for experiences table.
- The applications table was created by making suitable matchings between job details and experiences
- The login details table was created using the emailid of user or company along with a generated password purely for operational purposes.

2.3 Statistics

Table name	Time to load	Number of tuples	Size after cleanup
company_details	74.139 ms	6768	1.7 MB
job_details	1274.729 ms	19394	43.3 MB
user_details	16.873 ms	2000	195.2 kB
experiences	45.636 ms	400	246.4 kB
login_details	121.744 ms	8768	507.1 kB
applications	264.811 ms	13641	245.5 kB

Size of raw datasets before clean up

- Naukri job listings - 52.26 MB
- Indian names dataset - 670 kB
- Education dataset- 2.19 MB
- Indian cities dataset- 75 kB

Section 3

3.1

3.1.1 What a user sees

1. Login page: User enters login id and password, redirect to home page
2. Register Page: User enters login id and password, chooses user under ‘Register As’
3. User home page:
   (a) Jobs whose application window is open with option to apply
   (b) Filter to search jobs depending on job type, job category, company name, job location
   (c) Link to jobs applied by user
   (d) Link to User Profile
   (e) Link to Log out
4. Jobs Applied:
   (a) Job listings user has applied for with application status
5. User profile page
   (a) Edit user details (email id cannot be changed)
   (b) Add new experiences

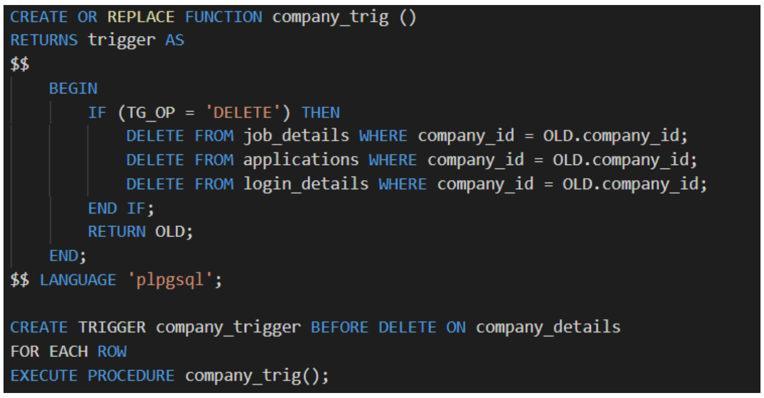
3.1.2 What a company sees

1. Login page: Company enters login id and password, redirect to home page
2. Register Page: Company enters login id and password, chooses company under ‘Register As’
3. Company home page:
   (a) Company details section on left
   (b) All jobs posted by this company. For each job
      i. Job details
      ii. Option to close or open application window
      iii. See users who have applied for this job, links to application page
      iv. Delete job listing
   (c) Applications page: All users who have applied for this job. For each user
      i. User details
      ii. Option to accept or reject
   (d) Link to company profile
   (e) Link to log out
4. Post Job: Create a new job listing by entering job details
5. Company profile page: Edit company details (email id cannot be changed)

3.2 System View

3.2.1 Special features

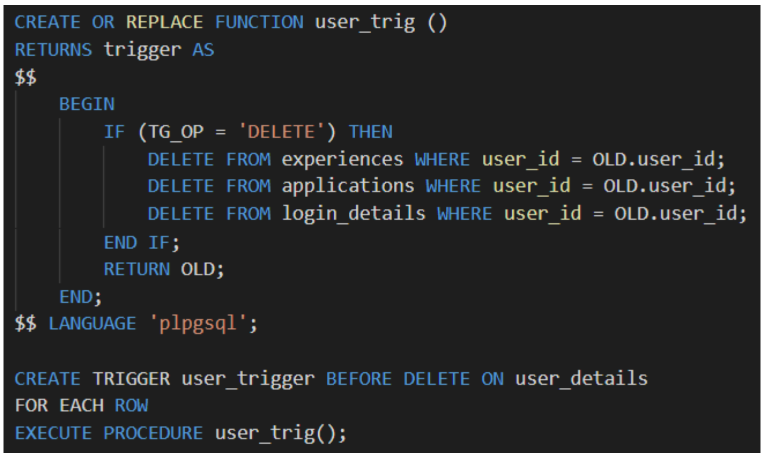
- We have created indexes on different columns of different tables to optimise the query processing. The columns/tables on which we created indexes are:
  - Column company_name of table company_details.
  - Column location of table company_details.
  - Column skill of table job_details.
  - Column title of table job_details.
  - Column user_id of table experiences.
  - Column job_id of table applications.
- We have created two triggers to manage change in database from the admin side. If the admin tries to delete tuples in user_details, then the tuples in experiences, applications and login_details tables corresponding to this user are also deleted. Similarly, when tuples in company_details are deleted, then job_details, applications and login_details corresponding to this company are also deleted.



```
CREATE OR REPLACE FUNCTION company_trig ()
RETURNS trigger AS
$$
    BEGIN
        IF (TG_OP = 'DELETE') THEN
            DELETE FROM job_details WHERE company_id = OLD.company_id;
            DELETE FROM applications WHERE company_id = OLD.company_id;
            DELETE FROM login_details WHERE company_id = OLD.company_id;
        END IF;
        RETURN OLD;
    END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER company_trigger BEFORE DELETE ON company_details
FOR EACH ROW
EXECUTE PROCEDURE company_trig();
```

Figure 2: *Trigger on company_details table*



```
CREATE OR REPLACE FUNCTION user_trig ()
RETURNS trigger AS
$$
    BEGIN
        IF (TG_OP = 'DELETE') THEN
            DELETE FROM experiences WHERE user_id = OLD.user_id;
            DELETE FROM applications WHERE user_id = OLD.user_id;
            DELETE FROM login_details WHERE user_id = OLD.user_id;
        END IF;
        RETURN OLD;
    END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER user_trigger BEFORE DELETE ON user_details
FOR EACH ROW
EXECUTE PROCEDURE user_trig();
```

Figure 3: *Trigger on user_details table*

3.2.2 Queries for user

1. Login page: Match credentials
   "SELECT * FROM login_details WHERE login_id = email AND password = password"

2. Register page: Check for unique constraint, create new user and login credentials
   “SELECT * FROM login_details WHERE login_id = email;
   “INSERT INTO user_details VALUES (userid, NULL, NULL, NULL, NULL, NULL, NULL, NULL, email, NULL, NULL)”;
   “INSERT INTO login_details VALUES (pid, email, password, user, userid, NULL)”

3. User home page:

   (a) Show jobs that are open to apply in a page wise manner
   “SELECT job_id, company_details.company_name, location, title, description, job_type, prerequisites, skills, pay_rate, no_positions, experience_required FROM (select * from job_details where status = ’1’ limit 10 offset pagenum) as jobs join company_details on company_details.company_id = jobs.company_id and job_id not in (select job_id from applications where user_id= userid)”
   Add application on clicking apply
   “insert into applications values(appid, jobid, userid, 0)”

   (b) User sees list of job types, categories, companies and locations
   “SELECT distinct(location) FROM company_details order by location”
   “SELECT distinct(company_name) FROM company_details order by company_name”
   “SELECT distinct(skills) FROM job_details”
   Filter to search jobs depending on job type, job category, company name, job location by choosing from the list
   “SELECT job_id, company_details.company_name, location, title, description, job_type, prerequisites, skills, pay_rate, no_positions, experience_required FROM (select * from job_details where status = ’1’) as jobs join company_details on company_details.company_id = jobs.company_id and job_type = jobtype and skills = jobcategory and location = loc and company_name = name”

   (c) Jobs applied

   (d) User profile

   (e) Link to Log out

4. Jobs Applied:

   (a) Job listings user has applied for with application status
   “SELECT jobs.job_id, company_details.company_name, title, description, job_type, prerequisites, skills, pay_rate, no_positions, experience_required , location, contact, email, applications.status FROM job_details as jobs join company_details on company_details.company_id = jobs.company_id join applications on applications.job_id = jobs.job_id and applications.user_id =userid”

5. User profile page

   (a) Edit user details (email id cannot be changed)
   “UPDATE user_details SET firstname =fname, lastname =lname, age =a, gender =g, ethnicity =eth, address =addr, state =stat, contact =num, Education =edu WHERE user_id =userid”

   (b) Add new experiences
   “INSERT INTO experiences VALUES(exp_id,company,start,end,role,userid)”

3.2.3 Queries for company

1. Login page: Same as for user
2. Register Page: Check for unique constraint, create new user and login credentials
   "SELECT * FROM login_details WHERE login_id = email;
   "INSERT INTO company_details VALUES (compid, NULL, NULL,NULL, NULL, NULL , NULL, NULL, email, NULL)"
   "INSERT INTO login_details VALUES (pid, email, password, 'company', NULL, compid)"
3. Company home page:
   (a) Company details section on left
   "SELECT * FROM company_details WHERE company_id = compid"
   (b) All jobs posted by this company. For each job
      i. Job details
      "SELECT * FROM job_details WHERE company_id = compid"
      ii. Option to close or open application window
      "UPDATE job_details SET status = newstatus WHERE job_id = jobid"
      iii. Link to applications page
      iv. Delete job listing (cascades to applications)
      "DELETE FROM job_details WHERE job_id = jobid"
   (c) Applications page: All users who have applied for this job. For each user
      i. Applicant details
      "SELECT application_id, firstname, lastname, age, gender, education, email, contact, status, user_id FROM (user_details NATURAL JOIN (SELECT * FROM applications WHERE job_id = j_id)s) b) a ORDER BY application_id"
      ii. Option to accept or reject
      "UPDATE applications SET status = 1 WHERE application_id = newstatus"
   (d) Link to log out
4. Post Job: Create a new job listing by entering job details
   "INSERT INTO job_details (job_id, company_id, title, description, job_type, prerequisites, skills, pay_rate, no_positions, experience_required, status) VALUES (job_id, company_id, title, description, job_type, prerequisites, skills, pay_rate, no_positions, experience_required, '1')"
5. Company profile page: Edit company details (email id cannot be changed)
   "UPDATE company_details SET company_name = cname, about_us = abt, department = dep, location = loc, awards = awd, contact = cont, website = webs WHERE company_id = compid"

3.3 Sample run times

We have tested our database on a lot of queries but the following are some representative examples of of the different queries that we ran on our dataset. The index in the below examples are same as those in section 3.2.2 and 3.2.3.
For queries in section 3.2.2

- 'Select * from login_details where login_id='aarti1@gmail.com' and password='670656"
  Time taken: 0.678ms

- ‘insert into user_details values (2001, ’Aniket’, ’Gupta’, 21, ’Male’, ’Indian’, ’UP’, ’UP’, ’guptaaniket261@gmail.com’, ’1234567890’, ’CBSE’)’
  Time taken: 3.410 ms

- ‘SELECT job_id, company_details.company_name, location, title, description, job_type, prerequisites, skills, pay_rate, no_positions, experience_required FROM (select * from job_details where status = ’1’ limit 10 offset 0) as jobs join company_details on company_details.company_id = jobs.company_id and job_id not in (select job_id from applications where user_id=2001)’
  Time taken: 2.359 ms

- ‘SELECT * FROM applications WHERE user_id = 2001’
  Time taken: 0.753 ms

- ‘SELECT application_id, firstname, lastname, age, gender, education, email, contact, status, user_id FROM (user_details NATURAL JOIN (SELECT * FROM applications WHERE job_id = 1) b) a ORDER BY application_id’
  Time taken: 1.243 ms

- ‘DELETE FROM job_details WHERE job_id = 100;’
  Time taken: 0.469 ms

- ‘SELECT jobs.job_id, company_details.company_name, title, description, job_type, prerequisites, skills, pay_rate, no_positions, experience_required , location, contact, email, applications.status FROM job_details as jobs join company_details on company_details.company_id = jobs.company_id join applications on applications.job_id = jobs.job_id and applications.user_id = 2001’
  Time taken: 2.431 ms

- ‘UPDATE company_details SET company_name = ’Airtel’, about_us = ’-’, department = ’Telecom’, location = ’Delhi’, awards = ’none’, contact = ’1234567890’, website = ’airtel.com’ WHERE company_id = 1000’
  Time taken: 1.021 ms