SKILL CONNECT
(Team ID – T306)

Group Members:

Parth Parekh	**-**	**202201200**
Meet Sarvan	**-**	**202201218**
Nishank Kansara	**-**	**202201227**
Harmit Khimani	**-**	**202201231**
Parth Prajapati	**-**	**202201250**

Prof. P. M. Jat
TA. Prachi Singh

Summary:

Skill Connect is an online platform designed to seamlessly bridge the gap between job seekers and companies. Our objective is to provide a centralized hub where users can showcase their skill sets and experience while enabling companies to post job vacancies and review applicant profiles efficiently. Moreover, our innovative platform allows users to directly contact companies with job vacancies, facilitating direct communication and enhancing the chances of meaningful connections. Additionally, companies can effortlessly select users from the platform based on their skills, ensuring a comprehensive and efficient recruitment process. This comprehensive approach makes it easier for both companies and job seekers to find the best match while encouraging direct engagement in the hiring process.

Experience:

During the creation of our project, Skill Connect, we encountered several challenges along the way. One significant hurdle was when we were developing the Entity-Relationship (ER) diagram. We struggled to determine the key attributes of each entity and establish their relationships with others.

Even after finalizing the ER diagram, we faced difficulties in creating queries and implementing them in SQL.

Additionally, when we learned about normalization, we realized the necessity of updating our relational schema, ER diagram, and Data Definition Language (DDL) scripts accordingly.

Even though we faced some tough challenges, working through them made us work better together and understand how databases work even more.

From this project, we learned that it's important to keep trying even when things get tough. Working together helped us solve problems better, and we realized the need to keep learning about new concepts like normalization. We also got better at managing complex databases, solving problems and working as a team. This experience taught us a lot that we can use in the future.

Top 3 Queries:

1.Find users who have both job and internship experiences.

```
SELECT DISTINCT u.user_id, u.user_name FROM user_t u
WHERE u.user_id IN (SELECT e.user_id FROM experience e
) AND u.user_id IN (SELECT ie.eid FROM
internship_experience ie);
```

2.Provide me with the application status for the company to which I have Applied.

```
SELECT u.user_name, c.company_name, a.status FROM user_t
u JOIN apply a ON u.user_id = a.user_id JOIN offers o ON
a.offer_id = o.offer_id JOIN company c ON o.cid = c.cid WHERE
a.user_id = 8 AND c.company_name = 'Wipro Limited';
```

3.Please provide a list of companies in my state with contracts shorter than x months and requiring a skill that I possess.

```
SELECT DISTINCT c.company_name, con.contract_end_date,
con.contract_start_date FROM company c JOIN offers o ON
c.cid = o.cid JOIN contract con ON o.offer_id = con.offer_id
JOIN offices off ON c.cid = off.cid WHERE off.state =
'Karnataka' AND o.skill_required='Android Development' AND
(con.contract_end_date - con.contract_start_date) < (12*30);
```

DDL SCRIPT:

```
CREATE TABLE user_t(
   user_id INT PRIMARY KEY,
   user_name VARCHAR(30) NOT NULL,
   github_handle VARCHAR(50),
   codeforces_handle VARCHAR(50),
   leetcode_handle VARCHAR(50),
   dob DATE NOT NULL,
   city VARCHAR(30) NOT NULL,
   state VARCHAR(30) NOT NULL
);

CREATE TABLE mobile_number (
   user_id INT,
   mobile_number VARCHAR(15),
   PRIMARY KEY (user_id, mobile_number),
   FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE
);

CREATE TABLE email_id (
   user_id INT,
   email_id VARCHAR(255),
   PRIMARY KEY (user_id, email_id),
   FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE
);

CREATE TABLE user_login (
   user_id INT,
   applicant_username VARCHAR(50) PRIMARY KEY,
   password VARCHAR(100) NOT NULL,
   FOREIGN KEY (user_id) REFERENCES user_t(user_id)
);

CREATE TABLE project (
   pno INT PRIMARY KEY,
   user_id INT,
   pname VARCHAR(50) NOT NULL,
   description TEXT,
   FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE
);

CREATE TABLE project_techstack (
   pno INT,
   tech VARCHAR(50),
   PRIMARY KEY (pno, tech),
   FOREIGN KEY (pno) REFERENCES project(pno) ON DELETE CASCADE
);

CREATE TABLE skills (
   skill_name VARCHAR(100),
   skill_category VARCHAR(100),
   description TEXT,
   PRIMARY KEY (skill_name)
);

CREATE TABLE possess(
   skill_name VARCHAR(100),
   user_id INT,
   PRIMARY KEY (skill_name,user_id),
   FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE,
   FOREIGN KEY (skill_name) REFERENCES skills(skill_name) ON DELETE
CASCADE
);


CREATE TABLE experience (
   eid INT PRIMARY KEY,
   user_id INT,
   company_name VARCHAR(100) NOT NULL,
   address VARCHAR(255) NOT NULL,
   mode VARCHAR(25) NOT NULL,
   duration VARCHAR(30) NOT NULL,
   FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE
```

```
);

CREATE TABLE internship_experience (
    eid INT PRIMARY KEY,
    supervision_level VARCHAR(50) NOT NULL,
    FOREIGN KEY (eid) REFERENCES experience(eid) ON DELETE CASCADE
);

CREATE TABLE job_experience (
    eid INT PRIMARY KEY,
    seniority_level VARCHAR(50) ,
    FOREIGN KEY (eid) REFERENCES experience(eid) ON DELETE CASCADE
);

CREATE TABLE schooling (
    seat_no_10th VARCHAR(20) PRIMARY KEY,
    seat_no_12th VARCHAR(20) NOT NULL,
    user_id INT,
    pass_out_year_12th INT NOT NULL,
    school_name_12th VARCHAR(100) NOT NULL,
    school_name_10th VARCHAR(100) NOT NULL,
    percentage_10th DECIMAL(5, 2) NOT NULL,
    percentage_12th DECIMAL(5, 2) NOT NULL,
    school_address_12th VARCHAR(100) NOT NULL,
    JEE_percentile DECIMAL(5, 2),
    FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE
);

CREATE TABLE expectation (
    exp_id INT ,
    user_id INT ,
    contract_time VARCHAR(50),
    salary DECIMAL(10, 2),
    preference_mode VARCHAR(100) NOT NULL,
    PRIMARY KEY (exp_id, user_id),
    FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE
);
```

```
CREATE TABLE degree (
   specialization VARCHAR(50),
   degree VARCHAR(50),
   PRIMARY KEY (specialization, degree)
);

CREATE TABLE college (
   college_name VARCHAR(50) PRIMARY KEY,
   address VARCHAR(255) NOT NULL
);

CREATE TABLE completed (
   user_id INT,
   specialization VARCHAR(50),
   degree VARCHAR(50),
   college_name VARCHAR(50),
   passout_year INT NOT NULL,
   CGPA DECIMAL(4, 2) NOT NULL,
   PRIMARY KEY (user_id, specialization, degree, college_name),
   FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE,
   FOREIGN KEY (specialization, degree) REFERENCES degree(specialization,
degree) ON DELETE CASCADE,
   FOREIGN KEY (college_name) REFERENCES college(college_name) ON
DELETE CASCADE
);

CREATE TABLE offer (
   college_name VARCHAR(50),
   specialization VARCHAR(50),
   degree VARCHAR(50),
   PRIMARY KEY (college_name, specialization, degree),
  FOREIGN KEY(specialization, degree) REFERENCES degree(specialization,
degree) ON DELETE CASCADE,
FOREIGN KEY (college_name) REFERENCES college(college_name) ON DELETE
CASCADE
);
```

```sql
CREATE TABLE company (
   cid INT PRIMARY KEY,
   company_name VARCHAR(50) NOT NULL,
   website_link VARCHAR(100),
   headquarter_id INT NOT NULL
);

CREATE TABLE company_email (
   cid INT,
   email_id VARCHAR(255),
   PRIMARY KEY (cid, email_id),
   FOREIGN KEY (cid) REFERENCES company(cid) ON DELETE CASCADE
);

CREATE TABLE company_contact_number (
   cid INT,
   contact_number VARCHAR(15),
   PRIMARY KEY (cid, contact_number),
   FOREIGN KEY (cid) REFERENCES company(cid) ON DELETE CASCADE
);

CREATE TABLE company_login (
   company_username VARCHAR(100) PRIMARY KEY,
   password VARCHAR(100) NOT NULL,
   cid INT,
   FOREIGN KEY (cid) REFERENCES company(cid)
);

CREATE TABLE offices (
   office_id INT ,
   cid INT,
   city VARCHAR(20) NOT NULL,
   state VARCHAR(20) NOT NULL,
   PRIMARY KEY (cid, office_id),
   FOREIGN KEY (cid) REFERENCES company(cid)
);
```

```
CREATE TABLE offers (
   offer_id INT PRIMARY KEY,
   cid INT,
   skill_required VARCHAR(30) NOT NULL,
   role VARCHAR(50) NOT NULL,
   number_of_vacancies INT NOT NULL,
   end_date DATE NOT NULL,
   salary DECIMAL(10, 2) NOT NULL,
   FOREIGN KEY (cid) REFERENCES company(cid)
);

CREATE TABLE contract (
   contract_number INT,
   contract_start_date DATE,
   contract_end_date DATE NOT NULL,
   min_duration INT NOT NULL,
   location VARCHAR(255) NOT NULL,
   mode VARCHAR(30) NOT NULL,
   shift VARCHAR(30) NOT NULL,
   offer_id INT,
   PRIMARY KEY (contract_number,offer_id),
   FOREIGN KEY (offer_id) REFERENCES offers(offer_id)
);

CREATE TABLE apply (
   user_id INT,
   offer_id INT,
   apply_date DATE NOT NULL,
   status VARCHAR(30) NOT NULL,
   PRIMARY KEY (user_id, offer_id),
   FOREIGN KEY (user_id) REFERENCES user_t(user_id) ON DELETE CASCADE,
   FOREIGN KEY (offer_id) REFERENCES offers(offer_id) ON DELETE CASCADE
);

CREATE TABLE search (
   cid INT ,
   user_id INT ,
           PRIMARY KEY (cid, user_id),
   FOREIGN KEY (cid) REFERENCES company(cid),
   FOREIGN KEY (user_id) REFERENCES user_t(user_id)
);
```

QUERIES:

1.Provide a list of companies in my state with contracts shorter than x months.

```
SELECT DISTINCT

c.company_name,con.contract_end_date,con.contract_start_date

FROM company c

JOIN offers o ON c.cid = o.cid

JOIN contract con ON o.offer_id = con.offer_id

JOIN offices off ON c.cid = off.cid

WHERE off.state = 'Karnataka'

AND (con.contract_end_date - con.contract_start_date) < (5*30);
```

2.Find users who have both job and internship experiences.

```
 SELECT DISTINCT u.user_id, u.user_name
FROM user_t u
WHERE u.user_id IN (
   SELECT e.user_id FROM experience e
) AND u.user_id IN (
   SELECT ie.eid FROM internship_experience ie
);
```

3.list of skills, arranged in descending order, showcasing the most commonly sought-after skills in the job market

```
SELECT skill_name, COUNT(*) AS frequency

FROM (

         SELECT skill_name

        FROM offers

      JOIN skills ON offers.skill_required = skills.skill_name
) AS subquery

GROUP BY skill_name

ORDER BY frequency DESC;
```

4.Provide me with the application status for the company to which I have Applied.

```
SELECT u.user_name, c.company_name, a.status

FROM user_t u

JOIN apply a ON u.user_id = a.user_id

JOIN offers o ON a.offer_id = o.offer_id

JOIN company c ON o.cid = c.cid

WHERE a.user_id = 8

AND c.company_name = 'Wipro
Limited';
```

5.Identify offers that require skills possessed by users who have applied but haven't been accepted:

```
SELECT DISTINCT o.offer_id, o.role, o.skill_required

FROM offers o

JOIN apply a ON o.offer_id = a.offer_id

JOIN possess p ON a.user_id = p.user_id AND o.skill_required = p.skill_name
WHERE a.status <> 'Accepted';
```

6.Provide a state wise count of all the users.

```
SELECT state, COUNT(user_id) AS user_count

FROM user_t

GROUP BY State;
```

7.Provide a state wise count of all the companies

```
SELECT c.cid, c.company_name, c.website_link, o.city, o.state

FROM company c

NATURAL JOIN offices o;
```

8.Calculate the average salary offered by companies for roles with at least 1 vacancies:

```
SELECT o.role, AVG(o.salary) AS average_salary
FROM offers o
GROUP BY o.role
HAVING COUNT(o.offer_id) >= 1;
```

9.Find offers with the highest number of vacancies and their corresponding company names:

```
SELECT o.offer_id, o.role, c.company_name, o.number_of_vacancies
FROM offers o
JOIN company c ON o.cid = c.cid
ORDER BY o.number_of_vacancies DESC
LIMIT 7;
```

10.List all registered Users with their information.

```
SELECT u.user_id, u.user_name, u.github_handle, u.codeforces_handle,
u.leetcode_handle, u.dob, u.city, u.state

FROM user_t u;
```

11.List of users with a JEE percentile above 90 and CGPA above 8:

```
SELECT u.user_id, u.user_name, s.JEE_percentile, c.CGPA
```

```
FROM user_t u
JOIN schooling s ON u.user_id = s.user_id
JOIN completed c ON u.user_id = c.user_id
WHERE s.JEE_percentile > 90 AND c.CGPA > 8;
```

12.Get the top 5 companies with the highest number of offers that have been applied to:

```
SELECT c.company_name, COUNT(DISTINCT a.offer_id) AS total_applied_offers
FROM company c
JOIN offers o ON c.cid = o.cid
JOIN apply a ON o.offer_id = a.offer_id GROUP
BY c.company_name
ORDER BY total_applied_offers DESC
LIMIT 5;
```

13.List of companies with no offers

```
SELECT c.company_name
FROM company c
LEFT JOIN offers o ON c.cid = o.cid WHERE
o.offer_id IS NULL;
```

14.Retrieve the highest salary offer for each role :

```
SELECT role, MAX(salary) as highest_salary
FROM offers
GROUP BY role;
```

15.Find companies that have offers for roles requiring a skill that no user possesses.

```
SELECT DISTINCT c.company_name, o.skill_required
```

```
FROM company c, offers o
WHERE c.cid = o.cid
AND NOT EXISTS (
    SELECT *
    FROM possess p
    WHERE p.skill_name = o.skill_required
);
```

16.List all registered Companies with their information

```
SELECT c.cid,c.company_name,c.website_link,o.city,o.state
FROM company c
NATURAL JOIN offices o;
```

17.List all the data of the currently available job openings.

```
SELECT *
FROM offers
WHERE end_date > CURRENT_DATE;
```

18.Give me the average salary of offers in companies located in each state about specific skills :

```
SELECT o.state,AVG(off.salary) as avg_salary
FROM offices o
JOIN company c ON o.cid = c.cid
JOIN offers off ON c.cid = off.cid
WHERE off.skill_required = 'Android Development'
GROUP BY o.state;
```

NORMALIZATION PROOFS:

“user_t” Relation:

user_t(user_id, user_name, github_handle, codeforces_handle, leetcode_handle, dob, city, state)

FDs:

{user_id} →user_name
{user_id} →github_handle
{user_id} → codeforces_handle
{user_id} → leetcode_handle
{user_id} → dob
{user_id}→ city
{user_id}→ state

Closure of {user_id}:

$\{user\_id\}^{+}$ ={user_id, user_name, github_handle, codeforces_handle, leetcode_handle, dob, city, state}

Closure of user_id covers all the attributes of the Relation “user_t”.
Thus user_id is a candidate key.

Candidate Key: user_id

Since Candidate key is the subset of Super Key.
The determinant of all the functional dependencies of the relation “user_t” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Mobile_number” Relation:

mobile_number (user_id,mobile_number)

FDs:

{user_id,mobile_number} → {user_id,mobile_number}

Closure of {user_id, mobile_number}:

{user_id,mobile_number}$^+$ ={user_id, mobile_number}

Closure of {user_id,mobile_number}covers all the attributes of the Relation “mobile_number”.
Thus {user_id,mobile_number} is a candidate key.

Candidate Key: {user_id,mobile_number}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “mobile_number” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Email” Relation:

Email(user_id,email_id)

FDs:

{user_id, email_id} → {user_id, email_id}

Closure of {user_id, email_id}:

{user_id, email_id}$^+$ ={user_id, email_id}

Closure of {user_id, email_id} covers all the attributes of the Relation “Email”. Thus {user_id, email_id} is a candidate key.

Candidate Key: {user_id, email_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Email” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“user_login” Relation:

user_login(user_id, applicant_username, password)

FDs:

{applicant_username} → {user_id}
{applicant_username} → {password}

Closure of {applicant_username}:

{applicant_username}$^{+}$ ={applicant_username, user_id, password}

Closure of {applicant_username} covers all the attributes of the Relation “user_login”. Thus {applicant_username} is a candidate key.

Candidate Key: {applicant_username}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “user_login” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Project” Relation:

Project(pno, user_id,pname,description)

FDs:

{pno} → user_id
{pno} → pname
{pno} → description

Closure of {pno}:

$\{pno\}^{+}$ = {pno, user_id, pname, description}

Closure of {pno}covers all the attributes of the Relation “Project”.
Thus {pno} is a candidate key.

Candidate Key: {pno}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Project” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Project_techstack” Relation:

project_techstack(pno,tech)

FDs:

{pno,tech}→{pno, tech}

Closure of {pno,tech}:

$\{pno,tech\}^{+}$ ={pno, tech}

Closure of {pno, tech} covers all the attributes of the Relation “Project_techstack”.
Thus {pno, tech} is a candidate key.

Candidate Key: {pno, tech}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Project_techstack” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Skills” Relation:

skill(skill_name, description, skill_category)

FDs:

{skill_name} → description
{skill_name} → skill_category

Closure of {skill_name}:

$\{skill\_name\}^{+} = \{skill\_name, description\}$

Closure of {skill_name} covers all the attributes of the Relation “Skills”.
Thus {skill_name} is a candidate key.

Candidate Key: {skill_name}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Skills” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Possess” Relation:

possess(skill_name,user_id)

FDs:

{skill_name,user_id}→{skill_name,user_id}

Closure of {skill_name,user_id}:

{skill_name,user_id}$^{+}$ ={skill_name,user_id}

Closure of {skill_name,user_id}covers all the attributes of the Relation “Possess”.
Thus {skill_name,user_id} is a candidate key.

Candidate Key: {skill_name,user_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Possess” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Experience” Relation:

experience(eid,user_id,company_name, address, mode, duration)

FDs:

{eid} → user_id
{eid} → company_name
{eid} → address
{eid} → mode
{eid} → duration

Closure of {eid}:

{eid}$^+$ ={eid,user_id,company_name,address,mode,duration}

Closure of {eid}covers all the attributes of the Relation “Experience”.
Thus {eid} is a candidate key.

Candidate Key: {eid}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Experience” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Internship_experience” relation:

internship_experience(eid,supervision_level)

FDs:

{eid} → supervision_level

Closure of {eid}:

$\{eid\}^{+}$ ={eid,supervision_level}

Closure of {eid}covers all the attributes of the Relation “Internship_experience”.
Thus {eid} is a candidate key.

Candidate Key: {eid}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Internship_experience” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Job_experience” Relation:

job_experience(eid,seniority_level)

FDs:

{eid} → seniority_level

Closure of {eid}:

$\{eid\}^{+}$ ={eid,seniority_level}

Closure of {eid} covers all the attributes of the Relation “Job_experience”.
Thus {eid} is a candidate key.

Candidate Key: {eid}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Job_experience” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Expectation” relation

Expectation(exp_id,user_id,contract_time,salary,preference_mode)

FDs:

{exp_id,user_id} → contract_time
{exp_id,user_id} → salary
{exp_id,user_id} → preference_mode

Closure of {exp_id,user_id}:

{exp_id,user_id}$^+$ ={exp_id,user_id, preference_mode, salary, contract_time}

Closure of {exp_id,user_id} covers all the attributes of the Relation “Expectation”.
Thus {exp_id,user_id} is a candidate key.

Candidate Key: {exp_id,user_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Expectation” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Degree” Relation:

Degree(specialization,degree)

FDs:

{specialization,degree} → {specialization,degree}

Closure of {specialization,degree}:

$\{specialization,degree\}^{+}$ ={specialization,degree}

Closure of {specialization,degree} covers all the attributes of the Relation “Degree”.
Thus {specialization,degree} is a candidate key.

Candidate Key: {specialization,degree}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Degree” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“College” Relation:

college(college_name,address)

FDs:

{College_name} → address

Closure of {College_name}:

$\{College\_name\}^{+}$ ={College_name,address}

Closure of {College_name} covers all the attributes of the Relation “college”. Thus {College_name} is a candidate key.

Candidate Key: {College_name}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “college” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Completed” Relation:

completed(user_id,specialization,degree,college_name,passout_year,CGPA)

FDs:

{user_id,specialization,degree,college_name} → passout_year
{user_id,specialization,degree,college_name} → CGPA

Closure of {user_id,specialization,degree,college_name}:

{user_id,specialization,degree,college_name}$^+$ = { user_id, specialization, degree, college_name, passout_year, CGPA}

Closure of {user_id, specialization, degree, college_name} covers all the attributes of the Relation “completed”.
Thus {user_id,specialization,degree,college_name} is a candidate key.

Candidate Key: {user_id, specialization, degree, college_name}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “completed” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Offer” Relation:

offer(college_name,specialization,degree)

FDs:

{college_name,specialization,degree} → {college_name,specialization,degree}

Closure of {college_name,specialization,degree}:

{college_name,specialization,degree}$^+$ ={college_name, specialization, degree}

Closure of {college_name,specialization,degree}covers all the attributes of the Relation “Offer”.
Thus {college_name,specialization,degree} is a candidate key.

Candidate Key: {college_name,specialization,degree}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “Offer” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Schooling” Relation:

schooling(seat_no_10th, seat_no_12th, passout_year_12th, school_name_12th, school_name_10th, percentage_10th, percentage_12th, school_address_12th, JEE_percentile)

FDs:

{seat_no_10th} → seat_no_12th
{seat_no_10th} → passout_year_12th
{seat_no_10th} → school_name_12th
{seat_no_10th} → school_name_10th
{seat_no_10th} → percentage_10th
{seat_no_10th} → percentage_12th
{seat_no_10th} → school_address_12th
{seat_no_10th} → JEE_percentile

Closure of {seat_no_10th}:

{seat_no_10th}$^+$ = {seat_no_10th, seat_no_12th, passout_year_12th, school_name_12th, school_name_10th, percentage_10th, percentage_12th, school_address_12th, JEE_percentile}

Closure of {seat_no_10th} covers all the attributes of the Relation “schooling”.
Thus {seat_no_10th} is a candidate key.

Candidate Key: {seat_no_10th}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “schooling” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Company” Relation:

company(cid,company_name,website_link,headquarter_id)

FDs:

{cid} → company_name
{cid} → website_link
{cid} → headquarter_id

Closure of { cid }:

$\{cid\}^{+}$ = {cid, company_name, website_link, headquarter_id}

Closure of {cid}covers all the attributes of the Relation “company”.
Thus {cid} is a candidate key.

Candidate Key:{cid}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “company” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Company_email” Relation:

company_email(cid,email_id)

FDs:

{cid, email_id} →{ cid, email_id }

Closure of { cid }:

$\{cid, email\_id\}^{+} = \{cid, email\_id\}$

Closure of {cid, email_id} covers all the attributes of the Relation “company_email”.
Thus {cid, email_id} is a candidate key.

Candidate Key: {cid, email_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “company_email” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Company_contact_number” Relation:

company_contact_number(cid,contact_number)

FDs:

{cid, contact_number} → {cid, contact_number}

Closure of {cid, contact_number}:

{cid, contact_number} $^{+}$ = {cid, contact_number}

Closure of {cid, contact_number} covers all the attributes of the Relation “company_contact_number”.
Thus {cid, email_id} is a candidate key.

Candidate Key: {cid, contact_number}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “company_contact_number” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Company_login” Relation:

company_login(company_username,password,cid)

FDs:

{company_username} → password
{company_username} → cid

Closure of { company_username } :

$\{ \text{company\_username} \}^{+} = \{ \text{company\_username}, \text{password}, \text{cid} \}$

Closure of { company_username }covers all the attributes of the Relation “company_login”.
Thus { company_username } is a candidate key.

Candidate Key: {company_username}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “company_login” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Offices” Relation:

offices(cid,office_id,city,state)

FDs:

{cid,office_id} → state
{cid,office_id} → city

Closure of {cid,office_id} :

{cid,office_id}$^{+}$={cid,office_id, state, city}

Closure of {cid,office_id} covers all the attributes of the Relation “offices”.
Thus {cid,office_id} is a candidate key.

Candidate Key: {cid,office_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “offices” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Contract” Relation:

contract(contract_number, contract_start_date, offer_id, contract_end_date, min_duration, location, mode, shift)

FDs:

{contract_number,offer_id} → min_duration
{contract_number,offer_id} → contract_end_date
{contract_number,offer_id} → location
{contract_number,offer_id} → mode
{contract_number,offer_id} → shift
{contract_number,offer_id} → contract_start_date

Closure of {contract_number,offer_id} :

{contract_number,offer_id}$^+$ ={contract_number,offer_id, min_duration, contract_end_date, location, mode, shift, contract_start_date}

Closure of {contract_number,offer_id} covers all the attributes of the Relation “contract”.
Thus {contract_number,offer_id} is a candidate key.

Candidate Key: {contract_number,offer_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “contract” is a Super Key.
Thus given relation is in BCNF (Boyce-Codd Normal Form)

“Offers” Relation:

offers(offer_id,cid, skill_required, role, number_of_vacancies, end_date, salary)

FDs:

{offer_id} → cid
{offer_id} →skill_required
{offer_id} → role
{offer_id} → number_of_vacancies
{offer_id} → end_date
{offer_id} → salary

Closure of {offer_id}:

{offer_id}$^+$ ={ offer_id, cid, skill_required, role, number_of_vacancies, end_date, salary}

Closure of {offer_id} covers all the attributes of the Relation “offers”.
Thus {offer_id} is a candidate key.

Candidate Key: {offer_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “offers” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Apply” Relation:

apply(user_id,offer_id,apply_date,status)

FDs:

{user_id, offer_id} → apply_date
{user_id, offer_id} → status

Closure of {user_id, offer_id}:

{ user_id, offer_id }$^+$ ={ user_id, offer_id, apply_date, status}

Closure of {user_id, offer_id} covers all the attributes of the Relation “apply”.
Thus {user_id, offer_id} is a candidate key.

Candidate Key: {user_id, offer_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “apply” is a Super Key.
Thus given relation is in **BCNF (Boyce-Codd Normal Form)**

“Search” Relation:

Search (cid, user_id)

FDs:

{cid, user_id} → {cid, user_id}

Closure of {cid, user_id}:

$\{cid, user\_id\}^+ = \{cid, user\_id\}$

Closure of {cid, user_id} covers all the attributes of the Relation “search”.
Thus {cid, user_id} is a candidate key.

Candidate Key: {cid, user_id}

Since Candidate key is the subset of Super Key

The determinant of all the functional dependencies of the relation “search” is a Super Key.
Thus given relation is in BCNF (Boyce-Codd Normal Form)