

**SUBMITED BY:**

# Uzair Hassan (48525)

# Daniyal Wajid (48528)

# **SUBMITTED TO:**

Sir Talha Tariq

**SECTION:** BSSE-4B

**Project: Job Portal**

**Contents:**

* Introduction
* Objective
* Goals
* Functions
* ERD Diagram
* Relational Schema
* Queries and their Outputs

**Introduction:**

In today's competitive job market, finding the right job or hiring the right talent can be a daunting task. Traditional methods of job hunting and recruitment often lack efficiency and transparency, leading to frustration for both job seekers and employers. Recognizing this need, we propose to develop a state-of-the-art job portal that leverages technology to simplify the job search and recruitment process.

**Objective:**

The objective of this project is to develop a comprehensive job portal platform that connects job seekers with employers efficiently. The platform aims to provide a user-friendly interface for both job seekers and employers, offering features such as job search, resume posting, job posting, application management, and more. By creating a robust job portal, we aim to streamline the job search and recruitment process, ultimately helping job seekers find suitable employment opportunities and assisting employers in finding qualified candidates for their vacancies.

**Goals:**

Organize Data Efficiently:

* Create a structured and normalized database to manage data related to users, resumes, company profiles, job postings, and job applications.
* Ensure data is stored without redundancy, maintaining data integrity.

Facilitate Data Retrieval:

* Enable efficient retrieval of data through well-defined relationships between tables.
* Support queries that allow for comprehensive reports on users, job applications, resumes, companies, and job postings.

Support Data Manipulation:

* Provide mechanisms for adding, updating, and deleting records across various tables.
* Ensure that data manipulations maintain database consistency and integrity.

Implement Data Constraints:

* Apply appropriate constraints such as primary keys, foreign keys, unique constraints, and not null constraints to ensure data validity and integrity.
* Enforce referential integrity between tables using foreign key constraints.

Normalize Data:

* Ensure the database schema is normalized to at least the third normal form (3NF) to eliminate redundancy and avoid anomalies during data operations.

Use Advanced SQL Features:

* Implement advanced SQL queries including joins, subqueries, aggregate functions, and date functions to facilitate complex data retrieval and reporting.
* Demonstrate the ability to handle complex queries and provide meaningful insights from the data.

Enhance Database Security and Integrity:

* Implement secure handling of user data, ensuring sensitive information like passwords is stored securely.
* Apply constraints to enforce data integrity and prevent invalid data entries.

**Core Functions**:

User Management Functions

* CreateUser
* RetrieveUserInformation
* UpdateUserInformation
* DeleteUser

Resume Management Functions

* CreateResume
* RetrieveResume
* UpdateResume
* DeleteResume

Company Profile Management Functions

* CreateCompanyProfile
* RetrieveCompanyProfile
* UpdateCompanyProfile
* DeleteCompanyProfile

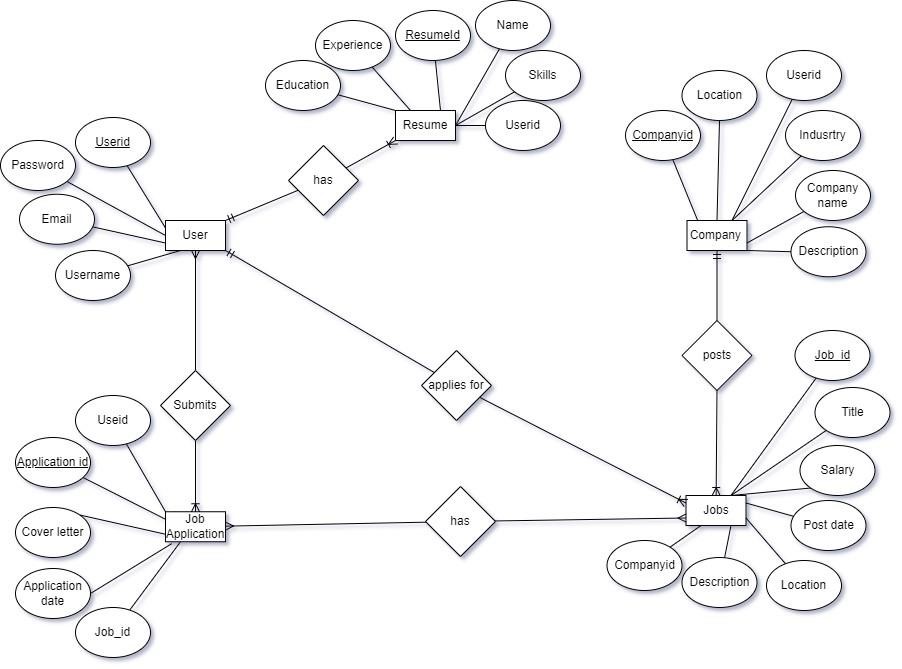
Job Management Functions

* CreateJob
* RetrieveJob
* UpdateJob
* DeleteJob

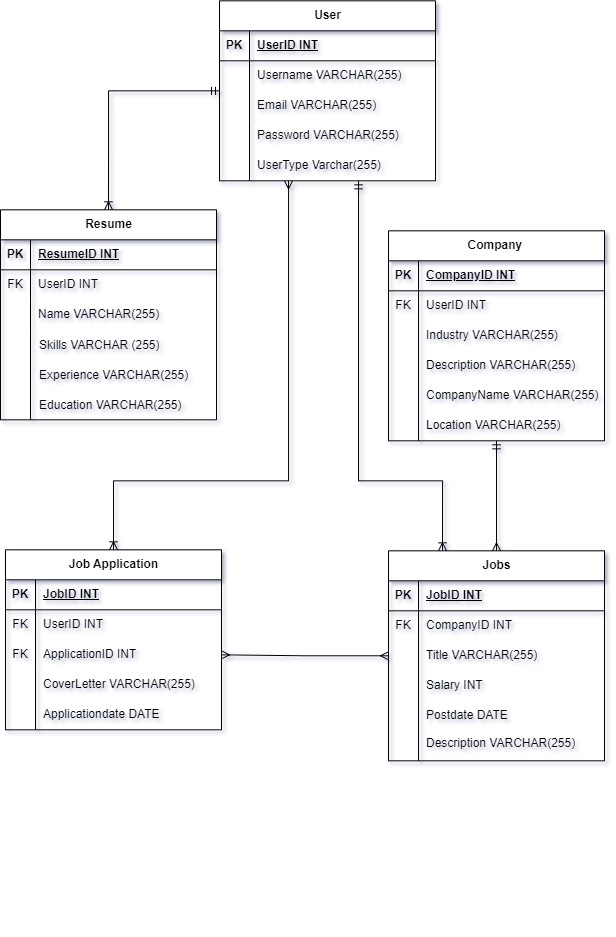
Job Application Management Functions

* SubmitJobApplication
* RetrieveJobApplication
* UpdateJobApplication
* DeleteJobApplication

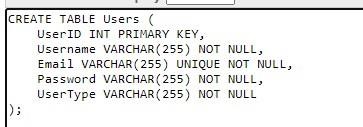
## ERD DIAGRAM

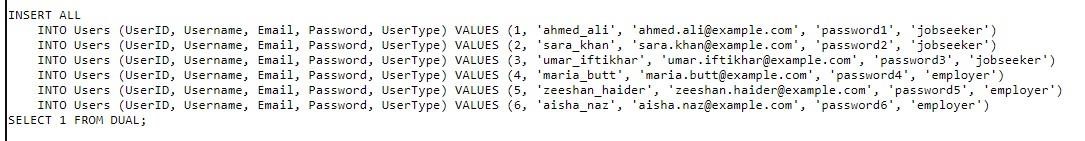


## Relational Schema



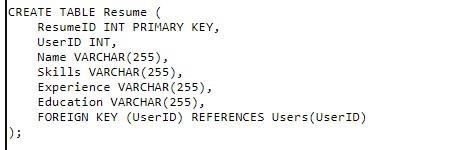
## Create and Insert Data in Users Table

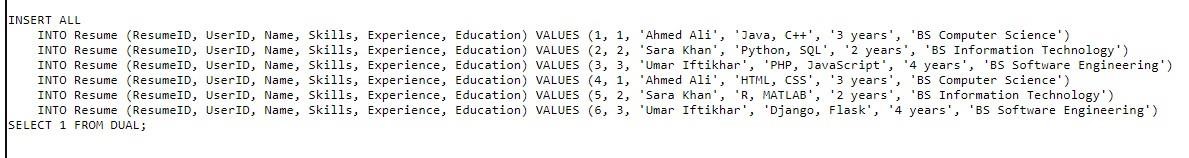


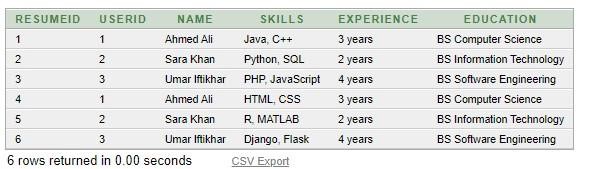




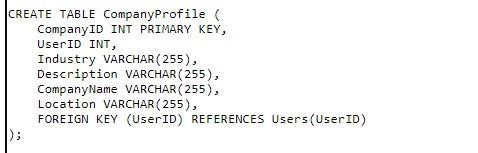
## Create and Insert Data in Resume Table

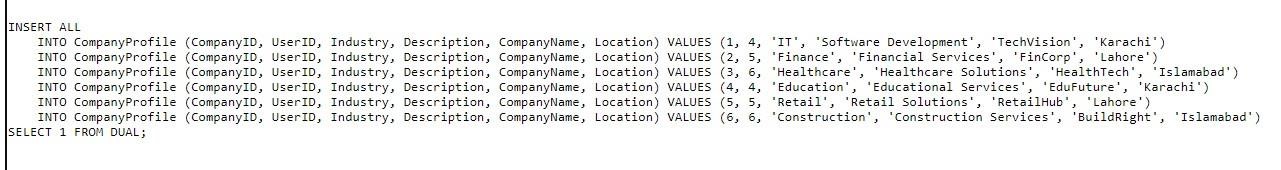






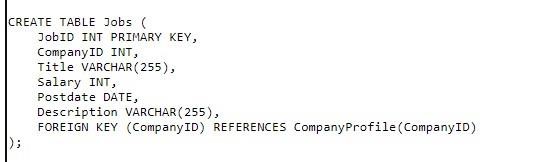
## Create and Insert Data in Company Profile

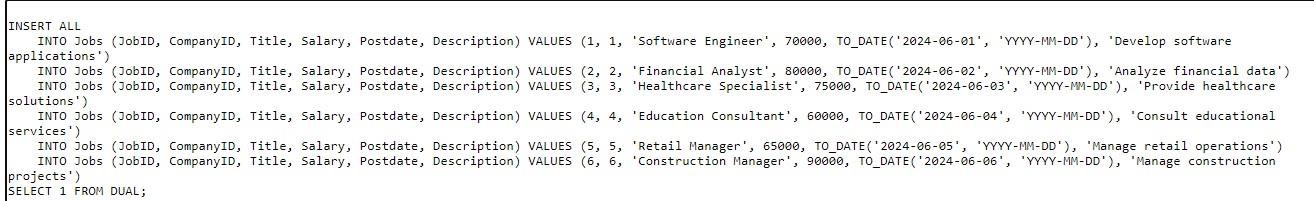






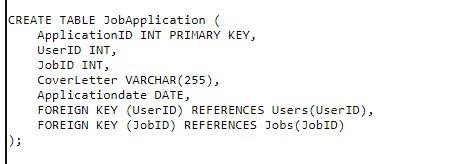
## Create and Insert Data in Jobs

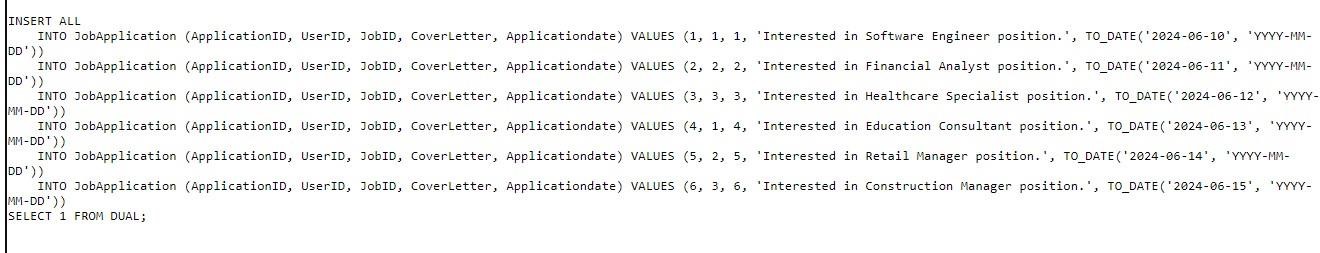






## Create and Insert Data in Jobs Application

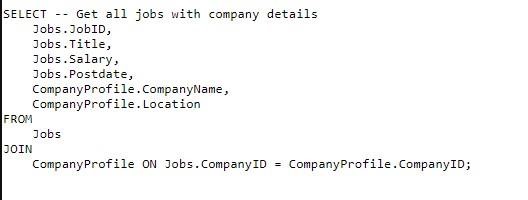


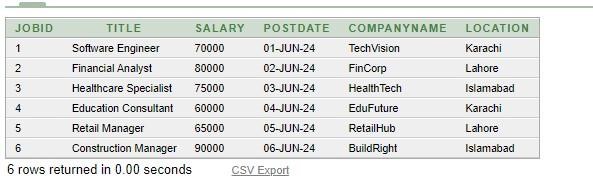




## JOINS Queries

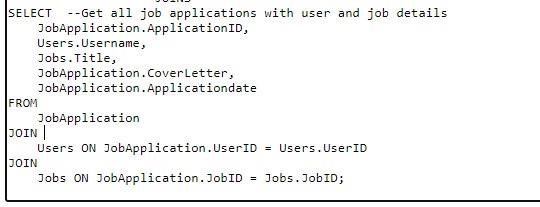
## Get all Jobs with Company Details

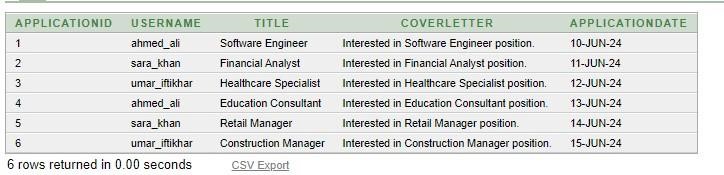




## JOINS Queries

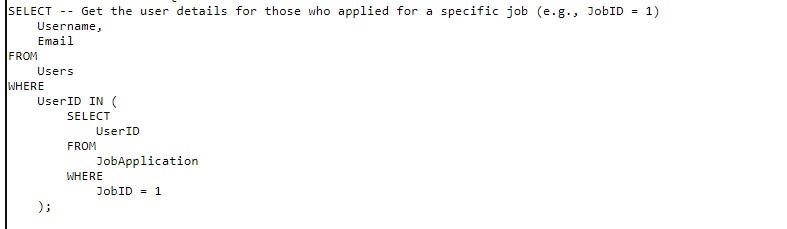
## Get all Job Applicants with User and Job details

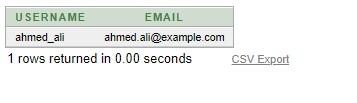




## Nested Queries

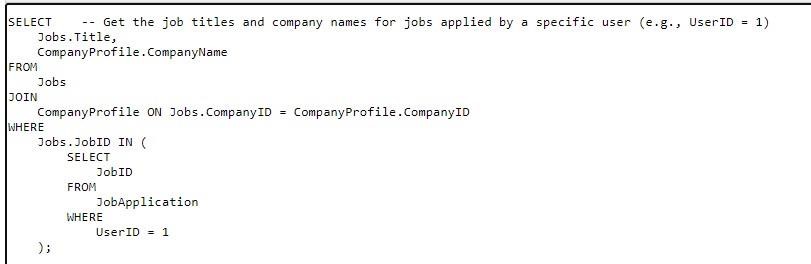
Get the user detail who applied for a specific job having a specific user id





## Nested Queries

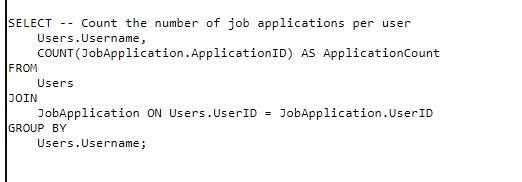
Get the job title and company names for jobs applied by a specific user

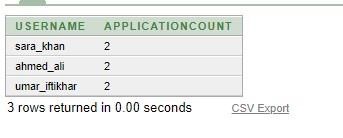




## Aggregate Functions

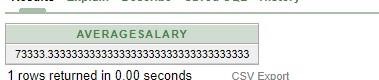
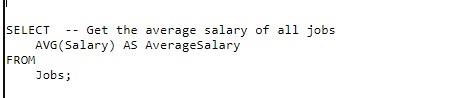
Count the number of job applications per user





## Aggregate Functions

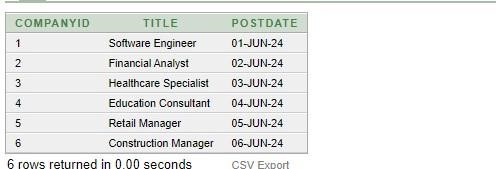
Get the average salary of all jobs



## Nested Queries

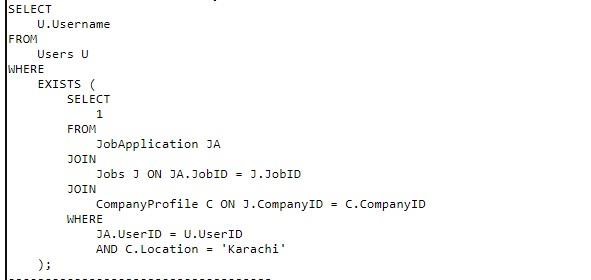
Get the latest job posted by each company

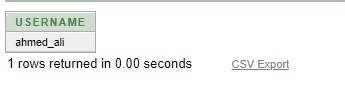




## Co related Query

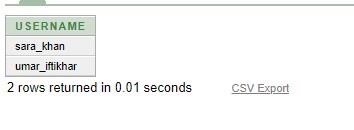
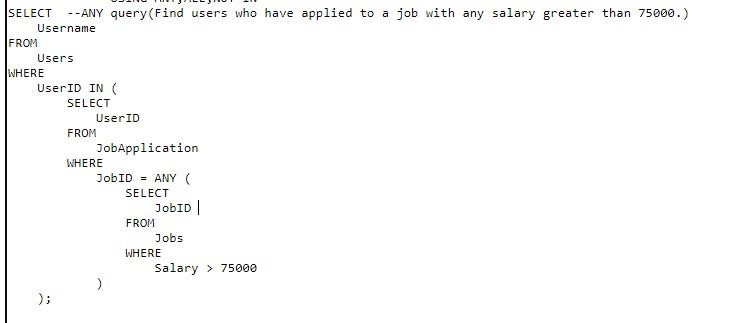
Find users who have applied for jobs in companies located in 'Karachi'.





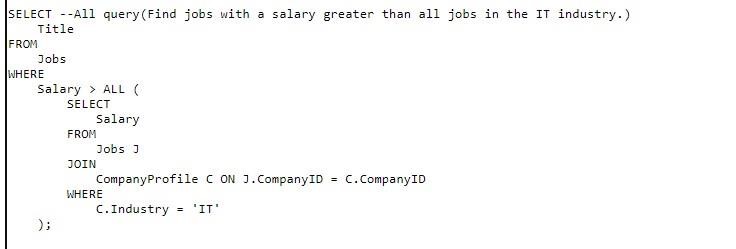
## USING ANY ALL NOT Queries

**Any Query** Find users who have applied to a job with any salary greater than 75000



**All Query**

Find jobs with a salary greater than all jobs in the IT industry.)

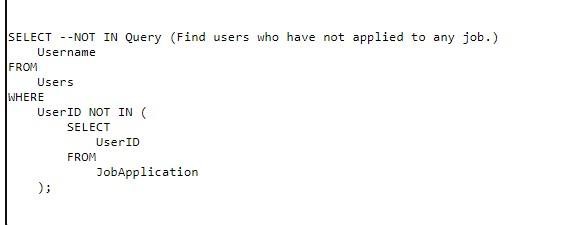




**NOT IN Query**

(Find users who have not applied to any

job.)



**Proof of Normalization**

### 1NF (First Normal Form)

**Definition**: A table is in 1NF if all columns contain atomic, indivisible values, and each column contains values of a single type with unique names.

**Proof for Each Table:**

* **Users Table**:
* **Attributes**: UserID, UserName, UserEmail, UserPassword o
* **Atomic Values**: Each attribute contains only atomic values. o
* **Single Type Values**: Each attribute contains values of a single type.
* **Unique Names**: Each attribute has a unique name.
* **Conclusion**: The Users table is in 1NF.
* **Resumes Table**:
* **Attributes**: ResumeID, UserID, ResumeContent
* **Atomic Values**: Each attribute contains only atomic values.
* **Single Type Values**: Each attribute contains values of a single type.
* **Unique Names**: Each attribute has a unique name.
* **Conclusion**: The Resumes table is in 1NF.
* **Company Profiles Table**:
* **Attributes**: CompanyID, CompanyName, CompanyLocation o
* **Atomic Values**: Each attribute contains only atomic values. o
* **Single Type Values**: Each attribute contains values of a single type.
* **Unique Names**: Each attribute has a unique name. o **Conclusion**: The Company Profiles table is in 1NF.
* **Jobs Table**:
* **Attributes**: JobID, CompanyID, JobTitle, Salary, Location o
* **Atomic Values**: Each attribute contains only atomic values. o
* **Single Type Values**: Each attribute contains values of a single type.
* **Unique Names**: Each attribute has a unique name.
* **Conclusion**: The Jobs table is in 1NF.
* **Job Applications Table**:
* **Attributes**: ApplicationID, UserID, JobID, ApplicationDate o
* **Atomic Values**: Each attribute contains only atomic values. o
* **Single Type Values**: Each attribute contains values of a single type.
* **Unique Names**: Each attribute has a unique name.
* **Conclusion**: The Job Applications table is in 1NF

### 2NF (Second Normal Form)

**Definition**: A table is in 2NF if it is in 1NF and all non-key attributes are fully functionally dependent on the primary key.

**Proof for each Table**

* **Users Table**:
* **Primary Key**: UserID
* **Functional Dependencies**: UserID → UserName, UserEmail, UserPassword
* **Full Dependency**: All non-key attributes fully depend on UserID. **Conclusion**: The Users table is in 2NF.
* **Resumes Table**:
* **Primary Key**: ResumeID
* **Functional Dependencies**: ResumeID → UserID, ResumeContent
* **Full Dependency**: All non-key attributes fully depend on ResumeID.
* **Conclusion**: The Resumes table is in 2NF.
* **Company Profiles Table**:
* **Primary Key**: CompanyID
* **Functional Dependencies**: CompanyID → CompanyName, CompanyLocation
* **Full Dependency**: All non-key attributes fully depend on CompanyID.
* **Conclusion**: The Company Profiles table is in 2NF.
* **Jobs Table**:
* **Primary Key**: JobID
* **Functional Dependencies**: JobID → CompanyID, JobTitle, Salary, Location.
* **Full Dependency**: All non-key attributes fully depend on JobID.
* **Conclusion**: The Jobs table is in 2NF.
* **Job Applications Table**:
* **Primary Key**: ApplicationID
* **Functional Dependencies**: ApplicationID → UserID, JobID, ApplicationDate
* **Full Dependency**: All non-key attributes fully depend on ApplicationID.
* **Conclusion**: The Job Applications table is in 2NF.

### 3NF (Third Normal Form)

**Definition**: A table is in 3NF if it is in 2NF and all attributes are functionally dependent only on the primary key.

**Proof For Each Table:**

* **Users Table**:
* **Primary Key**: UserID
* **Functional Dependencies**: UserID → UserName, UserEmail, UserPassword
* **No Transitive Dependency**: No non-key attribute depends on another non- key attribute.
* **Conclusion**: The Users table is in 3NF.
* **Resumes Table**:
* **Primary Key**: ResumeID
* **Functional Dependencies**: ResumeID → UserID, ResumeContent
* **No Transitive Dependency**: No non-key attribute depends on another non- key attribute.
* **Conclusion**: The Resumes table is in 3NF.
* **Company Profiles Table**:
* **Primary Key**: CompanyID
* **Functional Dependencies**: CompanyID → CompanyName, CompanyLocation
* **No Transitive Dependency**: No non-key attribute depends on another non- key attribute.
* **Conclusion**: The Company Profiles table is in 3NF.
* **Jobs Table**:
* **Primary Key**: JobID
* **Functional Dependencies**: JobID → CompanyID, JobTitle, Salary, Location
* **No Transitive Dependency**: No non-key attribute depends on another non- key attribute.
* **Conclusion**: The Jobs table is in 3NF.
* **Job Applications Table**:
* **Primary Key**: ApplicationID
* **Functional Dependencies**: ApplicationID → UserID, JobID, ApplicationDate
* **No Transitive Dependency**: No non-key attribute depends on another non- key attribute.
* **Conclusion**: The Job Applications table is in 3NF.

### BCNF (Boyce-Codd Normal Form)

**Definition**: A table is in BCNF if it is in 3NF and for every functional dependency (X → Y), X is a superkey. **Proof for Each Table**:

**Proof for Each Table**:

* **Users Table**:
* **Primary Key**: UserID
* **Functional Dependencies**: UserID → UserName, UserEmail, UserPassword.
* **Superkey**: UserID is a superkey.
* **Conclusion**: The Users table is in BCNF.
* **Resumes Table**:
* **Primary Key**: ResumeID
* **Functional Dependencies**: ResumeID→ UserID, ResumeContent
* **Superkey**: ResumeID is a superkey.
* **Conclusion**: The Resumes table is in BCNF.
* **Company Profiles Table**:
* **Primary Key**: CompanyID
* **Functional Dependencies**: CompanyID → CompanyName, CompanyLocation
* **Superkey**: CompanyID is a superkey.
* **Conclusion**: The Company Profiles table is in BCNF.
* **Jobs Table**:
* **Primary Key**: JobID
* **Functional Dependencies**: JobID → CompanyID, JobTitle, Salary, Location
* **Superkey**: JobID is a superkey.
* **Conclusion**: The Jobs table is in BCNF.
* **Job Applications Table**:
* **Primary Key**: ApplicationID
* **Functional Dependencies**: ApplicationID → UserID, JobID, ApplicationDate
* **Superkey**: ApplicationID is a superkey.
* **Conclusion**: The Job Applications table is in BCNF.

**Conclusion**

By thoroughly analyzing and proving the normalization process from 1NF through

BCNF for each table in our job portal database schema, we ensure that our design is efficient, free from redundancy, and maintains data integrity.