A red letter on a black background

Description automatically generated

DSCI-D 532 Applied Database Technologies

Job Quest Log

Adithya Singupati (adisingu@iu.edu)

Malhar Dhopate (mdhopate@iu.edu)

Shalini Kothuru(skothuru@iu.edu)

April 7, 2024

1. **Conceptual Diagram/Schema for Database**
   1. **Diagram**

A screenshot of a computer

Description automatically generated

* 1. **Explanation**

The image shows a database schema with two tables: Users and Jobs. There's a one-to-many relationship between Users and Jobs, indicated by the dashed line, which suggests that a single user can be associated with multiple job applications. This relationship is established through the User\_id field, which is common to both tables.

In practical terms, this means the Jobs table references the Users table to link each job application to a specific user. The Jobs table contains additional details about each job application, such as the job ID, company name, and application status.

1. **Database**
   1. **Tables**

The database for this project is going to be user generated, and consists of the of two (2) tables –

1. **‘Users’** Table – to keep track of the user demographic data. This table will have the following columns –
   1. **‘User\_id’ –** This column will keep track of the unique userid created by the user at the time of account creation.
   2. **‘Email’ –** This column will keep track of the unique emails used by the users at the time of account creation.
   3. **‘First\_name’ –** This Column will store the user’s First Name associated with the account.
   4. **‘Last\_name’ –** This Column will store the user’s Last Name associated with the account.
   5. **‘Password’ –** This Column will store the user password associated with the account.
   6. **‘Gender’ -** It indicates the gender of the column.
   7. **‘Account\_created’ –** This Column will store the Date & Time of the account creation for every user.
   8. **‘Contact\_No’ -** The mobile number of the user given by user at time of signup.
2. **‘Jobs’** Table – to keep track of the variety of job positions applied to by the user.
   1. **‘Id’ –** It is the primary key of this table.
   2. **‘Job\_id’** - Stores a unique or non-unique identifier assigned to each job posting, allowing for the aggregation of multiple records associated with the same job. This facilitates tracking different stages or applications of the same job posting over time.
   3. **‘User\_id’** - the id of the user.
   4. **‘Company\_name’ –** The name of the company or organization name to which user has applied.
   5. **‘Role’ –** The position in the company that user has applied. Ex- Data Scientist, Data Analyst etc.
   6. **‘Location’ –** This column will store the location of the position, eg – Miami, Remote, etc.
   7. **‘URL’ –** This column will store the web URL for the job position, providing easy access and retrieval for the job post for a user.
   8. **‘Job\_source’ –** A column to keep track of the portals used for applications (eg – LinkedIn, Indeed, etc.). This column is for the users analytical purposes to determine which portal has the highest count of applications.
   9. **‘Referral’ -** It is used to indicate if the person has applied using referral.
   10. **‘Application\_id’ -** A unique identifier, either generated by the application system or manually assigned, for tracking specific job applications. This is crucial for managing and following up on applications through various stages of the hiring process.
   11. **‘Status’ –** This column will store the status of the job application, which can be updated as and when a user receives any updates regarding the application. It contains integer value to indicate the phase like submitted, rejected, interview, offer.
   12. **‘Application\_date’ –** This is the date on which the user has applied. User mentions this date.
   13. **Data types, keys and Constraints**

The following constraints have been added to the selected columns for the above tables. The constraints might be updated later if required –

**Users Table:**

|  |  |
| --- | --- |
| **Column name** | **Data Type/key/Constraints** |
| User\_id | INT – Autoincremented value,  Primary key – It serves as a primary key of this table |
| Email | VARCHAR (50), UNIQUE, NOT NULL Constraints to ensure there are no multiple accounts for the same user |
| First\_name | VARCHAR (100), NOTNULL |
| Last\_name | VARCHAR (100), NOTNULL |
| Password | VARCHAR (25), NOTNULL,  Front End constraint - to verify length is more than 8 characters. |
| Gender | CHAR (1), NOTNULL |
| Account\_created | DATE, It stores current\_date at time of the account creation. |
| Contact\_NO | VARCHAR (10) NOTNULL, CHECK length of contact number is 10. |

**Jobs Table:**

|  |  |
| --- | --- |
| **Column name** | **Data Type** |
| Id | INT – Primary Key, Autoincremented value |
| User\_id | INT – FOREIGN KEY – Users (User\_id) |
| Job\_id | VARCHAR (50), NOT NULL |
| Company\_name | VARCHAR (50), NOT NULL |
| Role | VARCHAR (50) NOT NULL |
| Location | VARCHAR (25), NOT NULL |
| URL | VARCHAR (255) |
| Job\_source | VARCHAR (20), NOT NULL |
| Referral | VARCHAR (3), Default value is NO, i.e not applied by referral |
| Application\_id | VARCHAR (50), UNIQUE, NOT NULL |
| Status | VARCHAR (20), NOT NULL |
| Application\_date | DATE, NOT NULL |

1. **Code**

The following code script for table creation, views, along with the constraints can also be found at the - [GitHub](https://github.iu.edu/skothuru/JobTracker) repository (JobQuest Queries.sql), which has been set up to upload all the project-related documentation.

**Database & Table creation (Along with Constraints):**

**-- Creating the 'JobQuest' Database**

**-- Authored -> Malhar Dhopate**

CREATE SCHEMA IF NOT EXISTS JobQuest;

**-- Creating Users Table**

**-- Authored -> Malhar Dhopate**

CREATE TABLE IF NOT EXISTS Users (

User\_id INTEGER PRIMARY KEY AUTO\_INCREMENT,

Email VARCHAR(50) Unique NOT NULL,

First\_Name VARCHAR(100) NOT NULL, Last\_Name VARCHAR(100) NOT NULL,

Contact\_No CHAR(10) UNIQUE NOT NULL,

Password VARCHAR(25) NOT NULL, Gender CHAR(1) NOT NULL,

Account\_Created DATETIME DEFAULT CURRENT\_TIMESTAMP);

**-- Creating Jobs Table**

**-- Authored -> Malhar Dhopate**

CREATE TABLE Jobs (

Id SERIAL PRIMARY KEY AUTO\_INCREMENT,

User\_id INTEGER,

Job\_id VARCHAR(50) NOT NULL,

Company\_name VARCHAR(50) NOT NULL,

Role VARCHAR(50) NOT NULL,

Location VARCHAR(25) NOT NULL,

URL VARCHAR(255) NOT NULL,

Job\_source VARCHAR(20),

Referral VARCHAR(3) DEFAULT "NO",

Application\_id VARCHAR(50) UNIQUE NOT NULL,

Status VARCHAR(20) NOT NULL,

Application\_date DATE NOT NULL

);

**-- Adding Foreign Keys to the Jobs Table**

**-- Authored -> Malhar Dhopate**

ALTER TABLE Jobs ADD CONSTRAINT For\_Key1

FOREIGN KEY (User\_id) REFERENCES Users(User\_id);

**Insertion Query Syntax for Tables:**

**-- Insert into Users, upon Sign Up**

**-- Syntax -- Authored -> Shalini Kothuru**

INSERT INTO Users (Email, First\_name, Last\_Name, Contact\_No, Password, Gender) VALUES (<user\_id>, <email>, <first\_name>, <last\_name>, <contact\_no>, <password>, <gender>);

**-- Inserting data into Jobs table**

**-- user\_id will be retrieved based on login**

**-- Syntax -- Authored -> Shalini Kothuru**

INSERT INTO Jobs (user\_id, Job\_id, Company\_name, Role, Location, URL, Job\_source, Referral, Application\_id, Status, Application\_date)VALUES (<user\_id>, <job\_id>, <Company\_Name>, <Role>, <location>, <URL>, <Job\_source>, <Referral>, <Application\_id>, <Status>, <Application\_date>);

**Update Query Syntax for Tables:**

**-- Update Query Syntax -**

**-- Updating the User Info**

**-- Syntax to Update everything except User\_id & Email & Gender**

**-- Authored -> Adithya Singupati**

UPDATE Users

SET First\_Name = <first\_name>, Last\_Name = <last\_name>, Contact\_No = <contact\_no>, Password = <password>

where User\_id = 1;

**-- Syntax to update only the Application Status**

**-- Authored -> Adithya Singupati**

UPDATE Jobs

SET status = <status> where User\_id = <user\_id> and Job\_id = <job\_id>;

**Index Query User Emails:**

**-- Authored -> Shalini Kothuru**

CREATE UNIQUE INDEX Email\_index ON Users (Email);

**Queries for Creating Views:**

**-- A View to Display the User Profile**

**-- Authored -> Malhar Dhopate**

CREATE VIEW UserProfile AS(

SELECT user\_id, Email, First\_Name || ' ' || Last\_Name AS Full\_Name, Gender FROM users);

**-- A View to display job application for a User**

**-- Authored -> Adithya Singupati**

CREATE VIEW job\_applications AS (

SELECT \* FROM Jobs WHERE user\_id = <user\_id> -- This will be pulled from the user's log in info

ORDER BY application\_date DESC

);

**-- A View to Display the Number of jobs user has applied to in the last month**

**-- Authored -> Shalini Kothuru**

CREATE VIEW num\_of\_jobs\_applied\_monthly as (

SELECT COUNT(\*) AS num\_of\_jobs\_applied\_monthly FROM Jobs

WHERE User\_id = <user\_id> -- This will be pulled from the user's log in info

AND

Application\_date >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 1 MONTH));

**-- A View to Display the Number of jobs user has applied to in the last week**

**-- Authored -> Shalini Kothuru**

CREATE VIEW num\_of\_jobs\_applied\_weekly as (

SELECT COUNT(\*) AS num\_of\_jobs\_applied\_weekly

FROM Jobs

WHERE User\_id = <user\_id> -- This will be pulled from the user's log in info

AND YEAR(Application\_date) = YEAR(CURRENT\_DATE())

AND WEEK(Application\_date) = WEEK(CURRENT\_DATE()));

**-- Ratio of application status - This View Displays the number of applications for each Application Status**

**-- Authored -> Shalini Kothuru**

CREATE VIEW pie\_chart\_ratio as (

SELECT Status, COUNT(\*) AS count

FROM Jobs

WHERE User\_id = <user\_id> -- This will be pulled from the user's log in info

AND Status IN (SELECT DISTINCT Status FROM Jobs WHERE User\_id = <user\_id>)

GROUP BY Status);

**-- Map visualization by number of jobs by locations - This View displays the count of applications for every location**

**-- Authored -> Adithya Singupati**

CREATE VIEW map\_location\_wise as (

SELECT Location, COUNT(\*) AS num\_of\_applications

FROM Jobs

WHERE User\_id = <user\_id> -- This will be pulled from the user's log in info

GROUP BY Location

);

**-- Number of application by referrals - This View Displays the number of job applications done through each Referals.**

**-- Authored -> Adithya Singupati**

CREATE VIEW applications\_by\_referrals as (

SELECT Referral, COUNT(\*) AS num\_of\_applications

FROM Jobs

WHERE User\_id = <user\_id> -- This will be pulled from the user's log in info

AND Referral = 'YES'

GROUP BY Referral);

**-- Number of application by job source - This view Displays the number of job applications by each Job Source.**

**-- Authored -> Malhar Dhopate**

CREATE VIEW bar\_applications\_by\_job\_source as (

SELECT Job\_source, COUNT(\*) AS num\_of\_applications

FROM Jobs

WHERE User\_id = <user\_id> -- This will be pulled from the user's log in info

AND Job\_source IS NOT NULL

GROUP BY Job\_source);

1. **Overall Contribution Summary**

The following table will be updated as we progress towards finishing our project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Tasks** | **Contribution** | **Average Time Spent (Per Milestone)** |
| ALL 3 | * Conceptual design of Schema * Views to provide information to the user | ALL 3 of us have equally contributed to the schema design by connecting on Teams call. Each of us has written different queries to display information to users which are mentioned in SQL code. | 4.5 hours |
| Adithya Singupati | * Schema Explanation * Determining data types * Update queries | In developing the database schema, I clarified entity relationships and chose appropriate data types for the `Users` and `Jobs` tables to ensure data integrity. I also wrote update queries to keep our database accurate, supporting our application tracking system's reliability. | 4.5 hours |
| Malhar Dhopate | * Code reproducibility in MYSQL * Create table queries * user Table description * Error fixing while code reproducibility | Wrote code for Database and table creation, along with the constraints.  Updated the code in the Word file (section 3).  Assisted in Database Schema Creation (section 1).  Assisted in the table schema description (section 2).  Provided Ideas for View development. | 5 hours |
| Shalini Kothuru | * Ideas for visualization in UI * jobs table description * Determining Constraints * Insert and Index queries | Provided ideas for visualizing/presenting different analytics for users through views in UI.  Contributed to database schema. Designed different schemas and considered pros and cons of those schemas.  Determined different constraints and keys for columns in table.  Wrote queries for inserting data and index creation. | 4.5 hours |