

|  |
| --- |
| Business Template  **RECRUITMENT AGENCY** |
|  |

Contents

[1 Business Description 3](#_Toc62212630)

[1.1 Business background 3](#_Toc62212631)

[1.2 Problems. Current Situation 3](#_Toc62212632)

[1.3 The benefits of implementing a database. Project Vision 3](#_Toc62212633)

[2 Model description 3](#_Toc62212634)

[2.1 Definitions & Acronyms 3](#_Toc62212635)

[2.2 Logical Scheme 3](#_Toc62212636)

[2.3 Objects 3](#_Toc62212637)

# 

# Business Description

## Business background

A recruitment agency serves as a liaison between job pursuant or eligible candidates and employers that are looking for appropriate resources. The main responsibility of the recruitment agency is to facilitate the hiring process and associated procedures by matching qualified candidates with respective job vacancies offered by the employers.

Recruitment agency is tasked with reviewing job openings, candidate sourcing and screening, interview coordination and final placement.

To facilitate data management and employment decision making, relational database is required to efficiently store and handle data related to Candidates, Employers, Jobs, Interviews, Offers, etc.

## Problems. Current Situation

Recruitment agency struggles with operational inefficiencies due to outdated data management practices, which invokes additional manual labor, such as:

* Data fragmentation: Candidate profiles, job postings, and application records are often stored in different formats and systems, which requires additional manual intervention.
* Duplicate or inconsistent data: Duplicate records result in inefficient and repetitive data management.
* Tracking deficiencies: Without centralized database, tracking limitation can be noticed as related to interview, offer or employment status, which limits real time reporting as related to the recruitment agency successful performance.

## the Benefits of implementing a database. Project Vision

The implementation of a normalized relational database (3NF) will provide following benefits:

* Centralized data management: A structured database to store and manage information about candidates, jobs, applications, interviews, and placements in a single system.
* Data Integrity: 3NF ensures that data redundancy is eliminated while ensuring consistency across all records.
* Automated workflow: Each process milestone can be tracked in real time during recruitment from application submission to final placement which ensures cost efficiency and eliminates the need of extra manual labor.

Project vision is tightly interconnected with plans to ensure that the recruitment agency database is used as a scalable, efficient and structured data management system which enhances hiring process and improves recruiter productivity while reducing time required to conclude the employment and close the job vacancy. Seamless coordination between all stakeholders that will be ensured by new centralized database is aimed at faster and more efficient hiring decisions.

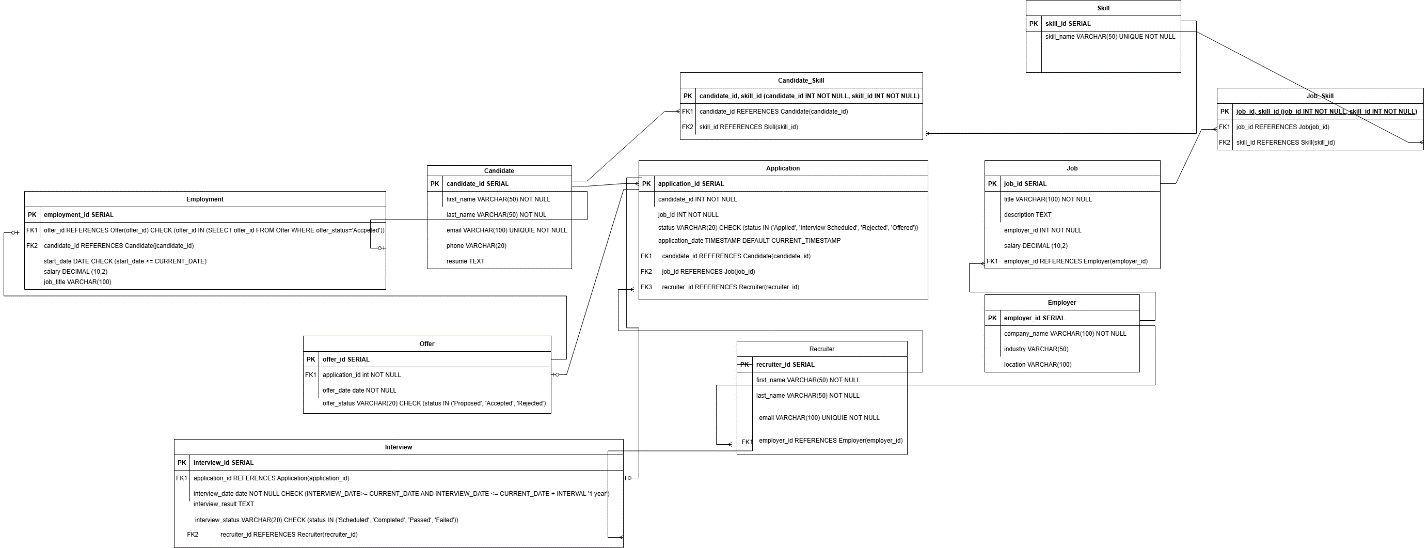
# Model description

## Definitions & Acronyms

This section defines key terms and acronyms used in the recruitment agency database model.

* Candidate: A job pursuant individual which applies for job vacancy via recruitment agency.
* Employer: A company or organization which is seeking to hire candidate for job vacancy.
* Recruiter: Employee of the recruitment agency responsible for matching candidates and job openings which also handles hiring process, from interview to employment.
* Job: A job opportunity posted by an employer via recruitment agency.
* Application: Request submitted by a candidate to be taken into consideration related to a specific job vacancy.
* Interview: A scheduled meeting between a candidate and a recruiter on behalf of the employer as part of the recruitment process.
* Offer: A job offer, extended to a candidate that successfully passed the interview phase.
* Employment: The successful hiring of a candidate for a specific job position.
* Job\_Skill: Refers to a specific ability or expertise required to perform a particular job, e.g. technical, industry specific knowledge, etc. and links job postings with required skills.
* Candidate\_Skill: Refers to a specific competency or ability that a candidate has by associating candidates with their skills.
* Skill: A skill is specific area of expertise that a candidate can possess or that a job vacancy can require, as it can be associated with candidates and job postings.

## Logical Scheme



## Objects

1. **Candidate**

Stores information about job pursuant individuals applying for position.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Candidate | candidate\_id | Unique identifier for the candidate, PK, Auto-increment | SERIAL |
| first\_name | Candidate`s first name, NOT NULL | VARCHAR(50) |
| last\_name | Candidate`s surname, NOT NULL | VARCHAR(50) |
| email | Candidate`s email address, UNIQUE, NOT NULL | VARCHAR(100) |
| phone | Candidate`s phone number | VARCHAR(20) |
| resume | URL to the candidate`s resume | TEXT |

Comments on table relationships

* Candidate and Skill (Many-to-Many) whereas Candidate\_Skill table acts as a junction table  
  A candidate can have multiple skills such as SQL, Python, etc.  
  Likewise a skill can be associated with multiple candidates, e.g. SQL can be a skill for both Candidate X and Candidate Y.  
  The Candidate\_Skill table bridges this many-to-many relationship by linking candidate\_id with skill\_id
* Candidate and Application (One-to-Many)   
  A candidate can apply to multiple job vacancies, and   
  Each application belongs to a specific candidate and corresponds to a specific job.  
  The relationship is one candidate to many applications.
* Candidate and Employment (One-to-Zero/One)  
  A candidate can either not be hired (if their application is unsuccessful) or be hired for a job, leading to one employment record per job application.   
  A job application can result in zero or one employment.

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| candidate\_id | | first\_name | last\_name | email | phone | resume |
| 1 | Jason | | Tesla | jasontesla@gmail.com | 0112122222 | link.com/jasontesla |

1. **Skill**

Stores information about specific skills required for job positions and possessed by candidates.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Skill | skill\_id | Unique identifier for the skill, PK, Auto-increment | SERIAL |
| skill\_name | Name of the skill, e.g. Python, UNIQUE, NOT NULL | VARCHAR(100) |

Comments on table relationships

* The Skill table has many-to-many relationships with both Job and Candidate through the Job\_Skill and Candidate\_Skill bridge tables.   
  Skill and Job (Many-to-Many via Job\_Skill) A job may require multiple skills. A single skill can be required for multiple jobs.  
  Skill and Candidate (Many-to-Many via Candidate\_Skill) A candidate may have multiple skills. A single skill can be possessed by multiple candidates.

Example with data

|  |  |  |
| --- | --- | --- |
| skill\_id | | skill\_name |
| 1 | Python | |

1. **Candidate\_Skill**

Acts as a junction table to represent the many-to-many- relationship between candidates and their skills.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Candidate\_Skill | (candidate\_id, skill id) | Unique identifier for the candidate-skill relationship, PK | Composite |
| candidate\_id | Reference to the candidate possessing the skill (FK, Part of PK), NOT NULL | INT |
| skill\_id | Reference to the skill the candidate possesses (FK, Part of PK), NOT NULL | INT |

Comments on table relationships

* Candidate and Skill (Many-to-Many via Candidate\_Skill)  
  A candidate can have multiple skills.  
  A skill can be possessed by multiple candidates.  
  The Candidate\_Skill table links candidates with their skills.

Example with data

|  |  |  |
| --- | --- | --- |
| candidate\_id | skill\_id | |
| 3 | | 10 | |

1. **Employer**

Stores information about a company or organization which is seeking to hire candidate for job vacancy.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Employer | employer\_id | Unique identifier for the employer, PK, Auto-increment | SERIAL |
| company\_name | Employer`s official name of a company or organization, NOT NULL | VARCHAR(100) |
| industry | Industry sector to which the employer belongs | VARCHAR(50) |
| location | Physical location or registered office of the emplyer | VARCHAR(100) |

Comments on table relationships

* Employer and Job (One-to-Many)   
  An employer can post multiple job vacancies.  
  Each job vacancy is associated with one employer.  
  The Job table stores employer\_id as a foreign key referencing Employer(employer\_id). The relationship is one employer to many job postings
* Employer and Recruiter (One-to-Many)   
  Each employer is assigned one or more recruiters who handle the hiring process on behalf of the employer.  
  A recruiter is responsible for managing jobs vacancies, screening candidates, performing interviews and facilitating employments. The Recruiter table stores employer\_id as a foreign key referencing Employer(employer\_id).  
  The relationship is one employer to many recruiters.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| employer\_id | | company\_name | industry | location |
| 1 | Health Ins Corp | | Healthcare | 13 Wall Street, NY |

1. **Recruiter**

Stores information about a recruiters responsible for managing job applications, interviews, and hiring on behalf of employers.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Recruiter | recruiter\_id | Unique identifier for the recruiter, PK, Auto-increment | SERIAL |
| first\_name | Recruiter`s first name, NOT NULL | VARCHAR(50) |
| last\_name | Recruiter`s surname, NOT NULL | VARCHAR(50) |
| email | Recruiter`s email, UNIQUE, NOT NULL | VARCHAR(100) |
| employer\_id | Reference to the employer the recruiter works for (FK) | INT |

Comments on table relationships

* Recruiter and Employer (Many-to-One)   
  Each recruiter is assigned to an employer.  
  An employer can have multiple recruiters handling recruitment tasks.
* Recruiter and Interview (One-to-Many)   
  Each recruiter can conduct multiple interviews.   
  A single interview is assigned to one recruiter.
* Recruiter and Application (One-to-Many)   
  Each recruiter is responsible for managing multiple job applications.  
  A single job application is assigned to one recruiter. Foreign key Application(recruiter\_id) references Recruiter(recruiter.id).

Example with data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| recruiter\_id | | first\_name | last\_name | email | employer\_id |
| 1 | Maria | | Smith | msmith@epam.com | 87 |

1. **Job**

Stores information about job vacancies posted by employers.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Job | job\_id | Unique identifier for the job, PK, Auto-increment | SERIAL |
| title | Job title (e.g. Data Analyst), NOT NULL | VARCHAR(100) |
| description | Detailed job description | TEXT |
| salary | Salary offered for the position | DECIMAL(10,2) |
| employer\_id | Reference to the employer posting the job (FK) | INT |

Comments on table relationships

* Job and Employer (Many-to-One)   
  Each job belongs to one employer.  
  An Employer can post multiple job vacancies. Tis ensures that each job is linked to a specific hiring company. Foreign key Job(employer\_id) references Employer(employer\_id)
* Job and Job Skill (Many-to-Many)   
  A job may require multiple skills, e.g. SW Engineer may require Python and Java.  
  A single skill can be required by multiple jobs, e.g. SQL may be needed for both Data Analyst and SW Engineer roles.  
  The Job\_Skill table acts as a junction table linking jobs to required skills. Foreign keys establish many to many relationships between Job and Skill.

Example with data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| job\_id | | title | description | salary | Employer\_id |
| 1 | Data Analyst | | Analyzes data and generate reports | 2,000.00 | 2 |

1. **Interview**

Stores information about interviews conducted for job applications.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Interview | interview\_id | Unique identifier for the interview, PK, Auto-increment | SERIAL |
| application\_id | References a specific job application linked to the interview (FK) | INT |
| interview\_date | Scheduled date and time of the interview, NOT NULL CHECK (INTERVIEW\_DATE>= CURRENT\_DATE AND INTERVIEW\_DATE <= CURRENT\_DATE + INTERVAL '1 year') | DATE |
| interview\_result | Short explanation of interview result | TEXT |
| interview\_status | Interview outcome, CHECK (status IN ('Scheduled', 'Completed', 'Passed', 'Failed')) | VARCHAR(20) |
| recruiter\_id | References the recruiter performing the interview (FK) | INT |

Comments on table relationships

* Interview and Application (One-to-One)   
  Each application gets only one interview based on the recruitment agency rule. The Interview table stores an application\_id to connect I to the specific job application it belongs to.
* Recruiter and Interview (One-to-Many)   
  Each recruiter can conduct multiple interviews.  
  A single interview is assigned to one recruiter, who is responsible for evaluating candidates.

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| interview\_id | | application\_id | interview\_date | interview\_result | interview\_status | recruiter\_id |
| 1 | 5 | | 2025-03-11 | Good technical skills | Passed | 3 |

1. **Offer**

Stores information about job offers provided to candidates after the interview process.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Offer | offer\_id | Unique identifier for the offer, PK, Auto-increment | SERIAL |
| application\_id | Reference to the job application that received the offer (FK), NOT NULL | INT |
| offer\_date | The date the offer was made, NOT NULL | DATE |
| offer\_status | Status of the job offer, CHECK (status IN ('Proposed', 'Accepted', 'Rejected')) | VARCHAR(20) |

Comments on table relationships

* Offer and Application (Zero/One-to-One)   
  Each job application can receive either none or only one offer.  
  If a candidate`s application is successful, an offer is made. The Offer table stores application\_id to link the offer to a specific job application.
* Offer and Employment (One-to-Zero/One)   
  A candidate can decline a job offer, but if a candidate accepts the offer, this candidate becomes employed and respective record is added to the Employment table.   
  Each offer can lead to at most one employment record.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| offer\_id | | application\_id | offer\_date | offer\_status |
| 1 | 50 | | 2025-03-17 | Accepted |

1. **Employment**

Stores information about candidates who have accepted job offers thus officially starting employment.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Employment | employment\_id | Unique identifier for the employment, PK, Auto-increment | SERIAL |
| offer\_id | Reference to the candidate who got employed (FK), CHECK (offer\_id IN (SELECT offer\_id FROM Offer WHERE offer\_status='Accpeted')) | INT NOT NULL |
|  | candidate\_id | Reference to the job offer that led to the employment (FK) | INT NOT NULL |
|  | start\_date | Date when the candidate starts the job, CHECK (start\_date <= CURRENT\_DATE) | DATE |
|  | salary | Salary offered for the job position | DECIMAL(10,2) |
|  | job\_title | Title of the job for which the candidate is employed | VARCHAR(100) |

Comments on table relationships

* Employment and Offer (One-to-One)   
  A job offer must be accepted before an employment record is created.   
  Each accepted offer leads to one employment record.
* Employment and Candidate (One-to-One)   
  A candidate can have at most one employment record at a time.  
  The Employment table links the candidate who got the job.

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| employment\_id | | offer\_id | candidate\_id | start\_date | salary | job\_title |
| 10 | 52 | | 7 | 2025-03-25 | 3,000.00 | Data Analyst |

1. **Application**

Stores information about job applications submitted by candidates for various job positions.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Application | application\_id | Unique identifier for the application, PK, Auto-increment | SERIAL |
| candidate\_id | Reference to the candidate who applied (FK), NOT NULL | INT |
|  | job\_id | Reference to the job being applied for (FK), NOT NULL | INT |
|  | recruiter\_id | Reference to the recruiter handling the application (FK), NOT NULL | INT |
|  | status | Current status of the application, CHECK (status IN ('Applied', 'Interview Scheduled', 'Rejected', 'Offered')) | VARCHAR(20) |
|  | application\_date | Date when the application was submitted, DEFAULT CURRENT\_TIMESTAMP | TIMESTAMP |

Comments on table relationships

* Application and Candidate (One-to-Many)  
  A candidate can apply to multiple job vacancies.  
  Each job application is linked to a specific candidate.
* Application and Recruiter (One-to-Many)  
  Each application is assigned to a recruiter.  
  A recruiter can manage multiple applications.
* Application and Offer (One-to-Zero/One)  
  Each job application can receive either zero or one job offer.   
  If an application is successful, an offer is created and linked to that application.  
  If the application is rejected, no offer is granted. Foreign key Offer(application\_id) references Application(application\_id)
* Application and Interview (One-to-One)  
  Each job application can receive either zero or one interview.  
  If a recruiter schedules an interview for an application, the interview record is created and linked to that application.  
  If the application is rejected before an interview, no interview record is created. Foreign key Interview(application\_id) references Application(application\_id).

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| application\_id | | candidate\_id | job\_id | recruiter\_id | status | application\_date |
| 1 | 3 | | 59 | 9 | Applied | 2025-03-17 14:32:00 |

1. **Job\_Skill**

Acts as a junction table to represent the many-to-many- relationship between jobs and required skills.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Job\_Skill | job\_skill\_id | Unique identifier for the job-skill relationship, PK, Auto-increment | SERIAL |
| job\_id | Reference to the job that requires the skill (FK), NOT NULL | INT |
| skill\_id | Reference to the required skill (FK), NOT NULL | INT |

Comments on table relationships

* Job and Skill (Many-to-Many via Job\_Skill)  
  A job can require multiple skills.  
  A skill can be needed for multiple jobs.   
  The Job\_Skill table links jobs to the required skills.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| job\_skill\_id | | job\_id | skill\_id |
| 1 | 3 | | 5 |