

|  |
| --- |
| Business Template  **Mondo recruitment Company** |
|  |

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

Mondo is the largest national staffing agency specializing exclusively in high-end, niche IT, Tech, and Digital Marketing talent. For nearly 20 years, Mondo has delivered solutions that bridge the talent gap and accelerate technology innovation for global companies. With a team of experienced recruiters, the agency aims to streamline the hiring process for both candidates and employers, providing a platform for job postings, applications, and candidate evaluations.

## Problems. Current Situation

The current application and candidate tracking processes are predominantly manual, leading to errors, inefficiencies, and delays in communication. The absence of a centralized database results in fragmented data across spreadsheets and documents, making it difficult to access candidate information, job postings, and application statuses. As the agency grows, managing an increasing number of candidates and job postings becomes increasingly challenging without an automated system. Generating reports on hiring trends, candidate demographics, and application statuses is time-consuming and often inaccurate due to manual data compilation.

## the Benefits of implementing a database. Project Vision

Database implementation includes automating the application and tracking processes, that will reduce errors and improve response times for candidates and employers. Also, a single database will allow easy access to up-to-date candidate and job information, enhancing decision-making capabilities. The database will facilitate the management of a growing number of candidates and job postings without compromising performance, which means improved scalability.

Another benefit of implementing a database is the ability to generate automated reports, that will provide insights into hiring trends and performance metrics, enabling strategic planning.

And lastly, the vision for this project is to create a robust and scalable database system that enhances the efficiency of the recruiting agency, ultimately leading to better matches between candidates and employers and improved overall satisfaction.

# Model description

## Definitions & Acronyms

Candidate: A person applying for a job.

Job: An employment opportunity posted by a company.

Company: An organization offering job positions.

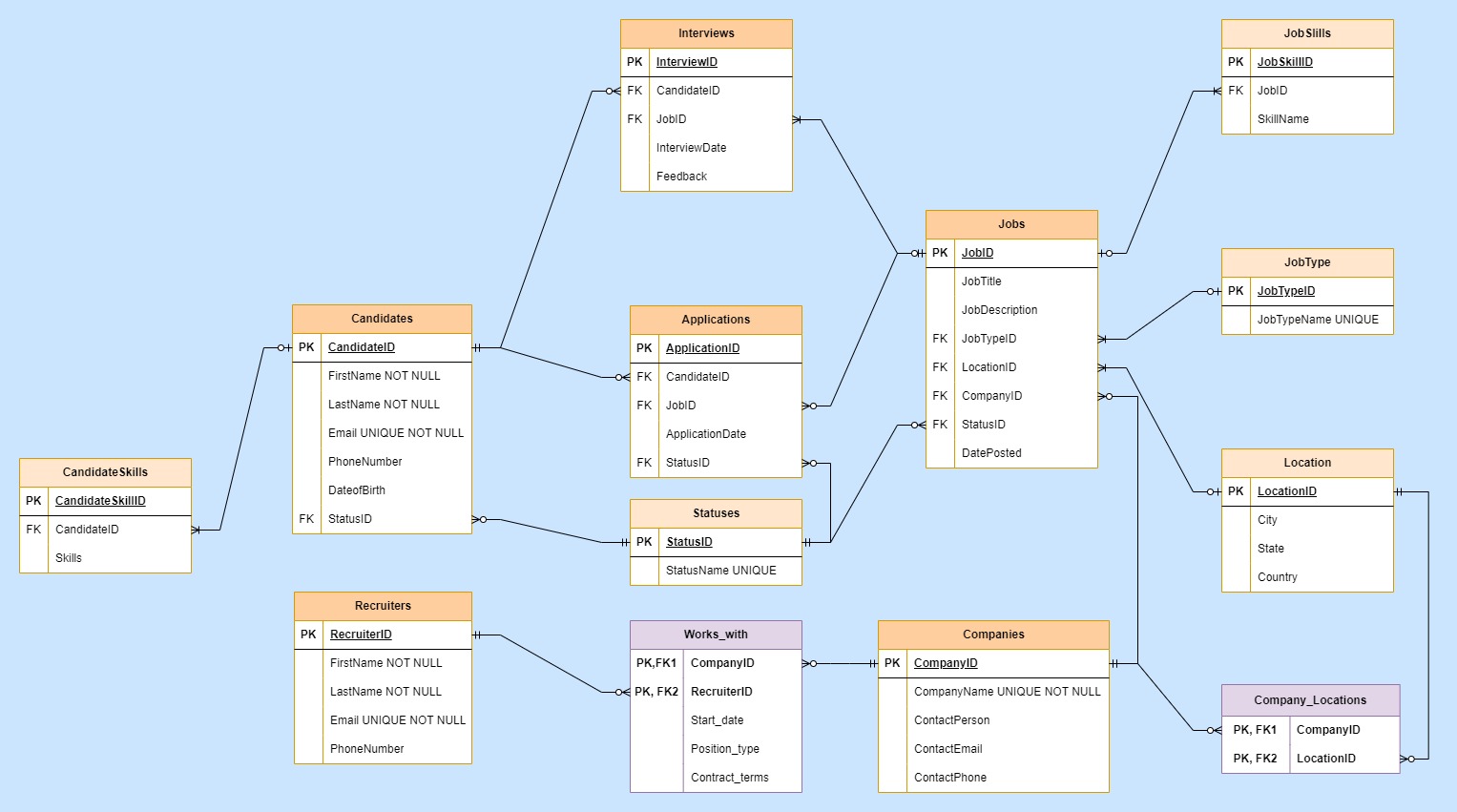
Recruiter: An employee of the recruitment agency responsible for managing job applications and client relationships.

Application: The process of a candidate applying for a job.

Interview: A step in the hiring process, where a candidate meets with the company for a job.

Status: The current state of a job, candidate, or application (e.g., "Open", "Hired", "Rejected").

## Logical Scheme



## Objects

**Candidates**

Stores information about job seekers

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Candidates | CandidateID | Unique identifier for candidate, PK | Int |
| FirstName | First name of the candidate | Text |
|  | LastName | Last name of the candidate | Text |
|  | Email | Email address of the candidate | Text |
|  | PhoneNumber | Contact number of the candidate | Text |
|  | Resume | URL to candidate's resume | Text |
|  | DateOfBirth | Candidate's date of birth | Date |
|  | StatusID | Candidate status, FK | Int |

Comments on table relationships

Candidates and Jobs have a many-to-many relationship via Applications. Each candidate can apply to multiple jobs, and each job can have multiple candidates.

Example with data

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| CandidateID | FirstName | LastName | Email | PhoneNumber | Resume | DateOfBirth | StatusID |
| 1 | John | Doe | john.doe@email.com | 123-456-7890 | john\_resume.pdf | 1990-01-01 | 1 |

Companies

Stores company information, including contact details.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Companies | CompanyID | Unique identifier for company, PK | Int |
| CompanyName | Name of the company | Text |
| ContactPerson | Contact person at the company | Text |
| ContactEmail | Email address of the contact | Text |
| ContactPhone | Contact number of the company | Text |

Comments on table relationships

Jobs and Companies have a many-to-one relationship. Each job is posted by one company, but a company can post multiple jobs.

Locations are referenced to Companies tables with Many-to-Many Relationship as one location can be associated with multiple companies and a company can be based in one location or more as companies also can have many locations in different countries for example. Company\_Locations table resolves the many-to-many relationship, allowing a company to operate in multiple locations and multiple companies to share the same location with creating One-to-Many relationships between the two related tables.

Recruiter and Companies have a many-to-many relationship, which is resolved by the Works\_With junction table. A recruiter working at the recruitment agency may manage or help multiple clients (companies) fill various job positions as recruiters often have a portfolio of clients, they are working with to help fill job positions across different industries or job roles. Clients (companies) sometimes work with more than one recruiter to ensure they have access to a wide pool of candidates or different expertise. Companies has a one-to-many relationship with Works\_With (since a company can work with many recruiters).

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CompanyID | CompanyName | ContactPerson | ContactEmail | ContactPhone |
| 1 | TechCorp | Jane Smith | info@techcorp.com | 555-1111 |

Jobs

Contains job postings from companies.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Jobs | JobID | Unique identifier for the job, PK | Int |
| JobTitle | Title of the job | Text |
| JobDescription | Description of the job | Text |
| JobTypeID | Job type (FK to JobTypes) | Int |
| CompanyID | Company offering the job (FK) | Int |
| LocationID | Location of the job (FK) | Int |
| DatePosted | Date when the job was posted | Date |

Comments on table relationships

Jobs and Companies have a many-to-one relationship. Each job is posted by one company, but a company can post multiple jobs. Applications and Jobs have a many-to-one relationship. One job can receive multiple applications, but each application is for one specific job. Jobs and Interviews have one-to-many relationship, as a job can have many interviews, but each interview relates to one job.

Jobs and JobSkills have a one-to-many relationship. Each job can require multiple skills, but each skill record relates to one job. Jobs and JobTypes have a one-to-many relationship. Each job type can be assigned to multiple job postings. Jobtypes table is referenced by the Jobs table, where each job is associated with a job type. Each job has a status (e.g., active, closed), but multiple jobs can share the same status type, therefore they has Many-to-One relationship. Location and Jobs has one-to-many relationship as each job is posted in one location, but a location can have many jobs.

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| JobID | JobTitle | JobDescription | JobTypeID | CompanyID | LocationID | DatePosted |
| 1 | Software Engineer | Develop and maintain applications | 1 | 1 | 1 | 2024-09-15 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

Applications

Tracks the jobs to which candidates have applied.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Applications | ApplicationID | Unique identifier for the application, PK | Int |
| CandidateID | Candidate applying (FK to Candidates) | Int |
| JobID | Job being applied for (FK to Jobs) | Int |
| ApplicationDate | Date the application was submitted | Date |
| StatusID | Status of the application (e.g., Hired) | Int |

Comments on table relationships

Applications and Jobs as well as Applications and Candidates have a many-to-one relationship. One job can receive multiple applications, but each application is for one specific job and one candidate can apply to multiple jobs, but each application is for a specific candidate and job.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ApplicationID | CandidateID | JobID | ApplicationDate | StatusID |
| 1 | 1 | 1 | 2024-10-01 | 1 |

Recruiters

Stores details about recruitment agents.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Recruiters | RecruiterID | Unique identifier for the recruiter, PK | Int |
| FirstName | First name of the recruiter | Text |
| LastName | Last name of the recruiter | Text |
| Email | Email address of the recruiter | Text |
| PhoneNumber | Contact number of the recruiter | Text |

Comments on table relationships

Recruiter and Companies have a many-to-many relationship. A recruiter working at the recruitment agency may manage or help multiple clients(companies) fill various job positions as recruiters often have a portfolio of clients they are working with to help fill job positions across different industries or job roles. Clients (companies) sometimes work with more than one recruiter to ensure they have access to a wide pool of candidates or different expertise. For not violating 3NF and normalization purposes a junction table (Works\_with table) was created. Recruiter has a one-to-many relationship with Works\_with (since a recruiter can work with many companies) and Company has a one-to-many relationship with Works\_with (since a company can work with many recruiters).

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RecruiterID | FirstName | LastName | Email | Phone |
| 1 | Alice | Johnson | alice@techcorp.com | 555-3333 |

JobSkills

Tracks the required skills for each job.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| JobSkills | JobSkillID | Unique identifier for the job skill, PK | Int |
| JobID | Job that requires the skill (FK) | Int |
| SkillIName | Name of the skill | Text |

Comments on table relationships

Jobs and JobSkills have a one-to-many relationship. Each job can require multiple skills, but each skill record relates to one job.

Example with data

|  |  |  |
| --- | --- | --- |
| JobSkillID | JobID | SkillName |
| 1 | 1 | Python |
| 2 | 1 | SQL |
| 3 | 2 | Data Analysis |

CandidateSkills

Tracks the skills of each candidate.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| CandidateSkills | CandidateSkillID | Unique identifier for candidate skill, PK | Int |
| CandidateID | Candidate possessing the skill (FK) | Int |
| SkillName | Name of the skill | Text |

Comments on table relationships

Each candidate can have multiple skills, but each skill record relates to one candidate, therefore there is One-to-Many relationship between Candidates and CandidateSkills.

Example with data

|  |  |  |
| --- | --- | --- |
| CandidateSkillID | CandidateID | SkillName |
| 1 | 1 | Java |
| 2 | 1 | Python |
| 3 | 2 | SQL |

Interviews

Stores interview details for candidates and jobs.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Interviews | InterviewID | Unique identifier for the interview, PK | Int |
| CandidateID | Candidate being interviewed (FK) | Int |
| JobID | Job being interviewed for (FK) | Int |
| InterviewDate | Date of the interview | Date |
| Feedback | Feedback from the interview | Text |

Comments on table relationships

A job can have many interviews, but each interview relates to one job. A candidate can attend multiple interviews, but each interview is for one candidate. In each case, it describes a One-to-Many relationship.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| InterviewID | CandidateID | JobID | InterviewDate | StatusID |
| 1 | 1 | 1 | 2024-10-05 | 1 |
| 2 | 2 | 2 | 2024-10-07 | 2 |

Statuses

Stores status information for jobs, applications, and candidates.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Statuses | StatusID | Unique identifier for the status, PK | Int |
| StatusName | Name of the status | Text |

Comments on table relationships

Each job has a status (e.g., active, closed), but multiple jobs can share the same status type, therefore they has Many-to-One relationship. Each application has a status (e.g., pending, approved), but multiple applications can share the same status type. Each candidate has a status (e.g., active, inactive), but multiple candidates can share the same status type.

Example with data

|  |  |
| --- | --- |
| StatusID | StatusName |
| 1 | Active |
| 2 | Pending |
| 3 | Interviewing |

Jobtypes

Stores different types of jobs (Full-time, Part-time, etc.).

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Jobtypes | JobTypeID | Unique identifier for the job type, PK | Int |
| JobTypeName | Name of the job type | Text |

Comments on table relationships

Jobs and JobTypes have a one-to-many relationship. Each job type can be assigned to multiple job postings.

Jobtypes table is referenced by the Jobs table, where each job is associated with a job type.

Example with data

|  |  |
| --- | --- |
| JobTypeID | JobTypeName |
| 1 | Full-time |
| 2 | Part-time |
| 3 | Contract |

Locations

Stores location details for job postings and companies.

|  |  |  |  |
| --- | --- | --- | --- |
| Locations | Field name | Field Description | Data Type |
|  | LocationID | Unique identifier for the location (PK) | Int |
| City | Name of the city | Text |
| State | Name of the state or province | Text |
| Country | Name of the country | Text |

Comments on table relationships

Locations are referenced to Companies tables with Many-to-Many Relationship as one location can be associated with multiple companies and a company can be based in one location or more as companies also can have many locations in different countries for example. Company\_Locations table resolves the many-to-many relationship, allowing a company to operate in multiple locations and multiple companies to share the same location with creating One-to-Many relationships between the two related tables.

Location and Jobs has one-to-many relationships as each job is posted in one location, but a location can have many jobs.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| LocationID | City | State | Country |
| 1 | New York | NY | USA |

Works\_with

Junction table between Companies and Recruiters table

|  |  |  |  |
| --- | --- | --- | --- |
| Works\_with | Field name | Field Description | Data Type |
|  | RecruiterID | Unique identifier for the recruiter, PK,FK1 | Int |
| CompanyID | Unique identifier for company, PK, FK2 | Int |
| StartDate | Date when the recruiter started working with the company | Date |
|  | Position\_type | The type of job position the recruiter is helping to fill | Text |
|  | Contract\_terms | The terms of the recruitment contract (e.g., 6 months) | Text |

Comments on table relationships

Recruiters and Companies tables have a many-to-many relationship, which is resolved by the Works\_With junction table. Recruiters table has a one-to-many relationship with Works\_With (since a recruiter can work with many companies) and Companies has a one-to-many relationship with Works\_With (since a company can work with many recruiters).

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RecruiterID | CompanyID | StartDate | Position\_type | Contract\_terms |
| 101 | 201 | 2023-05-01 | Software Engineer | 6-month contract |
| 101 | 202 | 2023-06-15 | Data Analyst | 1-year contract |
| 102 | 201 | 2023-07-01 | Project Manager | 1-year contract |
| 103 | 203 | 2024-01-10 | Marketing Manager | 3-month contract |
| 101 | 203 | 2023-11-01 | Data Scientist | Permanent |

Company\_Locations

Junction table between Companies and Locations table

|  |  |  |  |
| --- | --- | --- | --- |
| Company\_Locations | Field name | Field Description | Data Type |
|  | LocationID | Unique identifier for the location, PK, FK1 | Int |
| CompanyID | Unique identifier for company, PK, FK2 | Int |

Comments on table relationships

Locations and Companies tables have a many-to-many relationship. Company\_Locations table resolves the many-to-many relationship, allowing a company to operate in multiple locations and multiple companies to share the same location with creating One-to-Many relationships between the two related tables.

Example with data

|  |  |
| --- | --- |
| CompanyID | LocationID |
| 1 | 1 |
| 2 | 2 |
| 1 | 2 |
| 3 | 3 |
| 2 | 3 |