

# Database Management System

## Project

5/7/2022

## Project Report

### PAYROLL & HRM DATABASE

#### Analyzed Application URL

<https://aligauhar-trials7501.orangehrmlive.com/auth/seamlessLogin> ---for Promotion Panel

[https://www.google.com/url?sa=t&source=web&rct=j&url=https://www.grovehr.com/&ved=2ahUKEwi0ptPv\\_-H4AhXUwQIHbtUCicQFnoECAgQAQ&usg=AOvVaw1QwDwcWHCUJ0-JmVKvolrb](https://www.google.com/url?sa=t&source=web&rct=j&url=https://www.grovehr.com/&ved=2ahUKEwi0ptPv_-H4AhXUwQIHbtUCicQFnoECAgQAQ&usg=AOvVaw1QwDwcWHCUJ0-JmVKvolrb) – for Hiring Panel

<https://www.excel-skills.com/payroll-leave-templates.php?sesreq=7700416202207051456&ctime=1657032979> --- for Payroll panel

#### Class: BSCS 4 (A)

#### Group Members

Student Name	Enrollment	Viva Marks
Ali Gauhar	02-134202-006	

#### Project Marks

Head	Performance	Comments
Analysis & Report		
ERD		
Normalization		
DDL/DML/Triggers		
Stored Proc/ Views/ Stored Functions		

## INTRODUCTION

The data base management system we are going to make covers 3 major sides pay role, recruitment and promotion panel. This is done because HRM covers many aspects therefore breakdown our project in easy format we do this. We will include all the part of HRM except payrolls and hiring will be covered in promotion panel as much as we can

Since promotion is related to performance of employee therefore we have made our system in such a way that whenever any manager will start analyzing about a particular employee, he will easily find all the information and progress of the person who is going to be promoted, this can also be automated if the criteria of companies promotion is defined

## This is only promotion panel data till page # 38

### Analysis – Screenshots of the Application

***	Username ↑	User Role(s)	Employee Name	Status	Regions
	admin	Global Admin		Enabled	

Rows per page: 50 | 1 - 1 of 1

Here the HR administration page contains the Entities (Users, Manage User Role, Job, organization, Announcements, Configurations, Audit Traits, Assets)

***	Username ↑	User Role(s)	Employee Name	Status	Regions
	admin	Global Admin		Enabled	

Rows per page: 50 | 1 - 1 of 1

User(User\_Name, User\_Role(multivalued), Employee\_Name, Status, Region)

User Role ↑	User Role Type
Default ESS	ESS
Default Supervisor	Supervisor

Rows per page: 50 | 1 - 2 of 2

Manage\_User\_Role(user\_Role, User\_Role\_Type)

Job

Organization

- Manage Salary Components
- Manage Job Titles
- Manage Pay Grades
- Manage Employment Status
- Manage Job Categories
- Manage Work Shifts

inside job bar there are more entities  
 (Manage\_Salary\_Components, Manage\_Job\_Type,  
 Manage\_Pay\_Grades, Manage\_Employement\_Status,  
 Manage\_Pay\_Grades, manage\_Job\_Category,  
 Manage\_Work\_category, Manage\_Work\_Shifts)

Component Name ↑	Type	Part of Total Payable?	Cost to Company?
Annual Basic Payment	Earning	Yes	Yes

Rows per page: 50 | 1 - 1 of 1

Manage\_Salary\_Component(Component\_Name, Type, Part\_of\_Total\_Payable?, Cost\_TO\_Company?)

Job Title ↑	Job Description

Rows per page: 50 | 0 - 0 of 0

Manage\_Job\_Titles(Job\_Title, Job\_Discription)

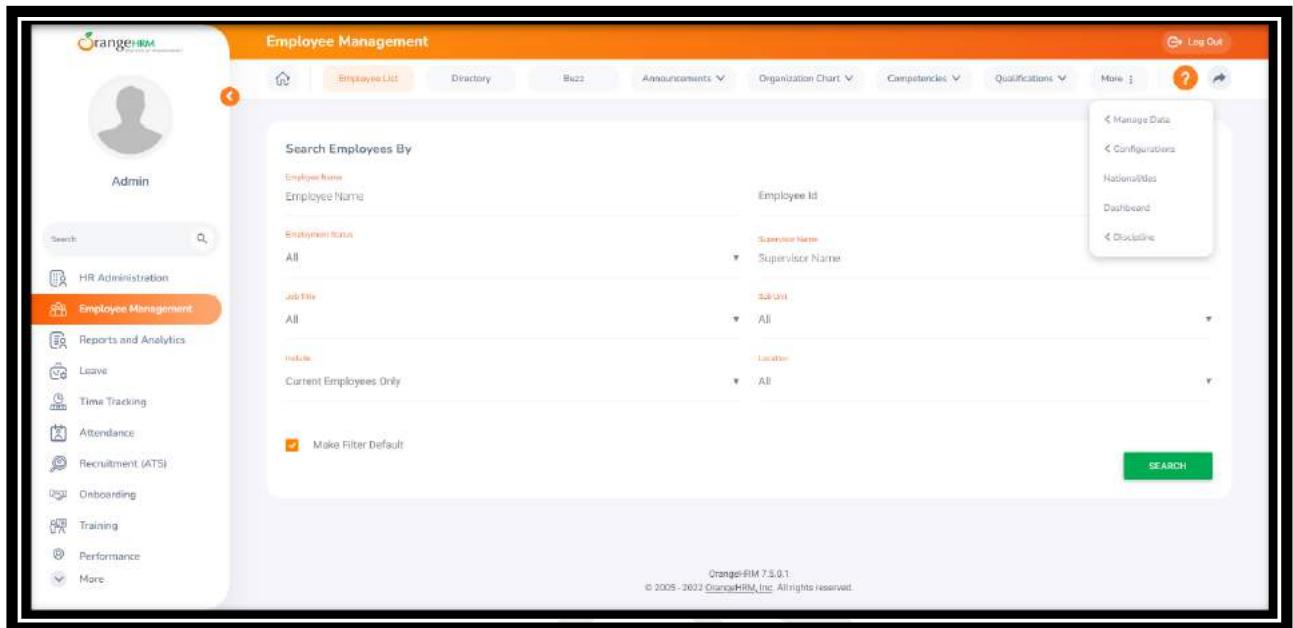
### Manage\_Pay\_Grades(Pay\_Grades,Currency)

### Manage\_Employment\_Status(Employment\_Status)

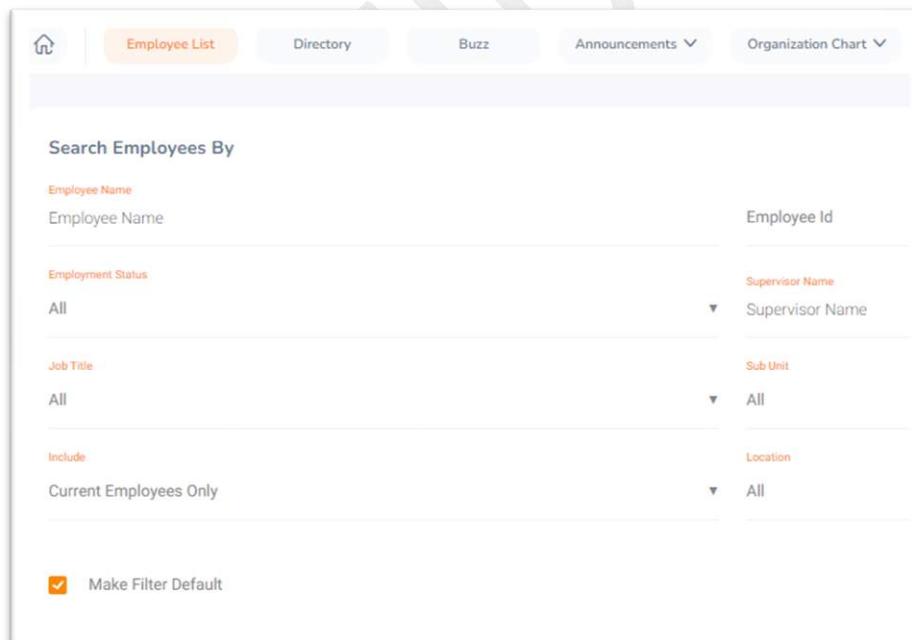
Activate Windows  
Go to Settings to activate Windows.  
Rows per page: 50 ▾ 1 - 9 of 9

### Manage\_Job\_Category(Job\_Category(multivalued))

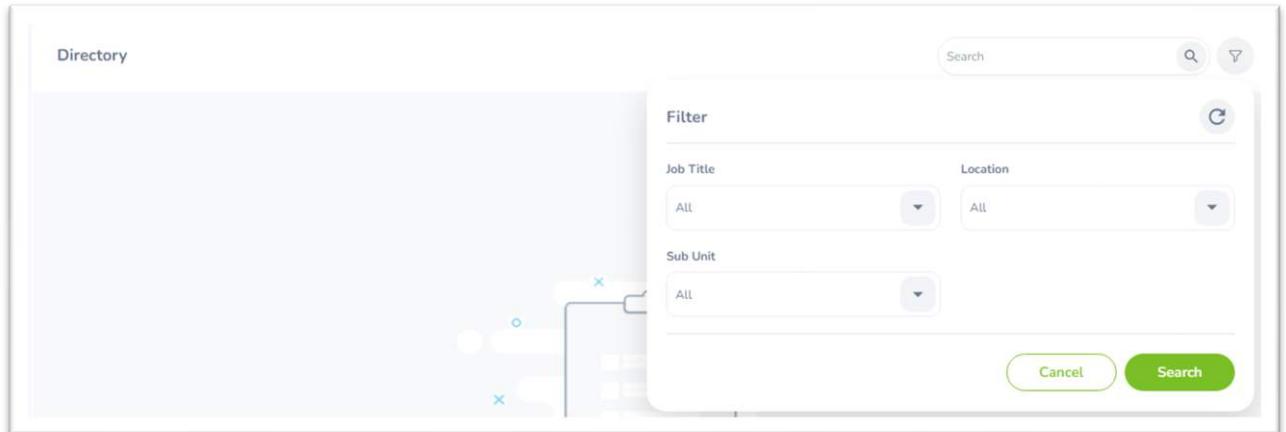
Manage\_Work\_Shift(Work\_Shift, From, To, Hours\_Per\_Day(dependent on days—derived attribute)



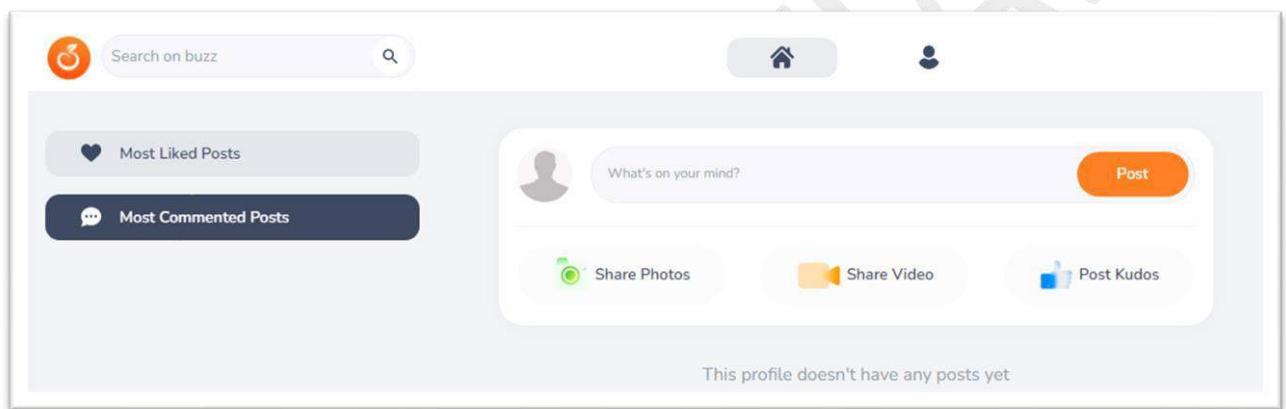
Employee\_Management page contains the entities (Employee\_List, Director, Buzz, Announcement, Organization\_Chart, Competeness, Qualification, manage\_Data, Configuration, Notification, Dashbord, Discipline)



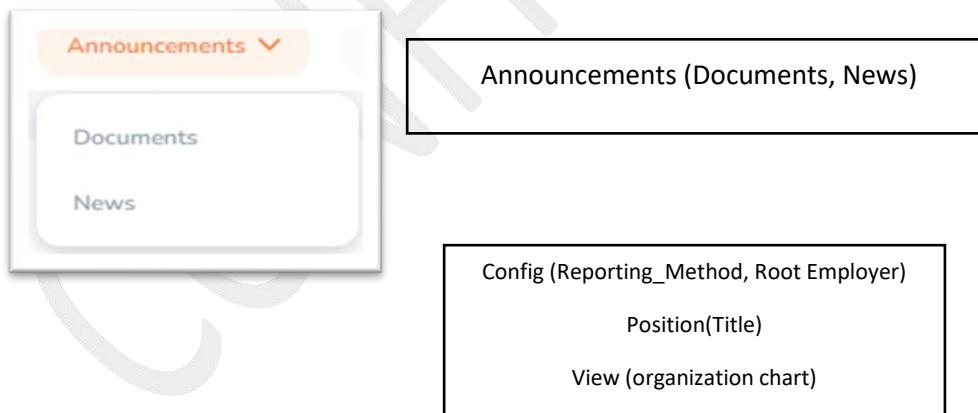
Employee\_List(Employee\_Name, Employee\_ID, Employee\_Status, Supervisor\_Name, Job\_Title, Sub\_Unit, Include, Location)



Directory(Job-Title, Location, Sub\_Unit)



Buzz(Most\_Liked\_Posts, Most\_Commented\_Posts, Post)



The screenshot displays several windows from a software application:

- Organization Chart**: A window titled "Organization Chart" with tabs for "Config" (selected), "Define Position", and "View".
- Configure Reporting Method**: A window titled "Configure Reporting Method" showing a "Reporting Method" dropdown with options like "select", "Employee", and "Top 10 list".
- Reporting Method**: A window titled "Reporting Method" showing a "Reporting Method" dropdown with options like "select", "Employee", and "Top 10 list".
- Competency List**: A window titled "Competency List" showing a table with columns "Name" and "Description". One row is visible: "Miscellaneous" under "Name" and "Miscellaneous Competency Group" under "Description".

Competency\_List (Miscellaneous, Description)

The screenshot shows a dashboard with the following sections:

- My Actions**: Shows a clipboard icon and the message "No Pending Actions to Perform".
- Employees on Leave Today**: Shows a clipboard icon and the message "Leave Period Not Defined".
- Quick Access**: Shows a clipboard icon and the message "No Shortcuts Added".
- Latest Documents**: Shows a clipboard icon and the message "No Documents Published".
- Latest News**: Shows a clipboard icon and the message "No News Published".
- Time At Work**: Shows a monitor icon and the message "No Permissions to View the Content".
- Performance Quick Feedback**: Shows a clipboard icon.

Dashboard (My\_Actions, Employe\_On\_Leave, Quick\_Access, Latest\_Documents, Latest\_News, Time\_At\_Work, Performance\_Feedback)

The screenshot shows the 'Employee Management / Qualifications' interface. At the top, there are tabs for Skills, Education, Licenses, Languages, and Memberships. Below the tabs is a table with columns for Skill, Description, and other details. The table header includes 'Skill ↑' and 'Description'.

Qualification (Skills, Education, Licenses, Languages, MemberShip)

Skills(Skill, Description)

The screenshot shows the 'Employee Management / Configurations' interface. At the top, there are tabs for Optional Fields, Custom Field Sections, Reporting Methods, Add Employee Wizard, Termination Reasons, and more. Below the tabs is a section for 'Show Deprecated Fields' with several checkboxes.

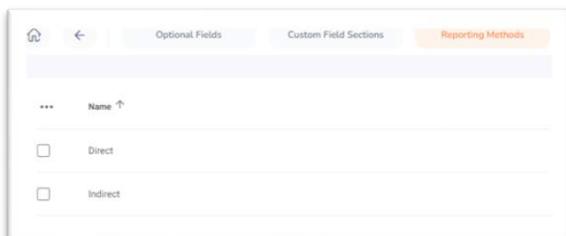
Configuration tab contains more entities (Optional\_Fields, custom\_Field\_Sections, Reporting\_Methods, Add\_Employee\_Wizard, Termination\_Reason, Directory\_Configuration, Document\_Templates)

The screenshot shows the 'Employee Management / Configurations' interface under the 'Optional Fields' tab. It displays settings for 'Show Deprecated Fields', 'Country Specific Information', and 'Dependents Related Information'. Under 'Country Specific Information', the 'Show Main Id in Personal Details' checkbox is checked, with a label 'Main Id' next to it.

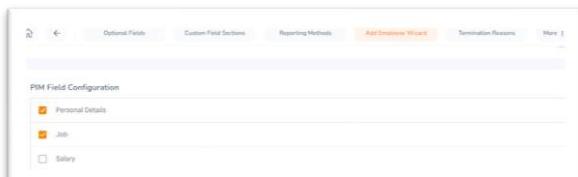
Optional\_Fields(Show\_Nick\_Name, US\_Tax\_Exemption, Show\_SIN\_Fields, Show\_Main\_ID, Show\_Nationality)



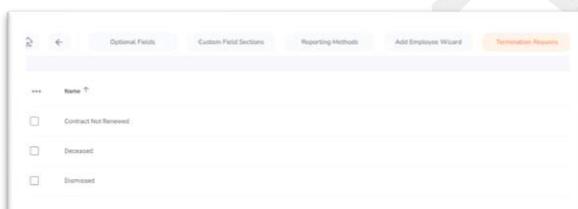
Custom\_Field\_Discription (Name,Job)



Reporting\_Method (Name)



Employee\_Wizard(Field\_Configuration, Enable)



Termination\_Reason(Name, Enable)



Director\_Configuration (Field\_Name, Enable)



Templtes\_Configuration (Templet\_Name, Description, Page\_Size, Section)

The screenshot shows a search and filter interface for managing employee data. It includes sections for selecting employees based on various criteria like job category, supervisor name, and employment status, and for selecting fields to update such as job title or location.

**Manage\_Data( Employee\_Name, Supervisor\_Name, Include, Job\_Category, Employment\_Status, Job\_Title, Sub\_Unit, Joined\_Date, Probation\_End\_Date, Date\_Of\_Permanency, Employment\_Status, Location, JobTitle, Work\_Shift)**

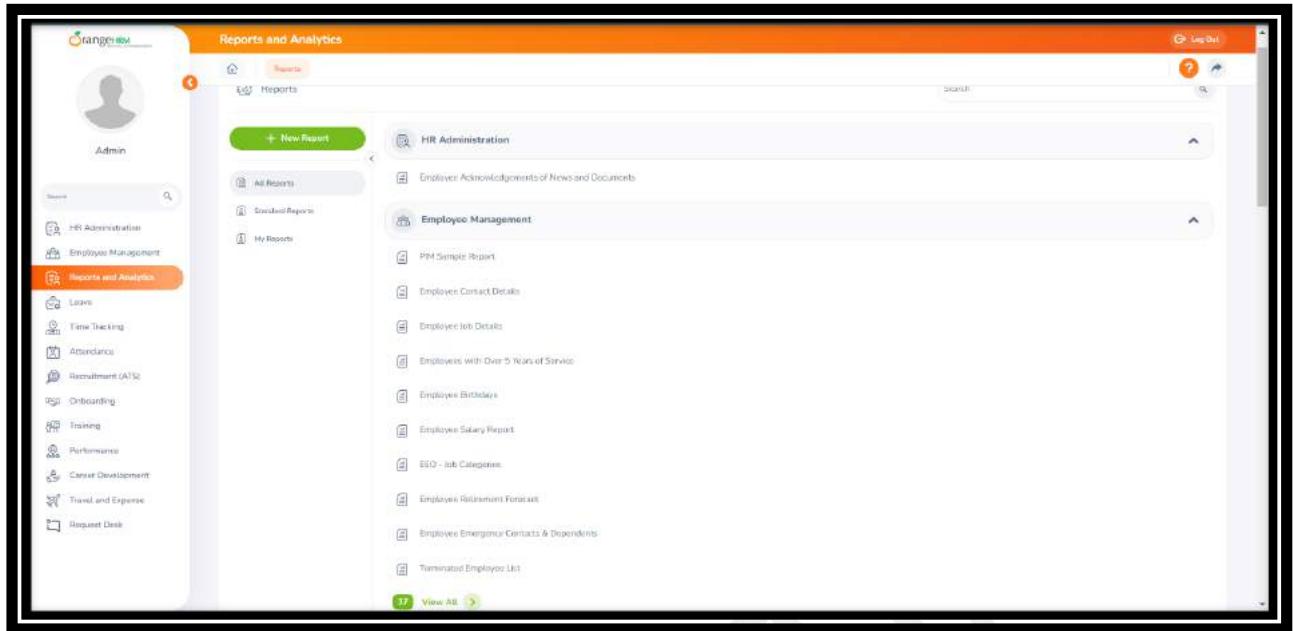
The screenshot shows a list of disciplinary cases. The columns include Employee, Case Name, Description, Created By, Created On, Disciplinary Actions, and Status. There are 0 results shown per page.

The screenshot shows a list of document templates. The columns include Template Name, Description, Page Size, and Action. There are 0 results shown per page.

**Discipline(Disciplinary\_Case, Document\_Templates)**

**Disciplinary\_Case(Employee, Case\_Name, Description, Created\_By, Created\_On, Disciplinary\_Action, Status)**

**Document\_Templates(Template\_Name, Discription, Page\_Size, Action)**

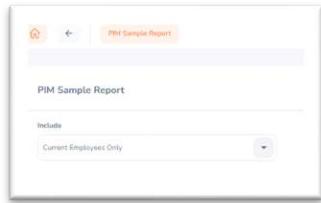


Report\_And\_Analysis contains entities (Employee\_Acknowledgments, PIM\_Sample\_Report, Employee\_Contacts\_Details, 5Year\_service\_Employee, Employee\_Birthday, Employee\_Salary\_Report, EEO\_Job\_Category, Employee\_Salary\_Report, Job\_Category, Retirement\_Forcecast, Employee\_Retirement, Employee\_emergency\_contacts, Termination\_Employee\_List)

Employee Acknowledgements of News and Documents

<b>Employee</b> Type for hints...	<b>Type</b> Document
<input type="checkbox"/> Show Acknowledgements of Past Employees	<b>User Role</b> Type for hints...
<b>Job Title</b> Type for hints...	<b>Employment Status</b> Type for hints...
<b>Location</b> Type for hints...	<b>Topic</b> Type for hints...
<b>Status</b> Published	
<b>Published Date</b> From	To
<b>Acknowledgement Status</b> Read	
<b>Acknowledgement Date</b> From	To

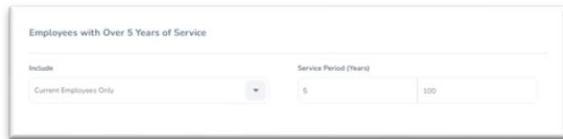
Employee\_Acknowledgments ( Employee\_Name, Supervisor\_Name, Include, Job\_Category, Employment\_Status, Job\_Title, Sub\_Unit, Joined\_Date, Probation\_End\_Date, Date\_Of\_Permanency, Employment\_Status, Location, JobTitle, Work\_Shift)



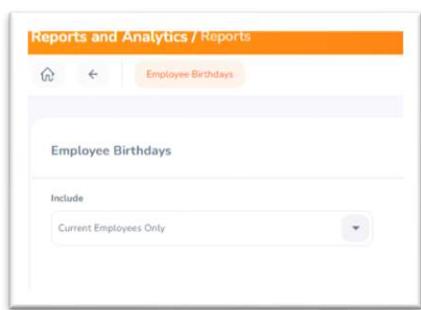
PIM\_Sample\_Report( includes)



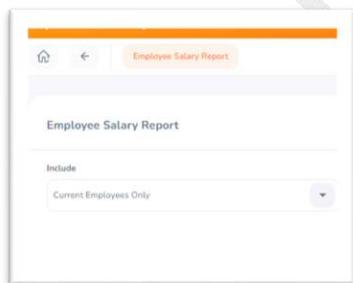
Employee\_Contact\_Details( includes)



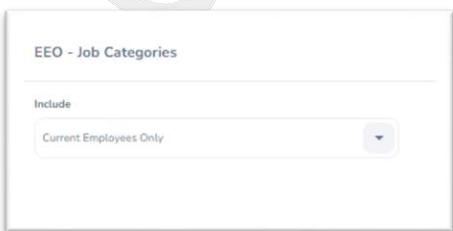
5Year\_service\_Employee( includes, time\_Period)



Employee\_Birthday( includes)



Employee\_Salary\_Report( includes)



Job\_Category( includes)

The screenshot shows a search interface titled "Employee Retirement Forecast". It includes a dropdown menu labeled "Include" set to "Current Employees Only", and two input fields for "Age (Years)" with values "54" and "100".

Forcast(includes, Age)

The screenshot shows a search interface titled "Employee Emergency Contacts & Dependents". It includes a dropdown menu labeled "Include" set to "Current Employees Only".

Contacts/Dependents(includes)

The screenshot shows a search interface titled "Terminated Employee List". It includes a dropdown menu labeled "Include" set to "Past Employees Only".

Terminated\_Employee(includes)

The screenshot shows a search interface titled "Leave Period". It has a large, mostly blank area for displaying leave period details.

Leave\_Periods(period, from, to, reasons)

The screenshot shows a search interface titled "Time Tracking". It includes a sub-section titled "Define Timesheet Start Day" with a dropdown menu set to "Tuesday". A note at the bottom indicates "\* Required".

Time\_Tracking( Start\_Day)

The screenshot shows the 'Attendance' module interface. At the top, there are tabs for 'Employee Records', 'Data Upload', and 'Configuration'. On the right, there are 'Logout' and help icons. Below the tabs, there's a section titled 'Employee Records' with fields for 'Date Range' (from 2022-07-01 to 2022-07-01), 'Employee Name', 'Sub Unit', 'Location', 'Job Title', 'Employment Status', and 'Include' (set to 'Current Employees Only'). There are also 'PDF' and 'CSV' download buttons. At the bottom left, there's a 'Data Format' dropdown set to 'HHh MMm (08h 15m)'.

Attendance includes entities( employee\_Records, Data\_Upload, Configuration)

This screenshot is identical to the one above, showing the 'Attendance' module interface with the 'Employee Records' tab selected. It displays the same search parameters: Date Range (2022-07-01 to 2022-07-01), Employee Name, Sub Unit, Location, Job Title, Employment Status, and Include (Current Employees Only). It also features PDF and CSV download options and a Data Format dropdown set to 'HHh MMm (08h 15m)'.

Employee\_records(Data\_Range(from, to), EmployeeName, Sub\_Unit, Location, Job\_Title, Employment\_status, includes, Data\_Format)

The screenshot shows the 'Attendance Configuration' section. It contains a list of checkboxes under the heading 'Attendance Configuration': 'Employee can change current time when punching in/out', 'Employee can edit/delete own attendance records', 'Supervisor/Head of Department can add/edit/delete attendance records of employees', 'Pay Policies enabled', and 'IP based Punch In/Out restriction enabled'.

Configuration(Change\_current\_time\_authority,  
Delete\_attendance\_Record\_authority,  
Pay\_Policy\_Enabled, Ip\_Base\_PunchIN)

The screenshot shows the 'Performance' module interface with the 'Appraisal List' tab selected. At the top, there are tabs for 'Appraisal List', 'Appraisal Cycles', 'Goals', 'Employee Trackers', 'Competency Profiles', and 'Configuration'. On the right, there are 'Logout' and help icons. The main area displays a table with columns: Employee Name, From, To, Due Date, Description, Appraisal Status, Review Progress, and Final Rating. A green '+' button is located in the bottom right corner of the table area. At the bottom, there are pagination controls for 'Rows per page: 50' and '0 - 0 of 0'.

Performance contains entities (Appraisal\_List, Appraisal\_Cycle, Goals, Employee\_Tracker, Competence\_profiles, Configuration)

Appraisal\_list(Employee\_List, From, To, Due\_Date, Description, Appraisal\_Status, Review\_Progress, Final\_Rating)

The screenshot shows a software interface for managing appraisal cycles. The top navigation bar includes links for Appraisal List, Appraisal Cycles (which is the active tab), Goals, Employee Trackers, Competency Profiles, and Configuration. Below the navigation is a search bar and a button for '+ Create Appraisal Cycle'. The main content area displays a table with the following columns: Appraisal Cycle Name, Due Date, Status, and Actions. A sidebar on the left shows a list of filters: All, Created, and a dropdown menu for 'Appraisal Cycle Name'.

Appraisal\_cycles ( Appraisal\_cycle\_Name, Due\_date, Status, Action)

The screenshot shows a software interface for managing goals. The top navigation bar includes links for Goal List (active) and Goal Library. Below the navigation is a search bar and export buttons for PDF and CSV. The main content area displays a table with columns: Goal Name, Level, Owner, Due Date, Status, and Priority. A sidebar on the left shows a list of filters: All, Pending, In Progress, Achieved, Not Achieved, and On Hold.

Goals(Goal\_List(Goal\_Name, Level, Owner, Due\_Date, Status, Priority), Goal\_Library(Goal\_Name,Level))

The screenshot shows a software interface for managing employee trackers. The top navigation bar includes links for Tracker List (active) and Manage Trackers. Below the navigation is a search bar and a 'Log Out' button. The main content area displays a table with columns: Tracker Name, Employee, Added Date, and Modified Date. A message at the top states 'No Records Found'.

Employee\_Tracker contains entities (Tracker\_List, Manage\_Trackers)

Tracker\_List( Tracker\_Name, Employee, Added\_Date, Modified\_Date)

The screenshot shows a software interface for managing performance trackers. The top navigation bar includes links for Tracker List (active) and Manage Trackers. Below the navigation is a search bar and a 'Log Out' button. The main content area displays a table with columns: Tracker Name, Employee, Added Date, and Modified Date. A message at the top states 'No Records Found'.

Manage\_Trackers( Tracker\_Name, Employee, Added\_Date, Modified\_Date)

Company\_Profile(Job\_Title,Sub\_Units)

Configuration(Evaluator, Questions, Templete)

Career\_development includes(individual\_DEV\_Plan, Configuration)

individual\_DEV\_Plan(Employee, IDP\_Name, Coach, Closed\_On, Status)

Configuration(9\_Box\_matrix\_Config, 9\_Box\_Answer)

All the above analysis is done for making a promotion panel in the HRM Management System that must contain data about the

employee progress so it can help in making bold decisions for company growth

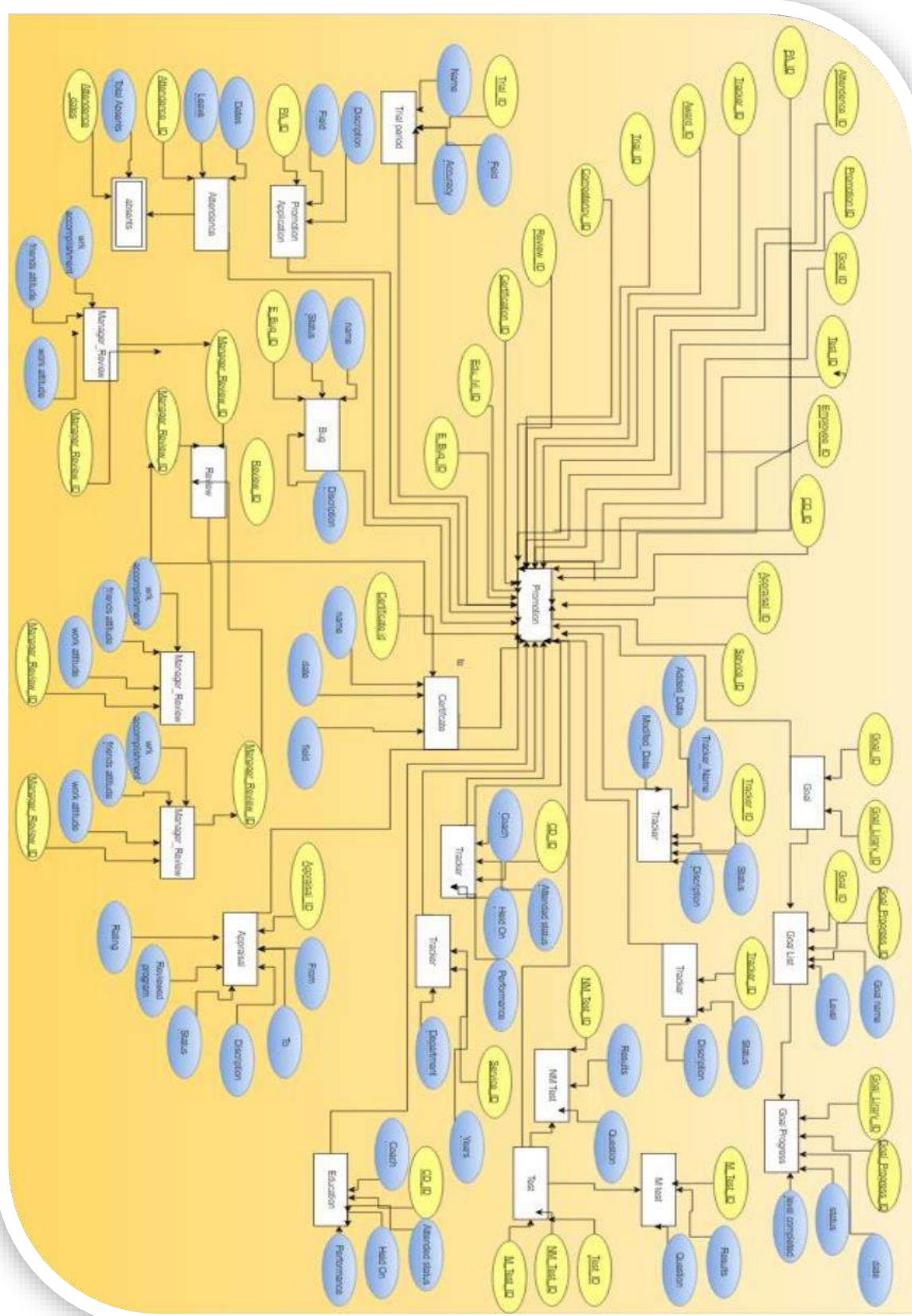
### Business rules of promotion Panel:

1. One employee can have one promoting id
2. Promoting section keep record of employee attendance of multiple days
3. Promoting id contains all the bug that occurred due to the inefficiency of employee
4. Manager, college or clients can give only one review to employee
5. Company can make the manager, colleges or clients fill multiple form on different aspects on employer
6. Employee can have multiple appraisal/interview
7. Company can take 0 or many trials on any certain work/procedure from the employee
8. employee can apply for 0 or 1 promoting application if he is eligible
9. Employee can have 0 or multiple award and certifications
10. Employee can have multiple educational levels
11. Employee can have served multiple roles in the company
12. Employee can have attended zero or multiple career development programs
13. One employee takes 0 or many tests
14. One test can contain 0 or many mandatory tests
15. One test can contain 0 or many non-mandatory tests
16. Employee can have competency single profile for recording competency whereas competency profile can contain
17. Company can put multiple trackers on the employee data to find out the performance, graphs of employee, for example if there is a sales man how would you find out the progress report of the sales man data? \*\*\* idea taken from orange HRM
18. Employee can fulfill 0 or many goals or mile stones that the company has set for HRM

## Entity Relationship Diagram

<https://drive.google.com/file/u/2/d/1YUcfc-kAXOPEI7JBsE8WDXf-c0SNq2T5/view?usp=sharing>

CONFIDENTIAL



## Conceptual to Logical Mapping

[https://drive.google.com/file/d/1U5\\_nOeWnbdsxNryQXQjAX61bqwtlkQOe/view?usp=sharing](https://drive.google.com/file/d/1U5_nOeWnbdsxNryQXQjAX61bqwtlkQOe/view?usp=sharing)

CONFIDENTIAL

## Promotion Panel

**employee  
data will be  
fetched  
through this  
route**



**Normalized Tables up to BCNF**

## Functional Dependencies before Normalization

1. **Promotion**(Promoting\_ID, Employee\_ID, Goals\_ID, Tracker\_ID, Compensation\_ID, Test\_ID, CD\_ID, Attendance\_ID, E\_BUG\_ID, Review\_ID, Award\_ID, Certification\_ID, Appraisal\_ID, Service\_ID, Edu\_LV\_ID, Trial\_ID, PA\_ID)
2. **Goal**(Goal\_ID, Goal\_ID, Goals\_Progress\_ID, Goal\_Name, Level, Due\_Date, Status)
3. **Performance\_Tracker**(Tracker\_ID, Tracker\_name, Status, Results, Date, Modified\_date)
4. **Competency\_Profile**(Tracker\_ID, area\_Of\_competency, assessment\_Discription)
5. **Test**(Test\_ID, Mandatory\_Questions, Non\_Mandatory\_Questions, Results)
6. **Career\_Development**(CD\_ID, Attended\_status, Coach, Held\_on, Performance\_Discription)
7. **Service\_info**(Service\_ID, Years\_Served, Department)
8. **Education**(Edu\_LVL\_ID, Education\_level, Date, Institution, Field)
9. **Promotion\_Application**(PA\_ID, field, description)
10. **Trial\_period**(Trial\_ID, Tril\_period, Trial\_name, Trial\_Field, Accuracy)
11. **Appraisal**(Appraisal\_ID, From, To, DueDate, Discription, Status, Review\_program, Final\_Rating)
12. **Certificates**(Certificate\_ID, Name, Date, Field)
13. **Awards**(Award\_ID, Name, Date, Due\_To)
14. **Review**(Review\_ID, Form\_ID, Manage\_Review, Colleagues\_Review, Clients\_Review, Form\_ID, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
15. **Bugs**(E\_Bug\_ID, Project\_Name, Status, Discription)
16. **Attendance**(Attendance\_ID, Total\_Absents, Absent\_date, Reason)

## Functional Dependencies after first Normalization

1. **Promotion**(Promoting\_ID, Employee\_ID, Goals\_ID, Tracker\_ID, Compensation\_ID, Test\_ID, CD\_ID, Attendance\_ID, E\_BUG\_ID, Review\_ID, Award\_ID, Certification\_ID, Appraisal\_ID, Service\_ID, Edu\_LV\_ID, Trial\_ID, PA\_ID)
2. **Goal**(Goal\_ID, Goal\_Library\_ID, Goals\_Progress\_ID)
3. **Goals\_Library**(Goal\_Library\_ID, Goal\_Name, Level)
4. **Goals\_Progress**(Goals\_Progress\_ID, Goal\_Name, Level, Due\_Date, Status)
5. **Performance\_Tracker**(Tracker\_ID, Tracker\_name, Status, Results, Date)
6. **Performance\_Tracker\_Multivalued**(Tracker\_ID, Modified\_date)
7. **Competency\_Profile**(competency\_ID, area\_Of\_competency, assessment\_Discription)
8. **Test**(Test\_ID, Mandatory\_Questions, Non\_Mandatory\_Questions, Results)
9. **Career\_Development**(CD\_ID, Attended\_status, Coach, Held\_on, Performance\_Discription)
10. **Service\_info**(Service\_ID, Years\_Served, Department)
11. **Education**(Edu\_LVL\_ID, Education\_level, Date, Institution, Field)
12. **Promotion\_Application**(PA\_ID, field, description)
13. **Trial\_period**(Tril\_period, Trial\_name, Trial\_Field, Accuracy)
14. **Appraisal**(Appraisal\_ID, From, To, DueDate, Discription, Status, Review\_program, Final\_Rating)
15. **Certificates**(Certificate\_ID, Name, Date, Field)
16. **Awards**(Award\_ID, Name, Date, Due\_To)
17. **Review**(Review\_ID, Form\_ID, Manage\_Review, Colleagues\_Review, Clients\_Review)

18. **Review\_Form**(Review\_ID, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
19. **Bugs**(E\_Bug\_ID, Project Name, Status, Discription)
20. **Attendance**(Attendance\_ID, Total\_Absents, Absent\_date, Reason)

## Functional Dependencies after Second Normalization

1. **Promotion**(Promoting\_ID, Employee\_ID, Goals\_ID, Tracker\_ID, Compensation\_ID, Test\_ID, CD\_ID, Attendance\_ID, E\_BUG\_ID, Review\_ID, Award\_ID, Certification\_ID, Appraisal\_ID, Service\_ID, Edu\_LVL\_ID, Trial\_ID, PA\_ID)
2. **Goal**(Goal\_ID, Goal\_Library\_ID, Goals\_Progress\_ID)
3. **Goals\_Library**(Goal\_Library\_ID, Goal\_Name, Level)
4. **Goals\_Progress**(Goals\_Progress\_ID, Goal\_Name, Level, Due\_Date, Status)
5. **Performance\_Tracker**(Tracker\_ID, Tracker\_name, Status, Results, Date)
6. **Performance\_Tracker\_Multivalued**(Tracker\_ID, Modified\_date)
7. **Competency\_Profile**(competency\_ID, area\_Of\_competency, assessment\_Discription)
8. **Test**(Test\_ID, Mandatory\_Questions, Non\_Mandatory\_Questions, Results)
9. **Mandatory\_Test**(Test\_ID, Mandatory\_Questions, Results)
10. **Non\_Mandatory\_Test**(Test\_ID, Non\_Mandatory\_Questions, Results)
11. **Career\_Development**(CD\_ID, Attended\_Status, Coach, Held\_on, Performance\_Discription)
12. **Service\_info**(Service\_ID, Years\_Served, Department)
13. **Education**(Edu\_LVL\_ID, Education\_level, Date, Institution, Field)
14. **Promotion\_Application**(PA\_ID, field, description)
15. **Trial\_period**(Tril\_period, Trial\_name, Trial\_Field, Accuracy)
16. **Appraisal**(Appraisal\_ID, From, To, DueDate, Discription, Status, Review\_program, Final\_Rating)
17. **Certificates**(Certificate\_ID, Name, Date, Field)
18. **Awards**(Award\_ID, Name, Date, Due\_To)
19. **Review**(Review\_ID, Form\_ID, Manage\_Review, Colleagues\_Review, Clients\_Review)
20. **Review\_Form**(Review\_ID, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
21. **Bugs**(E\_Bug\_ID, Project Name, Status, Discription)
22. **Attendance**(Attendance\_ID, Total\_Absents, Absent\_date, Reason)

## Functional Dependencies after Third Normalization

1. **Promotion**(Promoting\_ID, Employee\_ID, Goals\_ID, Tracker\_ID, Compensation\_ID, Test\_ID, CD\_ID, Attendance\_ID, E\_BUG\_ID, Review\_ID, Award\_ID, Certification\_ID, Appraisal\_ID, Service\_ID, Edu\_LVL\_ID, Trial\_ID, PA\_ID)
2. **Goal**(Goal\_ID, Goal\_Library\_ID, Goals\_Progress\_ID)
3. **Goals\_Library**(Goal\_Library\_ID, Goal\_Name, Level)
4. **Goals\_Progress**(Goals\_Progress\_ID, Goal\_Name, Level, Due\_Date, Status)
5. **Performance\_Tracker**(Tracker\_ID, Tracker\_name, Status, Results, Date)

6. **Performance\_Tracker\_Multivalued**(Tracker\_ID, Modified\_date)
7. **Competency\_Profile**(competency\_ID, area\_Of\_competency, assessment\_Discription)
8. **Test**(Test\_ID, Mandatory\_Questions, Non\_Mandatory\_Questions, Results)
9. **Mandatory\_Test**(Test\_ID, Mandatory\_Questions, Results)
10. **Non\_Mandatory\_Test**(Test\_ID, Non\_Mandatory\_Questions, Results)
11. **Career\_Development**(CD\_ID, Attended\_status, Coach, Held\_on, Performance\_Discription)
12. **Service\_info**(Service\_ID, Years\_Served, Department)
13. **Education**(Edu\_LVL\_ID, Education\_level, Date, Institution, Field)
14. **Promotion\_Application**(PA\_ID, field, description)
15. **Trial\_period**(Tril\_period, Trial\_name, Trial\_Field, Accuracy)
16. **Appraisal**(Appraisal\_ID, From, To, DueDate, Discription, Status, Review\_program, Final\_Rating)
17. **Certificates**(Certificate\_ID, Name, Date, Field)
18. **Awards**(Award\_ID, Name, Date, Due\_To)
19. **Review**(Review\_ID, Manage\_Review, Colleagues\_Review, Clients\_Review)
20. **Review\_Form\_Colleague** (Review\_ID, Colleagues\_Review, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
21. **Review\_Form\_Manager** (Review\_ID, Manage\_Review, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
22. **Review\_Form\_Client** (Review\_ID, Clients\_Review, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
23. **Bugs**(E\_Bug\_ID, Project\_Name, Status, Discription)
24. **Attendance**(Attendance\_ID, Total\_Absents, Reason)
25. **Total\_absents**(Absent\_Dates, Total\_Absents)

## Functional Dependencies after BCNF Normalization

26. **Promotion**(Promoting\_ID, Employee\_ID, Goals\_ID, Tracker\_ID, Compensation\_ID, Test\_ID, CD\_ID, Attendance\_ID, E\_BUG\_ID, Review\_ID, Award\_ID, Certification\_ID, Appraisal\_ID, Service\_ID, Edu\_LV\_ID, Trial\_ID, PA\_ID)
27. **Goal**(Goal\_ID, Goal\_Library\_ID, Goals\_Progress\_ID)
28. **Goals\_Library**(Goal\_Library\_ID, Goal\_Name, Level)
29. **Goals\_Progress**(Goals\_Progress\_ID, Goal\_Name, Level, Due\_Date, Status)
30. **Performance\_Tracker**(Tracker\_ID, Tracker\_name, Status, Results, Date)
31. **Performance\_Tracker\_Multivalued**(Tracker\_ID, Modified\_date)
32. **Competency\_Profile**(competency\_ID, area\_Of\_competency, assessment\_Discription)
33. **Test**(Test\_ID, Mandatory\_Questions, Non\_Mandatory\_Questions, Results)
34. **Mandatory\_Test**(Test\_ID, Mandatory\_Questions, Results)
35. **Non\_Mandatory\_Test**(Test\_ID, Non\_Mandatory\_Questions, Results)
36. **Career\_Development**(CD\_ID, Attended\_status, Coach, Held\_on, Performance\_Discription)
37. **Service\_info**(Service\_ID, Years\_Served, Department)
38. **Education**(Edu\_LVL\_ID, Education\_level, Date, Institution, Field)
39. **Promotion\_Application**(PA\_ID, field, description)
40. **Trial\_period**(Tril\_period, Trial\_name, Trial\_Field, Accuracy)

41. **Appraisal**(Appraisal\_ID, From, To, DueDate, Discription, Status, Review\_program, Final\_Rating)
42. **Certificates**(Certificate\_ID, Name, Date, Field)
43. **Awards**(Award\_ID, Name, Date, Due\_To)
44. **Review**(Review\_ID, Manage\_Review, Colleagues\_Review, Clients\_Review)
45. **Review\_Form\_Colleague** (Review\_ID, Colleagues\_Review, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
46. **Review\_Form\_Manager** (Review\_ID, Manage\_Review, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
47. **Review\_Form\_Client** (Review\_ID, Clients\_Review, Attitude\_Towards\_Work, Attitude\_with\_colleagues, Work\_accomplishment)
48. **Bugs**(E\_Bug\_ID, Project Name, Status, Discription)
49. **Attendance**(Attendance\_ID, Total\_Absents, Reason)
50. **Total\_absents**(Absent\_Dates, Total\_Absents)

## Code

----- Goal Progress Table -----

```
create table Goal_Progress (
    attributes
    Goal_Progress_ID int not null,
    Goal_Library_ID int not null,
    levels_Completed int null,
    DueDate date not null,
    Status Text null,
    Primary Keys
    primary key (Goal_Progress_ID)
    Foreign Keys
)
-----data insertion
insert into Goal_Progress (Goal_Progress_ID, Goal_Library_ID, levels_Completed, DueDate, Status)
values (1,1,3,'3 Dec 2022', 'Good')

*****
```

----- Goal List Table -----

```
create table Goal_List (
    attributes
```

```
Goal_Library_ID int not null,  
Goal_ID int not null,  
Goal_Name varchar(20) null,  
levels int not null,  
-----Primary Keys  
primary key (Goal_Library_ID),  
-----Foreign Keys  
Goal_Progress_ID int foreign key references Goal_Progress(Goal_Progress_ID)  
)
```

```
-----data insertion  
insert into Goal_list (Goal_Library_ID, Goal_ID, Goal_Name, levels)  
values (1,1,'Making a Website', 3)
```

```
-----Goal Table-----
```

```
create table Goal (  
-----attributes  
Goal_ID int not null,  
-----Primary Keys  
primary key (Goal_ID),  
-----Foreign Keys  
Goal_Library_ID int foreign key references Goal_List(Goal_Library_ID)  
)
```

```
-----data insertion  
insert into Goal (Goal_ID, Goal_Library_ID)  
values (1,1)
```

```
-----Performance Tracker Table-----
```

```
create table Performance_Tracker (  
-----attributes  
Tracker_ID int not null,  
Promotion_ID int not null,  
Tracker_Name varchar(30) not null,
```

```

Status varchar(20) null,
Results Text null,
Added_Date date not null,
Modified_Date Date null
-----Primary Keys
primary key (Tracker_ID)
-----Foreign Keys
)
-----data insertion
insert into Performance_Tracker (Tracker_ID, Promotion_ID, Tracker_Name, Status, Results, Added_Date, Modified_Date)
values (1,1,' User Reviews about Website','Running','Good','2 Jan 2022', '15 jan
2022')
=====
```

-----Competency Profile Table-----

```

create table Competency_Profile (
-----attributes
Competency_ID int not null,
Promotion_ID int not null,
area_of_competence varchar(20) null,
assesment_descripion Text null,
-----Primary Keys
primary key (Competency_ID)
-----Foreign Keys
)
-----data insertion
insert into Competency_Profile (Competency_ID, Promotion_ID, area_of_competence, assesment_descripion)
values (1,1,'Data Base','Lacking accuracy in Schema')
insert into Competency_Profile (Competency_ID, Promotion_ID, area_of_competence, assesment_descripion)
values (2,1,'Block Chain','Good in making Smart Contracts')
=====
```

-----Non Mandatory Test Table-----

```
create table Non_Mandatory_Test (
```

```

-----attributes
NM_Test_ID int not null,
Promotion_ID int not null,
Non_Mandatory_Question Text null,
Results Text null,
-----Primary Keys
primary key (NM_Test_ID)
-----Foreign Keys
)
-----data insertion
insert into Non_Mandatory_Test (NM_Test_ID, Promotion_ID, Non_Mandatory_Question, Results)
values (1,1,null,null)

---*****
---*****

```

-----Mandatory Test Table-----

```

create table Mandatory_Test (
-----attributes
M_Test_ID int not null,
Promotion_ID int not null,
Mandatory_Question Text null,
Results Text null,
-----Primary Keys
primary key (M_Test_ID),
-----Foreign Keys
)
-----data insertion
insert into Mandatory_Test (M_Test_ID, Promotion_ID, Mandatory_Question, Results)
values (1,1,'What Is The Difference Between ERD and Schema',
'Correct Answer')

insert into Mandatory_Test (M_Test_ID, Promotion_ID, Mandatory_Question, Results)
values (2,1,'What does Phishing Maens','Correct Answer')

---*****
---*****

```

-----Test Table-----

```

create table Test (
-----attributes

```

```

Test_ID int not null,
promotion_ID int not null,
-----Primary Keys
primary key (Test_ID),
-----Foreign Keys
NM_Test_ID int foreign key references Non_Mandatory_Test(NM_Test_ID) null,
M_Test_ID int foreign key references Mandatory_Test(M_Test_ID) null
)
-----data insertion
insert into Test (Test_ID, promotion_ID, NM_Test_ID, M_Test_ID)
values (1,1,1,1)

insert into Test (Test_ID, promotion_ID, M_Test_ID)
values (2,1,1)

*** ****
*** ****

```

## -----Career Development Table-----

```

create table Career_Development (
-----attributes
CD_ID int not null,
Promotion_ID int not null,
Attended_Status Varchar(20) not null,
Coach_Name varchar(20) not null,
Held_On date not null,
Performance_Discription Text null,
-----Primary Keys
primary key (CD_ID)
-----Foreign Keys
)
```

## -----data insertion

```

insert into Career_Development (CD_ID, Promotion_ID, Attended_Status, Coach_Name,
Held_On, Performance_Discription)
values (1,1,'Yes','Muneeb','3 Jan 2022',
'Held to Improve the Communication Skills')

=====
```

-----Service Table-----

```
create table Service (
    attributes
    Service_Id int not null,
    Promotion_ID int not null,
    Years_served int null,
    Department varchar(20) null
)
-----Primary Keys
primary key (Service_Id)
-----Foreign Keys
)
```

-----data insertion

```
insert into Service (Service_Id, Promotion_ID, Years_served, Department)
values (1,1,2,'Frontend Developer')
insert into Service (Service_Id, Promotion_ID, Years_served, Department)
values (2,1,1,'BlockChain Developer')
=====
```

-----Education Table-----

```
create table Education (
    attributes
    EDU_Lvl_ID int not null,
    Promotion_ID int not null,
    Education_level varchar(20) not null,
    date date null,
    Institute varchar(20) null,
    Field Varchar(20) not null,
)
-----Primary Keys
primary key (EDU_Lvl_ID)
-----Foreign Keys
)
```

-----data insertion

```
insert into Education (EDU_Lvl_ID, Promotion_ID, Education_level, date, Institute, Field)
values (1,1,'Graduate','3 Jan 2022','Bahria','BSCS')
insert into Education (EDU_Lvl_ID, Promotion_ID, Education_level, date, Institute, Field)
```

```
values (2,1,'Masters','3 Jan 2026','MIT','MSIT')
```

```
=====
```

-----Promotion Application Table-----

```
create table Promotion_Application (
    .....attributes
    PA_ID int not null,
    Promotion_ID int not null,
    Field varchar(20) not null,
    discription Text null,
    .....Primary Keys
    primary key (PA_ID)
    .....Foreign Keys
)
```

-----data insertion

```
insert into Promotion_Application (PA_ID, Promotion_ID, Field, discription)
values (1,1,'BlockChain','I THINK I AM ELIGIBLE FOR THIS ROLE')
```

```
=====
```

.....Trial Table.....

```
create table Trial (
    .....attributes
    Trial_ID int not null,
    Promotion_Table int not null,
    Trial_Name varchar(20) not null,
    Trial_Field varchar(20) null,
    Accuracy int null,
    .....Primary Keys
    primary key (Trial_ID)
    .....Foreign Keys
)
```

-----data insertion

```
insert into Trial (Trial_ID, Promotion_Table, Trial_Name, Trial_Field, Accuracy)
values (1,1,'Website Bug Analysis','Debugging',80)
```

=====

=====

.....Appraisal Table.....

```
create table Appraisal (
    attributes
    Appraisal_ID int not null,
    Promotion_ID int not null,
    Frrom Date not null,
    Tto Date not null,
    Discription Text null,
    Appraisal_status varchar(20) null,
    Reviewed_Program varchar(20) not null,
    Final_rating int,
    Primary Keys
    primary key (Appraisal_ID)
    Foreign Keys
)
```

-----data insertion

```
insert into Appraisal (Appraisal_ID, Promotion_ID, Frrom, Tto, Discription, Appraisal_status,
Reviewed_Program, Final_rating)
values (1,1,'3 Aug 2022','3 Sep 2022','Assesment of Debugging Skills',
'Under Progress','Web Development',5)
```

=====

=====

.....Certificate Table.....

```
create table Certificate (
    attributes
    Certificate_ID int not null,
    Promotion_ID int not null,
    Certificate_Name varchar(20) null,
    date date null,
```

```
Field varchar(20) not null,  
-----Primary Keys  
primary key (Certificate_ID)  
-----Foreign Keys  
)  
  
-----data insertion  
insert into Certificate (Certificate_ID, Promotion_ID, Certificate_Name, date, Field)  
values (1,1,'CEH', '4 June 2022', 'Cyber Security')  
insert into Certificate (Certificate_ID, Promotion_ID, Certificate_Name, date, Field)  
values (2,1,'NSE-3', '3 Feb 2022', 'Cyber Security')  
=====  
=====  
-----Award Table-----  
create table Award (  
-----attributes  
Award_ID int not null,  
Promotion_ID int not null,  
Award_Name varchar(20) not null,  
date date null,  
due_to Text null,  
-----Primary Keys  
primary key (Award_ID)  
-----Foreign Keys  
)  
  
-----data insertion  
insert into Award (Award_ID, Promotion_ID, Award_Name, date, due_to)  
values (1,1,'Best Coder', '3 April 2022','Making best Applications')  
insert into Award (Award_ID, Promotion_ID, Award_Name, date, due_to)  
values (2,1,'Oscar', '9 April 2022','Making best Youtue Video')  
=====  
=====
```

.....Manager Review Table.....

```
create table Review_Form_Manager (
    .....attributes
    Manager_Review_ID int not null,
    Review_ID int not null,
    Attitude_towards_Work Text null,
    Attitude_with_Colleague Text null,
    Work_Accomplishment Text null,
    .....Primary Keys
    primary key (Manager_Review_ID)
    .....Foreign Keys
)
```

.....data insertion

```
insert into Review_Form_Manager (Manager_Review_ID, Review_ID, Attitude_towards_Work,
    Attitude_with_Colleague, Work_Accomplishment)
    Values(1, 1,'Good', 'Helping', 'On Time')

*****
```

.....Colleague Review Table.....

```
create table Review_Form_Colleague (
    .....attributes
    Colleague_Review_ID int not null,
    Review_ID int not null,
    Attitude_towards_Work Text null,
    Attitude_with_Colleague Text null,
    Work_Accomplishment Text null,
    .....Primary Keys
    primary key (Colleague_Review_ID)
    .....Foreign Keys
)
```

.....data insertion

```
insert into Review_Form_Colleague (Colleague_Review_ID, Review_ID, Attitude_towards_Work,
    Attitude_with_Colleague, Work_Accomplishment)
    Values(1, 1,'Good', 'Helping', 'On Time')
```

```
---*****  
---*****
```

-----Client Review Table-----

```
create table Review_Form_Client (
```

```
    -----attributes
```

```
    Client_Review_ID int not null,
```

```
    Review_ID int not null,
```

```
    Attitude_towards_Work Text null,
```

```
    Attitude_with_Colleague Text null,
```

```
    Work_Accomplishment Text null,
```

```
    -----Primary Keys
```

```
    primary key (Client_Review_ID)
```

```
    -----Foreign Keys
```

```
)
```

-----data insertion

```
insert into Review_Form_Client (Client_Review_ID, Review_ID, Attitude_towards_Work,  
                                Attitude_with_Colleague,  
                                Work_Accomplishment)  
values (1,1,'Good', 'Helping', 'On Time')
```

```
---*****  
---*****
```

-----Review Table-----

```
create table Review (
```

```
    -----attributes
```

```
    Review_ID int not null,
```

```
    Promotion_ID int not null,
```

```
    -----Primary Keys
```

```
    primary key (Review_ID),
```

```
    -----Foreign Keys
```

```
    Manager_Review_ID int foreign key references Review_Form_Manager(Manager_Review_ID),  
    Colleague_Review_ID int foreign key references Review_Form_Colleague(Colleague_Review_ID),  
    Client_Review_ID int foreign key references Review_Form_Client(Client_Review_ID),  
)
```

-----data insertion

```
insert into Review (Review_ID, Promotion_ID, Manager_Review_ID, Colleague_Review_ID,
Client_Review_ID)
values (1,1,1,1)
```

=====

=====

Bug Table .....

```
create table Bug (
    attributes
E_Bug_ID int not null,
Promotion_ID int not null,
Project_Name varchar(20) null,
Status Varchar(20) null,
Discription Text null,
-----Primary Keys
primary key (E_Bug_ID),
-----Foreign Keys
)
```

-----data insertion

```
insert into Bug (E_Bug_ID, Promotion_ID, Project_Name, Status, Discription)
values (1,1,'Web Dev','Critical','Wrong Image Syntax')

insert into Bug (E_Bug_ID, Promotion_ID, Project_Name, Status, Discription)
values (2,1,'Backend','Not Critical','Logical Mistake')
```

=====

=====

Attendence Table .....

```
create table Attendence (
    attributes
Attendance_ID int not null,
Promotion_ID int not null,
Absent_Date date not null,
Leave_reason Text null,
-----Primary Keys
primary key (Attendance_ID),
-----Foreign Keys
```

)

-----data insertion

```
insert into Attendance (Attendance_ID, Promotion_ID, Absent_Date, Leave_reason)
values (1,1,'25 jan 2022','No Reason')

insert into Attendance (Attendance_ID, Promotion_ID, Absent_Date, Leave_reason)
values (2,1,'7 sep 2021','Sick Leave')
```

---



---

-----Temporary Employee Table-----

```
Create table Temporary_Employee(
Employee_ID int not null,
Employee_Name varchar(20) not null,
primary Key(Employee_ID)
)

insert into Temporary_Employee (Employee_ID, Employee_Name)
values (1,'Ali Gauhar')
```

---



---

-----Promotion Table-----

```
create table Promotion (
-----attributes
Promotion_ID int not null,
-----Primary Keys
primary key (Promotion_ID),
-----Foreign Keys
Employee_ID int foreign key references Temporary_Employee(Employee_ID) not null,
Goals_ID int foreign key references Goal(Goal_ID) not null,
Tracker_ID int foreign key references Performance_Tracker(Tracker_ID) not null,
Competency_ID int foreign key references Competency_Profile(Competency_ID) not null,
Test_ID int foreign key references Test(Test_ID) not null,
CD_ID int foreign key references Career_Development(CD_ID) not null,
Attendance_ID int foreign key references Attendance(Attendance_ID) not null,
E_Bug_ID int foreign key references Bug(E_Bug_ID) not null,
Review_ID int foreign key references Review(Review_ID) not null,
Award_ID int foreign key references Award(Award_ID) not null,
```

```

Certification_ID int foreign key references Certificate(Certificate_ID) not null,
Appraisal_ID int foreign key references Appraisal(Appraisal_ID) not null,
Service_ID int foreign key references Service(Service_ID) not null,
EDU_Lvl_ID int foreign key references Education(EDU_Lvl_ID) not null,
Trial_ID int foreign key references Trial(Trial_ID) not null,
PA_ID int foreign key references Promotion_Application(PA_ID) not null
)

```

-----data insertion

```

insert into Promotion (Promotion_ID,Employee_ID, Goals_ID, Tracker_ID, Competency_ID, Test_ID,
CD_ID, Attendance_ID, E_Bug_ID, Review_ID, Award_ID, Certification_ID,
Appraisal_ID, Service_ID, EDU_Lvl_ID, Trial_ID, PA_ID)
values (1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)
=====
```

## Sample DDL/DML/Triggers/Stored Proc/Views/Stored Functions

### Function:

```

---*****Functions and views*****---
create function Find_Total_Absents(@Employee_ID int)
returns table
as
return(select count(*) as Total_absents from Attendance, promotion, Temporary_Employee where
Temporary_Employee.Employee_ID=Promotion.Employee_ID and
Promotion.Promotion_ID=Attendance.Promotion_ID
and Temporary_Employee.Employee_ID=1)

select * from Find_Total_Absents(1)

---*****Functions and views*****---
create function Find_Absents_Of_Employee(@Employee_ID int)
returns table
as
return(select Temporary_Employee.Employee_Name, Absent_Date from Attendance, promotion,
Temporary_Employee where
Temporary_Employee.Employee_ID=Promotion.Employee_ID and
Promotion.Promotion_ID=Attendance.Promotion_ID
and Temporary_Employee.Employee_ID=@Employee_ID)

select * from Find_Absents_Of_Employee(1)

---*****Functions and views*****---
Create Function Work_Inaccuracy(@Employee_ID int)
returns table
as
return(select Temporary_Employee.Employee_Name as EmployeeName, Bug.Project_Name as in_the_project,
```

```

Bug.Status as Status,Bug.Description as Description from promotion, Temporary_Employee ,Bug
where Bug.promotion_ID=Promotion.Promotion_ID and Temporary_Employee.Employee_ID=Promotion.Employee_ID
and Temporary_Employee.Employee_ID=@Employee_ID)

select * from Work_Inaccuracy(1)
--*****Functions and views*****---

Create Function Service_Years(@Employee_ID int)
returns table
as
return(select Temporary_Employee.Employee_Name as EmployeeName, service.Years_served as years_served,
service.Department from promotion, Temporary_Employee ,Service
where service.promotion_ID=Promotion.Promotion_ID and
Temporary_Employee.Employee_ID=Promotion.Employee_ID
and Temporary_Employee.Employee_ID=@Employee_ID)

select * from Service_Years(1)
--*****Functions and views*****---

Create Function Performance_Tracking(@Employee_ID int)
returns table
as
return(select Temporary_Employee.Employee_Name as EmployeeName, Performance_Tracker.Tracker_Name as
Tracker,
Performance_Tracker.Status as status, Performance_Tracker.Results as results,
Performance_Tracker.Added_Date as date_Added,
Performance_Tracker.Modified_Date as date_Modified from promotion, Temporary_Employee
,Performance_Tracker
where Performance_Tracker.promotion_ID=Promotion.Promotion_ID and
Temporary_Employee.Employee_ID=Promotion.Employee_ID
and Temporary_Employee.Employee_ID=@Employee_ID)

select * from Performance_Tracking(1)
--*****Functions and views*****---

Create Function Skills_to_be_Considered(@area_of_competence varchar(20))
returns table
as
return(select Temporary_Employee.Employee_Name as EmployeeName, Competency_Profile.area_of_competence,
Competency_Profile.assessment_description
from promotion, Temporary_Employee ,Competency_Profile
where Competency_Profile.promotion_ID=Promotion.Promotion_ID and
Temporary_Employee.Employee_ID=Promotion.Employee_ID
and Competency_Profile.area_of_competence=@area_of_competence)

```

## Procedures

```

--*****Procedures*****---

create Procedure Attitude_Towards_Work
@Employee_ID int
as begin
select Temporary_Employee.Employee_Name as Employee_Name,
Review_Form_Manager.Attitude_towards_Work as Manager_Review,
Review_Form_Client.Attitude_towards_Work as Client_Review,
Review_Form_Colleague.Attitude_towards_Work as Colleague_Review
from Review_Form_Manager, Review_Form_Client,
Review_Form_Colleague, promotion, Temporary_Employee ,review
where Review.Review_ID=Review_Form_Manager.Review_ID and
Review_Form_Client.Review_ID=Review.Review_ID and
Review_Form_Colleague.Review_ID=Review.Review_ID and
Review.promotion_ID=Promotion.Promotion_ID
and Temporary_Employee.Employee_ID=@Employee_ID
end

Exec Attitude_Towards_Work 1
--*****Procedures*****---

create Procedure Attitude_With_Colleague

```

```

@Employee_ID int
as begin
select Temporary_Employee.Employee_Name as Employee_Name,
Review_Form_Manager.Attitude_With_Colleague as Manager_Review,
Review_Form_Client.Attitude_With_Colleague as Client_Review,
Review_Form_Colleague.Attitude_With_Colleague as Colleague_Review
from Review_Form_Manager, Review_Form_Client,
Review_Form_Colleague, promotion, Temporary_Employee ,review
where Review.Review_ID=Review_Form_Manager.Review_ID and
Review_Form_Client.Review_ID=Review.Review_ID and
Review_Form_Colleague.Review_ID=Review.Review_ID and
Review.promotion_ID=Promotion.Promotion_ID
and Temporary_Employee.Employee_ID=@Employee_ID
end

Exec Attitude_With_Colleague 1

---*****Procedures*****---

create Procedure Work_Accomplishment
@Employee_ID int
as begin
select Temporary_Employee.Employee_Name as Employee_Name,
Review_Form_Manager.Work_Accomplishment as Manager_Review,
Review_Form_Client.Work_Accomplishment as Client_Review,
Review_Form_Colleague.Work_Accomplishment as Colleague_Review
from Review_Form_Manager, Review_Form_Client,
Review_Form_Colleague, promotion, Temporary_Employee ,review
where Review.Review_ID=Review_Form_Manager.Review_ID and
Review_Form_Client.Review_ID=Review.Review_ID and
Review_Form_Colleague.Review_ID=Review.Review_ID and
Review.promotion_ID=Promotion.Promotion_ID
and Temporary_Employee.Employee_ID=@Employee_ID
end

Exec Work_Accomplishment 1

```

## Updated Constraints

```

alter table Goal_List ADD CONSTRAINT Upper_Goal_Name CHECK (Goal_name = upper(Goal_name))
alter table Performance_Tracker ADD CONSTRAINT Upper_Tracker_Name CHECK (Tracker_Name =
upper(Tracker_Name))
alter table Competency_Profile ADD CONSTRAINT Upper_area_of_competence CHECK (area_of_competence =
upper(area_of_competence))
alter table Career_Development ADD CONSTRAINT Upper_Coach_Name CHECK (Coach_Name = upper(Coach_Name))
alter table Education ADD CONSTRAINT Upper_Education_level CHECK (Education_level =
upper(Education_level))
alter table Service ADD CONSTRAINT Upper_years_Served CHECK (years_Served = upper(years_Served))
alter table Promotion_Application ADD CONSTRAINT Upper_Field CHECK (Field = upper(Field))
alter table Trial ADD CONSTRAINT Upper_Trial_Name CHECK (Trial_Name = upper(Trial_Name))
alter table Appraisal ADD CONSTRAINT Upper_Reviewed_Program CHECK (Reviewed_Program =
upper(Reviewed_Program))
alter table Certificate ADD CONSTRAINT Upper_Certificate_Name CHECK (Certificate_Name =
upper(Certificate_Name))
alter table Award ADD CONSTRAINT Upper_Award_Name CHECK (Award_Name = upper(Award_Name))
alter table Bug ADD CONSTRAINT Upper_Status CHECK (Status = upper(Status))
alter table Attendance ADD CONSTRAINT Upper_Absent_date CHECK (Absent_date = upper(Absent_date))

```

## Triggers

```

---*****Triggers*****---

create trigger Trigger_Promotion_Insert
on Promotion
for insert

```

```

as
begin
    declare @ID int
    select @ID=Promotion_ID from inserted
    Print ('New Reord with Promotion_id =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)
End

```

---\*\*\*\*\*Triggers\*\*\*\*\*---

```

create trigger Trigger_PerformanceTracker_Insert
on Performance_Tracker
for insert
as
begin
    declare @ID int
    select @ID=Tracker_ID from inserted
    Print ('New Tracker with Tracker_ID =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)
End

```

---\*\*\*\*\*Triggers\*\*\*\*\*---

```

create trigger Trigger_Certificate_Insert
on Certificate
for insert
as
begin
    declare @ID int
    select @ID=Certificate_ID from inserted
    Print ('New Certificate with Certificate_ID =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)
end

```

---\*\*\*\*\*Triggers\*\*\*\*\*---

```

create trigger Trigger_Award_Insert
on Award
for insert
as
begin
    declare @ID int
    select @ID=Award_ID from inserted
    Print ('New Award with Award_ID =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)
end

```

---\*\*\*\*\*Triggers\*\*\*\*\*---

```

create trigger Trigger_Review_Insert
on Review
for insert
as
begin
    declare @ID int
    select @ID=Review_ID from inserted
    Print ('New Review with Review_ID =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)

```

```
end
```

```
create trigger Trigger_Bug_Insert
on Bug
for insert
as
begin
    declare @ID int
    select @ID=E_Bug_ID from inserted
    Print ('New Review with Bug_ID =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)
end
```

```
create trigger Trigger_Attendance_Insert
on Attendance
for insert
as
begin
    declare @ID int
    select @ID=Attendance_ID from inserted
    Print ('New Attendance with Attendance_ID =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)
End
```

```
create trigger Trigger_Mandatory_Test_Insert
on Mandatory_Test
for insert
as
begin
    declare @ID int
    select @ID=M_Test_ID from inserted
    Print ('New Mandatory Test with M Test ID =' +
    cast(@ID as nvarchar(5))+
    'is added at '+ cast(GETDATE() as nvarchar(20)))
)
End
```

## Views

```
-- **** Views **** --
```

```
create view MileStones_Status as
select Temporary_Employee.Employee_Name as Employee_Name,
Goal_List.Goal_Name as All_MileStone_Of_Employee,
Goal_List.Levels as Total_Level_Of_MileStone,
Goal_Progress.levels_Completed as Levels_Completed_By_Employee,
Goal_Progress.status as Status_of_Employee_MileStone,
Goal_Progress.DueDate as DueDate_of_Employee_MileStone
from Goal, Promotion, Goal_List, Goal_Progress, Temporary_Employee
where Goal.Goal_ID=Goal_list.Goal_ID and
Goal_List.Goal_Library_ID=Goal_Progress.Goal_Library_ID and
Goal.Goal_ID=Promotion.Goals_ID
and Temporary_Employee.Employee_Name='Ali Gauhar'
```

```
Select * from MileStones_Status
```

```
-- **** Views **** --
```

```
create view Assesment_Record as
select Temporary_Employee.Employee_Name as Employee_Name,
```

```
Appraisal.Reviewed_Program as Program_Selected_For_Assessment,  
Appraisal.Final_rating as Rating_of_Assessment,  
Appraisal.Description as Description_of_Assessment  
from Appraisal, Promotion, Temporary_Employee  
where promotion.Employee_ID=Temporary_Employee.Employee_ID and  
Appraisal.Promotion_ID=Promotion.Promotion_ID  
and Temporary_Employee.Employee_ID=1
```

```
Select * from Assessment_Record
```

-- \*\*\*\* Views \*\*\*\* --

```
create view Service_Record as  
select Temporary_Employee.Employee_Name as Employee_Name,  
service.Years_served as Years_Served_in_Department,  
service.Department as Department  
from service, Promotion, Temporary_Employee  
where promotion.Employee_ID=Temporary_Employee.Employee_ID and  
service.Promotion_ID=service.Promotion_ID  
and Temporary_Employee.Employee_ID=1
```

```
Select * from Service_Record
```

-- \*\*\*\* Views \*\*\*\* --

```
create view mandatory_Questions_Asked_in_Test as  
select Temporary_Employee.Employee_Name as Employee_Name,  
Mandatory_Test.Mandatory_Question as Questions,  
Mandatory_Test.Results as Results  
from Mandatory_Test, Promotion, Temporary_Employee  
where promotion.Employee_ID=Temporary_Employee.Employee_ID and  
Mandatory_Test.Promotion_ID=Mandatory_Test.Promotion_ID and  
Temporary_Employee.Employee_ID=1
```

```
Select * from Service_Record
```

## ScreenShots

### Tables

	Goal_Progress_ID	Goal_Library_ID	levels_Completed	DueDate	Status		
1	1	1	3	2022-12-03	Good		
2	2	2	3	2022-12-03	Good		
	Goal_Library_ID	Goal_ID	Goal_Name	levels	Goal_Progress_ID		
1	1	1	Making a Website	3	NULL		
2	2	2	Making a Website	3	NULL		
	Tracker_ID	Promotion_ID	Tracker_Name	Status	Results	Added_Date	Modified_Date
1	1	1	User Reviews about Website	Running	Good	2022-01-02	2022-01-15
	Competency_ID	Promotion_ID	area_of_competence	assesment_description			
1	1	1	Data Base	Lacking accuracy in Schema			
2	2	1	Block Chain	Good in making Smart Contracts			
	NM_Test_ID	Promotion_ID	Non_Mandatory_Question	Results			
1	1	1	NULL	NULL			
	M_Test_ID	Promotion_ID	Mandatory_Question	Results			
1	1	1	What Is The Difference Between ERD and Schema	Correct Answer			
2	2	1	What does Phishing Maens	Correct Answer			
	Test_ID	promotion_ID	NM_Test_ID	M_Test_ID			
1	1	1	1	1			
2	2	1	NULL	1			
	CD_ID	Promotion_ID	Attended_status	Coach_Name	Held_On	Performance_Discription	
1	1	1	Yes	Muneeb	2022-01-03	Held to Improve the Communication Skills	
	Service_Id	Promotion_ID	Years_served	Department			
1	1	1	2	Frontend Developer			
2	2	1	1	BlockChain Developer			
	EDU_Jvl_ID	Promotion_ID	Education_level	date	Institute	Field	
1	1	1	Graduate	2022-01-03	Bahria	BSCS	
2	2	1	Masters	2026-01-03	MIT	MSIT	

	Appraisal_ID	Promotion_ID	From	Tto	Description	Appraisal_status	Reviewed_Program	Final_rating			
1	1	1	2022-08-03	2022-09-03	Assesment of Debugging Skills	Under Progress	Web Development	5			
	Certificate_ID	Promotion_ID	Certificate_Name	date	Field						
1	1	1	CEH	2022-06-04	Cyber Security						
2	2	1	NSE-3	2022-02-03	Cyber Security						
	Award_ID	Promotion_ID	Award_Name	date	due_to						
1	1	1	Best Coder	2022-04-03	Making best Applications						
2	2	1	Oscar	2022-04-09	Making best Youtue Video						
	Manager_Review_ID	Review_ID	Attitude_towards_Work	Attitude_with_Colleague	Work_Accomplishment						
1	1	1	Good	Helping	On Time						
	Colleague_Review_ID	Review_ID	Attitude_towards_Work	Attitude_with_Colleague	Work_Accomplishment						
1	1	1	Good	Helping	On Time						
	Client_Review_ID	Review_ID	Attitude_towards_Work	Attitude_with_Colleague	Work_Accomplishment						
1	1	1	Good	Helping	On Time						
	Review_ID	Promotion_ID	Manager_Review_ID	Colleague_Review_ID	Client_Review_ID						
1	1	1	1	1	1						
	E_Bug_ID	Promotion_ID	Project_Name	Status	Description						
1	1	1	Web Dev	Critical	Wrong Image Syntax						
2	2	1	Backend	Not Critical	Logical Mistake						
	Attendance_ID	Promotion_ID	Absent_Date	Leave_reason							
1	1	1	2022-01-25	No Reason							
2	2	1	2021-09-07	Sick Leave							
	Employee_ID	Employee_Name									
1	1	Ali Gauhar									
	Promotion_ID	Employee_ID	Goals_ID	Tracker_ID	Competency_ID	Test_ID	CD_ID	Attendance_ID	E_Bug_ID	Review_ID	Aw
1	1	1	1	1	1	1	1	1	1	1	1
	Award_ID	Certification_ID	Appraisal_ID	Service_ID	EDU_Lvl_ID	Trial_ID	PA_ID				
	1	1	1	1	1	1	1				

## Functions

Total_absents						
1	2					
<hr/>						
Employee_Name	Absent_Date					
1 Ali Gauhar	2022-01-25					
2 Ali Gauhar	2021-09-07					
<hr/>						
EmployeeName	in_the_project	Status	Discription			
1 Ali Gauhar	Web Dev	Critical	Wrong Image Syntax			
2 Ali Gauhar	Backend	Not Critical	Logical Mistake			
<hr/>						
EmployeeName	years_served	Department				
1 Ali Gauhar	2	Frontend Developer				
2 Ali Gauhar	1	BlockChain Developer				
<hr/>						
EmployeeName	Tracker	status	results	date_Added	date_Modified	
1 Ali Gauhar	User Reviews about Website	Running	Good	2022-01-02	2022-01-15	
<hr/>						
EmployeeName	area_of_competence	assesment_description				
1 Ali Gauhar	Data Base	Lacking accuracy in Schema				
<hr/>						
EmployeeName	area_of_competence	assesment_description				
1 Ali Gauhar	Block Chain	Good in making Smart Contracts				

## Procedures

	Employee_Name	Manager_Review	Client_Review	Colleague_Review
1	Ali Gauhar	Good	Good	Good
<hr/>				
	Employee_Name	Manager_Review	Client_Review	Colleague_Review
1	Ali Gauhar	Helping	Helping	Helping
<hr/>				
	Employee_Name	Manager_Review	Client_Review	Colleague_Review
1	Ali Gauhar	On Time	On Time	On Time

## Views

	Employee_Name	All_MileStone_Of_Employee	Total_Level_Of_MileStone	Levels_Completed_By_Employee	Status_Of_Employee_MileStone	DueDate_Of_Employee_MileStone
1	Ali Gauhar	Making a Website	3	3	Good	2022-12-03

## Analysis – Screenshots of the Hiring Panel

Accessing recruitment hrm

Candidates (vacancy\_id, candidate\_id, email, contact\_no, date\_applied\_stage)

Recruitment (ATS)

Log Out

Candidates Vacancies Configuration

\*\*\* Vacancy Candidate Email Contact Number Date Applied ↑ Stage

Rows per page 50 0 - 0 of 0 < >

Vacancies(vacancies\_id, job\_title, Hiring-manager, location, sub-unit, published\_date, status)

Recruitment (ATS)

Log Out

Candidates Vacancies Configuration

Vacancy ↑ Job Title Hiring Manager Location Sub-Unit Published Date Status

Rows per page 50 0 - 0 of 0 < >

Configuration(vacancy\_template, question\_pool, interview\_template, standard\_tedtd)

The screenshot shows the 'Recruitment (ATS)' configuration interface. In the top navigation bar, the 'Vacancies' tab is active. A dropdown menu under 'Configuration' is open, showing options: Vacancy Templates, Question Pool, Interview Templates, and Standard Tests. The main content area displays a table with columns: Location, Sub-Unit, Published Date, and Status. At the bottom of the table, there are filters for 'Rows per page' (set to 50) and a page indicator '0 - 0 of 0'.

Vacancy template (vacancy\_templates\_name,description ,added by,added on)

This screenshot shows the same 'Recruitment (ATS) / Configuration' interface as the previous one, but the 'Vacancy Templates' tab is now active. The table columns remain the same: Location, Sub-Unit, Published Date, and Status. The bottom filtering and pagination controls are also present.

Questionpool(questionpoolid)

In this screenshot, the 'Question Pool' tab is active. The table columns are: Vacancy Template Name, Description, Added By, and Added On. The bottom filtering and pagination controls are visible.

Interview templates(interview-tempalte\_name,descriptions)

Interview Template Name      Description      Added By

Unstructured	This is an interview where the interviewer can take notes. System does not provide in...	Admin
--------------	--	-------

Rows per page: 10 | 1 - 1 of 1 | < >

Standard test(aptitude test,job knowledge test,personality test,description of aptitude,description of job knowledge test, description of personality test)

Test Name      Description      Added By

<input type="checkbox"/>	Aptitude Test	Default Aptitude Test	Admin
<input type="checkbox"/>	Job Knowledge Test	Default Job Knowledge Test	Admin
<input type="checkbox"/>	Personality Test	Default Personality Test	Admin
<input type="checkbox"/>			

Rows per page: 10 | 1 - 3 of 3 | < >

Standard(testname,test outcome ,description,aptitudetest, jobknowledge test,personality test,pre\_recruitmenttest,premedicaltest,psychometricstest)

**Standard Test**

<b>Test Name *</b>	<b>Test Outcome</b>
Aptitude Test	Pass or Fail
<b>Description</b>	
Default Aptitude Test	

\*Required field

**CANCEL** **SAVE**

**Standard Test**

<b>Test Name *</b>	<b>Test Outcome</b>
Job Knowledge Test	Pass or Fail
<b>Description</b>	
Default Job Knowledge Test	

\*Required field

**CANCEL** **SAVE**

Go to **Recruitment -> Configuration -> Standard Tests**. Upon following this path, the following screen will be triggered.

TEST NAME	DESCRIPTION	ADDED BY
Aptitude Test	Test designed to determine a person's ability in a particular skill or field of knowledge.	Admin
Job Knowledge Test	Measures employee's demonstrated job relevant knowledge and essential skills	Admin
Personality Test	determines your strengths and talents.	Admin
Pre-recruitment Medical Test	Ensures that the employees are physically fit and able to do the job.	Admin
Psychometric Tests	analyze your emotional or psychological stability	Admin

Rows per page: 10 | 1 - 5 of 5 | +

On the above page,

**Step 1** - Select the candidate from the list that is under the 'Interview' stage and click on the candidate's profile, then click the icon to start the interview. Then a series of questions appear based on the template defined for the interview.

Kir Singh Vacancy for Technical Support Engineer

Tech Interview Guidelines

Job Relevance is the Key Factor

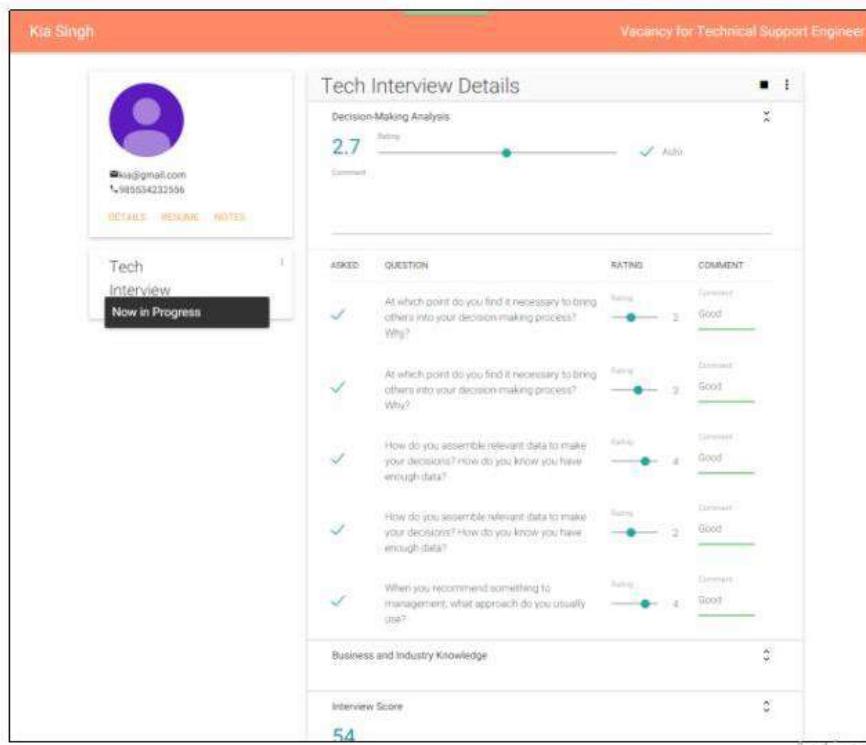
You interview questions should be designed to determine a candidate's capability to perform the essential functions you have defined for the job. Just be sure to couch your inquiries in job-relevant language, and don't make assumptions about a candidate's ability or disability. For example, let's say you are interviewing a wheelchair-bound candidate for an account manager position, and you have determined that an essential function of the job is to visit client sites. It's perfectly legal to ask how the candidate would perform the essential function: "This job will require you to be out of the office meeting with clients several days per week. Can you tell me how you would get around?" It is not OK to say to this same candidate, "How long have you been disabled?"

In other areas, where a disability is not visible, again you should confine your questions to essential job functions or workplace environment issues. For example, while you cannot ask a candidate if he or she has children or has inadequate child care, you can ask about ability to perform the job: "This job requires you to travel overnight about 2 days per week and to attend out-of-town conferences once per month. Does this travel schedule prevent a problem for you?"

Legal and illegal inquiries

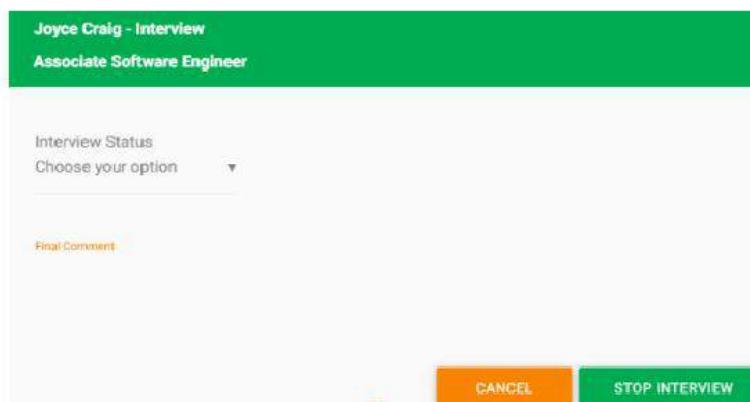
Tech Interview Now in Progress

 icon to start the interview. Then a series of questions appear based on the template defined for the interview.



ASKED	QUESTION	RATING	COMMENT
✓	At which point do you find it necessary to bring others into your decision making process? Why?	Rating: 2	Comment: Good
✓	At which point do you find it necessary to bring others into your decision making process? Why?	Rating: 3	Comment: Good
✓	How do you assemble relevant data to make your decisions? How do you know you have enough data?	Rating: 4	Comment: Good
✓	How do you assemble relevant data to make your decisions? How do you know you have enough data?	Rating: 2	Comment: Good
✓	When you recommend something to management, what approach do you usually use?	Rating: 4	Comment: Good
Business and Industry Knowledge			
Interview Score 54			

**Step 5** - Click the  icon to stop the interview. Upon stopping the interview the following screen appears.



Joyce Craig - Interview  
Associate Software Engineer

Interview Status  
Choose your option ▾

Final Comment

CANCEL STOP INTERVIEW

**Step 3** - Use the displayed questions to conduct the interview. As the candidate answers each question, rate their answer, add any desired comments, and click **Next**.

How would you describe your understanding of quantum mechanics?

Rating: 3 Note: Please explain your rating.

Question to candidate Next

**Step 4** - Once all questions within a section have been prompted, the candidate's overall score for that section appears. Change the score if needed, add any desired comments, and click **Proceed** to move to the next section.

End of Competency Section Business and Industry Knowledge

Competency Section Score: 3.2

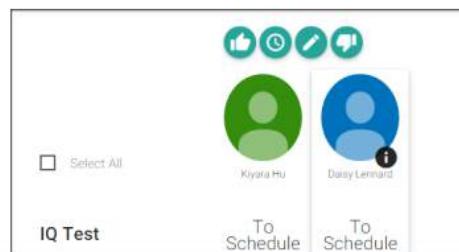
Competency Section Comment:

PROCEED

Go to **Recruitment -> Vacancies -> View Candidate**

VACANCY	JOB TITLE	HIRING MANAGER	LOCATION	SUB-UNIT	PUBLISHED DATE	STATUS	View Candidates
<input type="checkbox"/> Associate Software Engineer	Software Engineer	Charlie Carter	Canadian Development Center	Development Team	2019-03-09	PUBLISHED	
<input type="checkbox"/> Customer Success Executive	Customer Success Executive	Fiona Grace	HQ - CA, USA	Customer Success	2018-03-05	PUBLISHED	

**Step 1** - Click the "View Candidate" icon and select the candidate stage, "Application Received", Interview, Shortlisted, etc.



Activate Windows

**RECRUITMENT**

**Step 6-** After clicking 'STOP INTERVIEW' the candidate's final score appears on the bottom of the page. The image below illustrates the final score of the candidate.

The screenshot shows the 'Tech Interview Details' page for a candidate named Kia Singh. The top navigation bar includes 'Vacancy for Technical Support Engineer'. On the left, there's a sidebar with the candidate's photo, email (kia@gmail.com), phone number (9899004232356), and three buttons: 'DETAILS', 'RESUME', and 'NOTES'. Below this, it says 'Tech Interview Now In Progress'. The main area displays 'Tech Interview Details' with two sections: 'Decision-Making Analysis' (rating 2.7) and 'Business and Industry Knowledge' (rating 3.2). A 'Comment' section is present. Below these are three questions asked during the interview, each with a rating (3, 4, 3) and a 'Comment' field. At the bottom, the 'Interview Score' is listed as 60.

**Step 2** - Select the candidate for whom you wish to schedule the test, then click the icon. Then the Test Scheduling Screen appears.

The screenshot shows the 'Vacancy for Technical Support Engineer - Schedule IQ Test' page. It has a table for scheduling an 'IQ Test' for a candidate named Daisy Leonard, with fields for Date (02/28/2016) and Time (10:00). There's a 'Comment' field and a 'DONE' button at the bottom. A red circular icon with a white dot is located in the bottom right corner of the main content area.

It is possible to select multiple candidates and schedule tests for them.

The screenshot shows the 'Production Manager' page with a list of candidates: Elizabeth Scott and Sierra Miller. There are icons for selecting all candidates and for each individual candidate. Below the list, there's a 'Job Knowledge' section and a 'Activate Windows' watermark. The navigation bar includes 'OrangeHRM > HR Admin Handbook (Below 7.4.1) > Recruitment'.

## Recruitment

HR Admin Handbook (Above 7.4.1)
Employee Handbook (Above 7.4.1)
HR Admin Handbook (Below 7.4.1)
Employee Handbook (Below 7.4.1)
IT Admin Handbook
FAQs

- Add a new Vacancy
- Hiring Process
- Scheduling a Standard Test or an Interview
- Candidate Profile
- Add the Candidate as an Employee
- Add Standard Tests
- Question Pool
- Vacancy Templates
- Conducting an Interview
- Add Candidates
- Filter Candidates
- Shortlisting and Rejecting Candidates
- Interview Templates
- Vacancy Succession Report

## Grove HR Analysis

Enter name or email addresses...

Hiring Workflow + New Stage

N Naeem Rehman rehmannaeeem043@gmail.com

- Applied
- Screening
- 1st Interview
- 2nd+ Interview
- Offered
- Hired
- Rejected

Activate Windows  
Go to Settings to activate Windows 10



FEATURES PRICING ACADEMY LEARN SQL LOG IN SIGN UP

← BACK TO ARTICLES LIST

May 8, 2019 · 11 minutes read.

DESIGN PATTERNS

# Designing a Database for a Recruitment System



employees:

1. Companies contact recruitment agencies to hire on their behalf. In some cases, companies recruit employees directly.
2. The person responsible for recruitment starts the recruiting process. This process can have multiple steps, such as the initial screening, a written test, the first interview, the follow-up interview, the actual hiring decision, etc.
3. Once the recruiters have agreed on a particular process – and this can change depending on the client, the company, or the job in question – the vacancy is advertised on various platforms.
4. Applicants start applying for the job.
5. The applicants are shortlisted and invited to a test or initial interview.
6. The applicants appear for the test/interview.
7. The tests are graded by the recruiters. In some cases, tests are forwarded to specialists for grading.
8. Applicants' interviews are scored by one or more recruiters.
9. Applicants are evaluated on the basis of tests and interviews.
10. The hiring decision is made.

## A Recruitment System Database Schema

In view of the aforementioned process, our database schema is divided into five subject areas:

- **Process**
- **Jobs**
- **Application, Applicant, and Documents**
- **Test and Interviews**
- **Recruiters and Application Evaluation**

Analysis

Recruitment(office,dept,employee type,status)

The screenshot shows the 'Recruitment' section of the Grove HR software. At the top, there are navigation links: Employees, Checklists, Time Off, Attendance, Payroll, Performance, **Recruitment**, More..., and a green '+' button. Below the navigation is a search bar with dropdown filters for All Offices, All Departments, All Employment Types, and All Status. A large central icon features a briefcase and the text 'There's no opening new job. Create new one and attract talents.' A green 'New Job' button is visible. On the right, there is an 'Activate Windows' message with a link to Settings.

All offices (company name)

This screenshot shows the same 'Recruitment' interface as above, but with the 'All Offices' filter selected in the dropdown menu. The search results page displays a similar layout with the central message 'There's no opening new job. Create new one and attract talents.' and the 'New Job' button.

Department (Finance,HR,IT,management, marketing,operations,sales)

The screenshot shows the 'Recruitment' section of the Grove software. At the top, there are navigation tabs: Employees, Checklists, Time Off, Attendance, Payroll, Performance, Recruitment, and More... A green '+' button, a search icon, and a user profile icon are also present. Below the tabs, there are four filter dropdowns: All Offices, All Departments, All Employment Types, and All Status. To the right of these filters is a search bar and a grid icon. In the center, a large circular icon contains a briefcase icon. Below the icon, a message says 'There's no opening new job. Create new one and attract talents.' A green '+ New Job' button is located below the message. On the right side of the main area, there is a notification for 'Activate Windows' with a link to 'Go to Settings to activate Win'. The taskbar at the bottom shows various application icons and the system clock.

Employee types(contractor,freelancer,Fulltime,intern,part-time)

This screenshot is identical to the one above, showing the 'Recruitment' section of the Grove software. The main difference is the open dropdown menu for 'All Employment Types', which lists Contractor, Freelancer, Full-time, Intern, and Part-time. The rest of the interface, including the filters, central message, and activation notification, remains the same.

Status(draft,published,closed,internal use)

All Offices ▾ All Departments ▾ All Employment Types ▾ All Status ▾

Draft  
Published  
Closed  
Internal Use

There's no opening new job. Create new one and attract talents.

+ New Job

Activate Windows  
Go to Settings to activate Win

Create new job(job title,employment type,Department, office,Quantity,expected closingdate,description)

### Create New Job

JOB INFO > HIRING TEAM & WORKFLOW

Job Title \*  
manager

Employment Type \*  
Full-time

Department \*  
IT

Office \*  
rehman goods transport

Quantity \*  
1

Expected Closing Date  
08 Jul 2022

Description \*  
good

Activate Windows  
Go to Settings to activate Win

### Recruitment

JOBS CANDIDATES SETTINGS UPGRADE PLAN

Employment Type \*  
Select Employment Type

Department \*  
Select Department

Office \*  
Select Office

Quantity \*  
1

Expected Closing Date  
Select Date

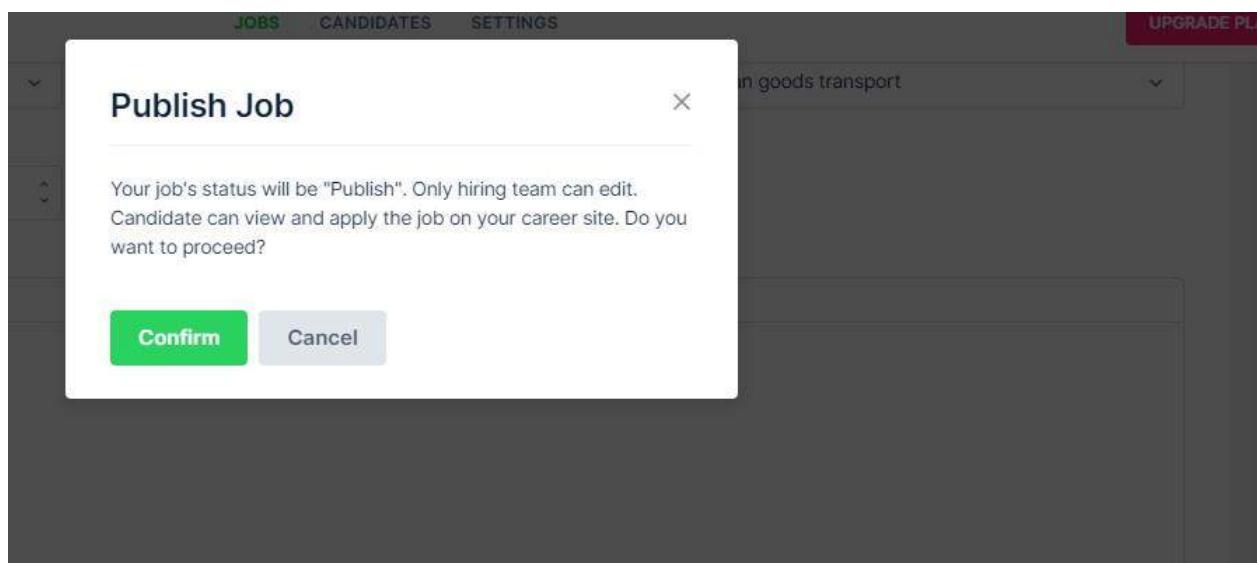
Description \*  
B I U | E |

Job Description (required)

Save & Continue ▾ Cancel

Activate Windows  
Go to Settings to activate Win

Job status(published,internal,draft)



Candidates (fullname,email,phonenumber,cc,job,created date,expected join date)

The screenshot shows a form titled "New Candidate". The form has several input fields: "CV" (with a dashed box for file upload and a green "Upload File" button), "Photo" (with a placeholder icon of a person), "First Name \*" (placeholder: "E.g. Phuong"), "Last Name \*" (placeholder: "E.g. Nguyen Thi"), "Email Address \*" (placeholder: "name@company.com"), and "Phone Number" (placeholder: "+92", with a dropdown menu for country code). There are also notes: "File format: doc, docx, pdf. Max file size: 10MB." under the CV field, and "File format: png, jpg, jpeg. Max file size: 2MB." under the Photo field. Asterisks (\*) are used to denote required fields.

The screenshot shows the 'CANDIDATES' tab selected in a navigation bar. Below the header are several search filters: 'All Jobs', 'All Stages', 'All Tags', 'All Sources', 'All Skills', and a search bar. A large central area displays columns for 'Full Name', 'Email Address', 'Phone Number', 'CV', 'Job', 'Created Date', and 'Expected Join Date'. In the center, there's a placeholder icon with a checkmark and a question mark, and a message: 'There's no candidate here. Add some to build your talent pool.' A green button labeled 'New Candidate' is visible. In the top right corner, there's a red 'UPGRADE PLAN' button and a Microsoft Windows activation notice.

All Jobs which we create(manager)

A dropdown menu is open under the 'All Jobs' filter. It lists three items: 'All Jobs' (which is highlighted with a blue background), 'manager', and 'manager Copy'.

All stage(applied,hired,offered,rejected )

The screenshot shows a user interface for managing recruitment stages and tags. At the top, there are two dropdown menus: "All Stages" and "All Tags".

The "All Stages" menu is expanded, showing the following options:

- All Stages (selected)
- Applied
- Hired
- Offered
- Rejected

The "All Tags" menu is also expanded, showing the following options:

- All Tags (selected)
- Em

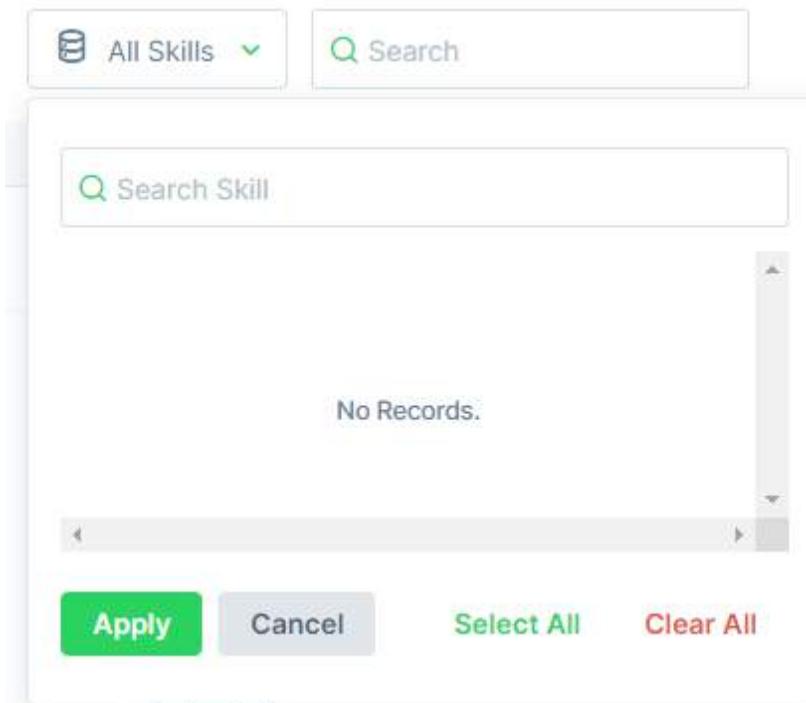
Below these filters, the text "All tags(person name)" is displayed.

On the right side of the interface, there is a section titled "Settings" which includes "Hiring Workflow", "Tags and Sources" (which is currently selected), and "Email Templates".

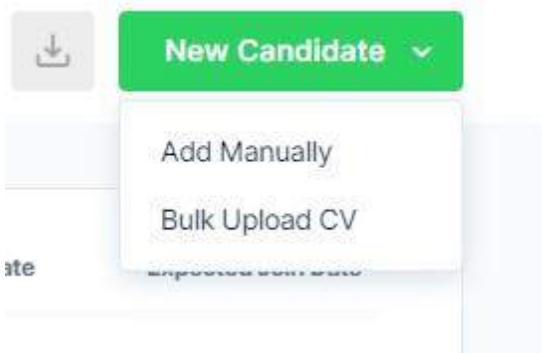
Under "Tags and Sources", there are two tabs: "TAGS" (selected) and "SOURCES". The "TAGS" tab has a button labeled "Add new tag".

The text "All sources (agency,career site,facebook,linkedin)" is displayed below the "Sources" tab.

At the bottom left, the text "Add skill options()" is visible.



New candidate (add manually,bulk upload cv)



Manually add(cv,photo,first name,last name,email address, phone number)Upload cv(resume)

File format: doc, docx, pdf. Max file size: 10MB.

**Photo**



File format: png, jpg, jpeg. Max file size: 2MB.

**First Name \***  
E.g. Phuong

**Last Name \***  
E.g. Nguyen Thi

**Email Address \***  
name@company.com

**Phone Number**  
+92

**Job**  
Select Job

**Source**  
Add Source

**Cover Letter**

Cover Letter

**Create** **Cancel**

## Setting (hiring workflow,tags and source,eamil)

**Settings**

- Hiring Workflow
- Tags and Sources
- Email Templates**

**Email Templates**

+ New

Rejection	...
Offer	...
Auto confirmation	...

Tags (tags,source)

All Tags ▾

All Sources ▾

All Skills ▾

Search

Search Tag

No Records.

Apply Cancel Select All Clear All

Source(addnew source,Twitter, agency,referral,facebook,LinkedIn,career site)

All Sources ▾

All Skills ▾

Search

Search Source

Agency

Career Site

Facebook

LinkedIn

Apply Cancel Select All Clear All

**Settings**

Hiring Workflow

**Tags and Sources**

Email Templates

**SOURCES**

- Add new source
- Twitter ... 0 Candidates
- Agency ... 0 Candidates
- LinkedIn ... 0 Candidates
- Facebook ... 0 Candidates
- Referral ... 0 Candidates
- Career Site ... 0 Candidates

### Email(rejection, offer, Auto confirmation)

**Settings**

Hiring Workflow

Tags and Sources

**Email Templates**

**Edit Email Template**

Stage \*

Rejected

Email Template \*

Rejection

Subject \*

{{job\_title}} position at {{company\_name}}.

B I U | E E E E | ⌂

Hi {{candidate\_first\_name}}.

Thank you so much for your interest in the {{job\_title}} job at {{company\_name}}.

Unfortunately, we have decided to move forward with other candidates for this position, but we would like to thank you for talking to our team and giving us the opportunity to learn about your skills and experience.

**Settings**

Hiring Workflow

Tags and Sources

**Email Templates**

**New Template**

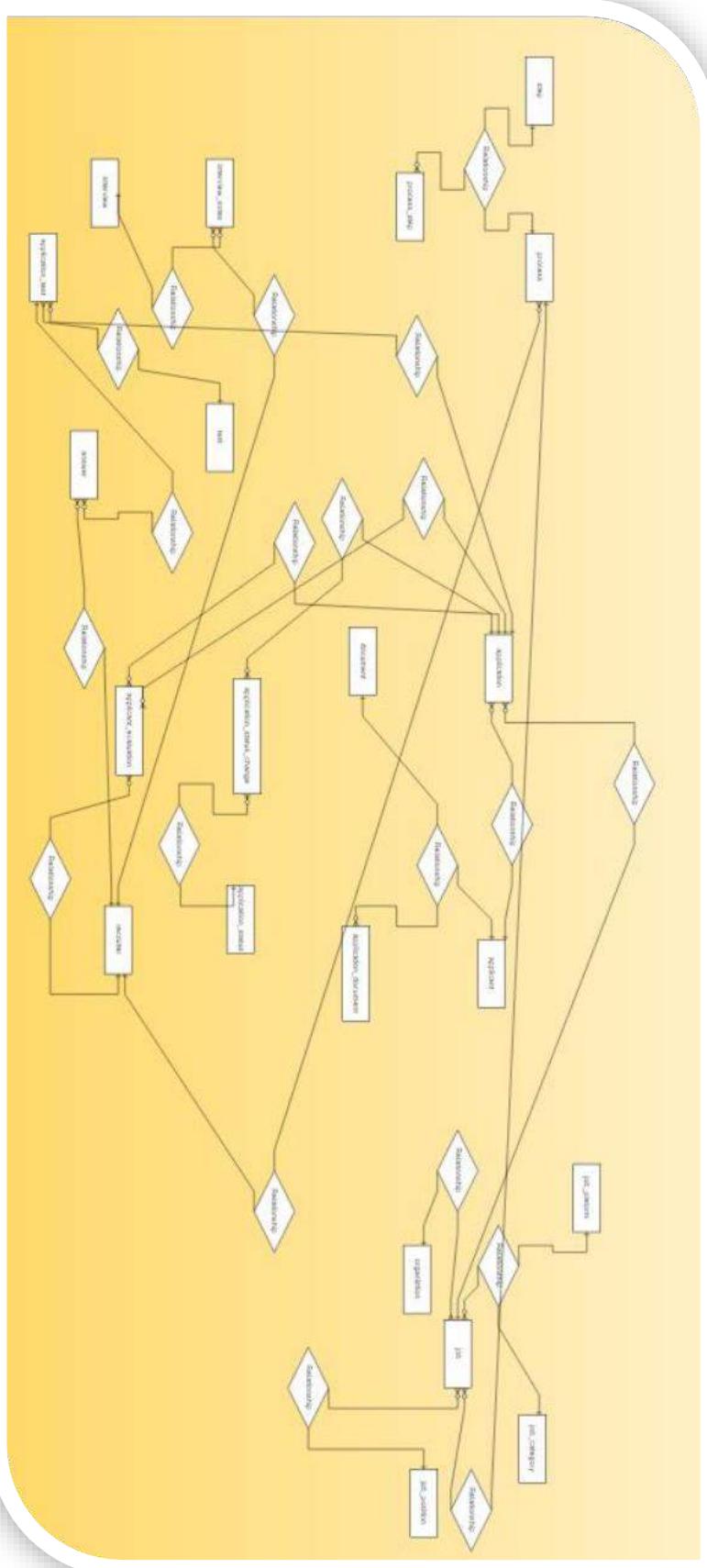
- Rejection
- Offer
- Auto confirmation

## Business rules of promotion Panel:

1. Process Process have one and many step to conducting hiring process..
2. step. Many Step belong to one relationship.
3. Job have optional and many relationship with one job\_platform.job have optional and many relationships with one job\_category.
4. Job have many to one relationship with organization.job have optional and many relationships with one job position.
5. job-category. Job category have One to optional and many relationships with job.
6. Organization's Organization's have one to many relationships with job.
7. Job position Job position have one to optional and many relationships with job.
8. Job platform have one to optional and many relationships with job .
9. Applicant have one to optional and many relationships with application.
10. Applicant will have only one email and phone which should not be multi valued.
11. Application have one to optional and many relationships with applications document.
12. Document have one to optional and many relationships with application and document.
13. Application document have optional and many relationships with application .application-document have optional and many relationships with one application.
14. Test have one to many relationships with application-test.
15. Answer have optional and many relationship with one application-test.
16. Interview have one to many optional and many relationships.
17. Interview notes have optional and many relationships with interview.
18. Application -test. have one to many and optional relationship with answer.
19. Recruiter have one to optional and many relationships with applicant\_evaluation.
20. Application\_status have one to optional and many relationships with application-status-change.
21. Application\_status-change have optional and many relationships with application\_status.
22. -Application\_evaluation have one to optional and many relationships with applicant\_evaluation.

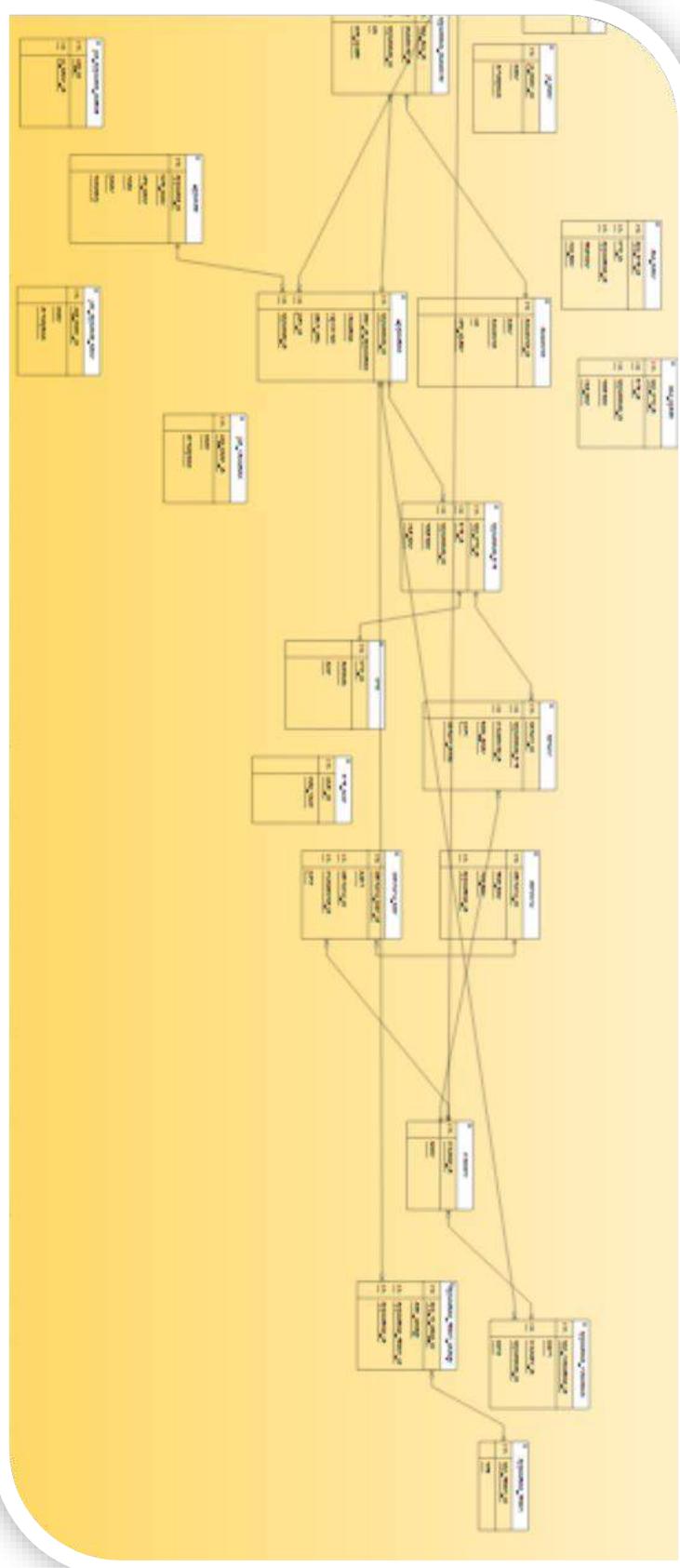
## Entity Relationship Diagram

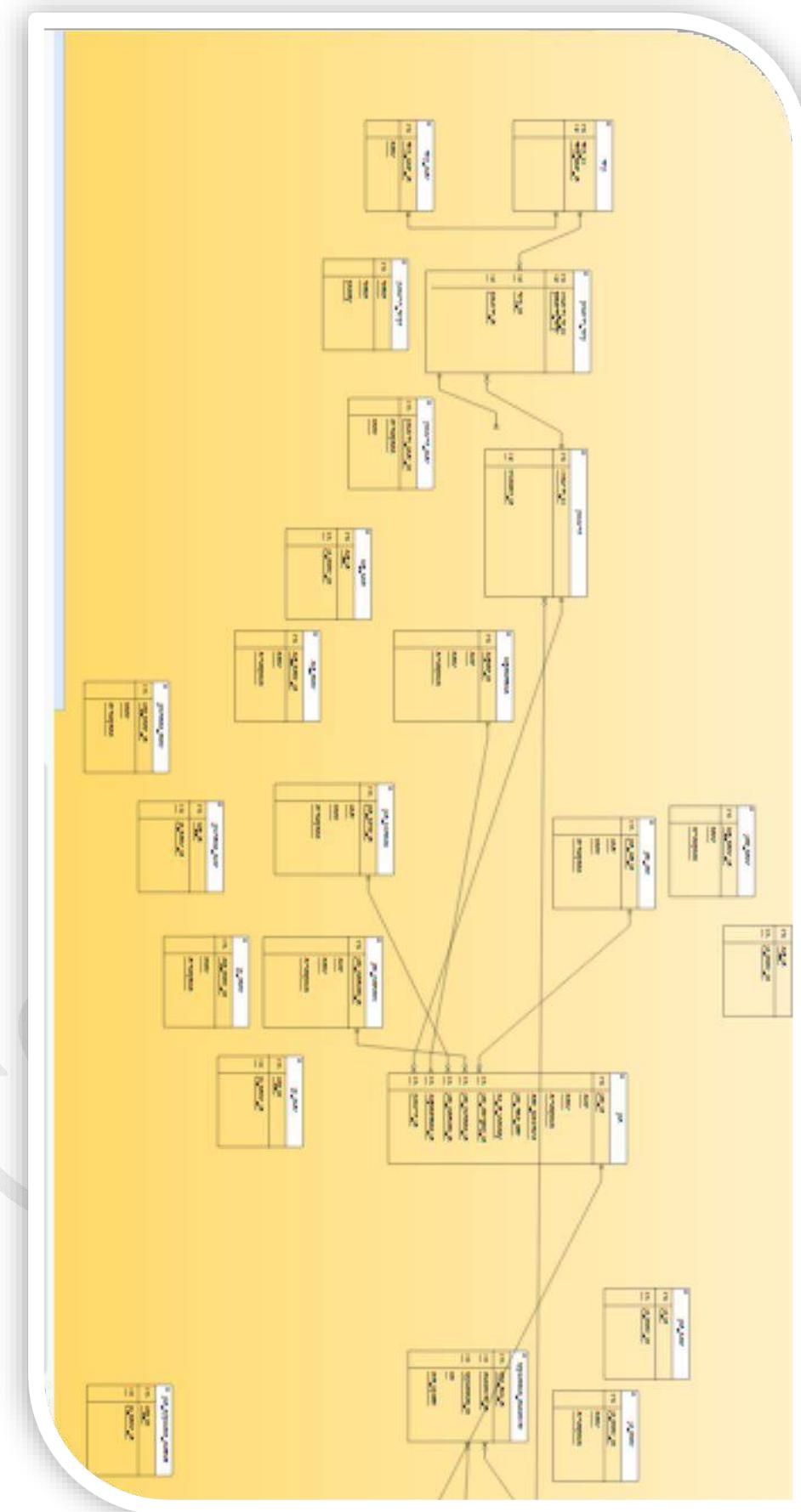
CONFIDENTIAL



## Conceptual to Logical Mapping

CONFIDENTIAL





## Normalized Tables up to BCNF

### Functional Dependencies after BCNF Normalization

Applicant (***id, phone, name, summary***)

Application(application\_id, date\_of\_application, ***jobs\_id, education\_id, applicant\_id***)

Education(***id, experience, other\_info***)

Document(***document\_id, name, url, document\_last\_update***)

Application\_document(***id, applicant\_id, contact\_id, document\_id, education\_id***)

#### Job

Job(***id, code, name, description, date\_published, job\_start\_date, no\_of\_vaccanci, job\_category\_id, job\_position\_id, job\_platform\_id, organization\_id, process\_id***)

Job\_category(***id, code, name, descriptio***)

Job\_platform(***id, code, name, descriptio***)

Organization(***org\_id, code, name, descriptio***)

Job\_position(***job\_position\_id, code, name, descriptio***)

Application\_evaluation(***id, notes, recruiter\_id, application\_id, hired***)

Recruiter(***id, name***)

Application\_status(***id, status***)

Application\_status\_change(***id, date\_changed, application\_status, application\_id***)

Test(***id, code, duration, max\_score***)

Application\_test(***id, test\_id, application\_id, start\_time, end\_time***)

Answer(***id, application test, recruiter\_id, total grade, pass, answer\_detail***)

Interview(***id, start\_time, end\_time, application\_id***)

Interview note(***id, notes, interview\_id, recruiter\_id, pass***)

Process(***id, code, descriptio, recruiter\_id***)

#### Code

```
create table Recruiter (
-----attributes
```

```
Recruiter_ID int primary key,  
First_name nvarchar(100),  
Last_name nvarchar(100),  
)  
create table process (  
-----attributes  
Process_id int primary key,  
code nvarchar(10),  
descriptio nvarchar(max),  
-----Foreign Keys  
  
Recruiter_ID int foreign key references recruiter(Recruiter_ID)  
)  
create table step (  
-----attributes  
step_ID int primary key,  
code varchar(10),  
name nvarchar(10),  
)  
create table process_step (  
-----attributes  
Process_step_ID int primary key,  
-----Foreign Keys  
Process_id int foreign key references Process (Process_id),  
step_ID int foreign key references step(step_ID)  
)  
  
create table job_category (  
-----attributes  
job_category_id int primary key,  
code nvarchar(20),  
name nvarchar(100),  
Descripti nvarchar(max),
```

```
)  
create table job_platform (  
-----attributes  
job_platform_ID int primary key,  
code varchar(20),  
name nvarchar(100),  
Descriptio nvarchar(max)  
)  
create table position (  
-----attributes  
position_ID int primary key,  
Code nvarchar(10),  
name varchar(100),  
Descriptio nvarchar(max)  
)  
create table organization (  
-----attributes  
organization_ID int primary key,  
Code nvarchar(10),  
Name nvarchar(100),  
Descriptio nvarchar(max)  
)  
create table job (  
-----attributes  
job_ID int primary key,  
Code nvarchar(10),  
Name nvarchar(100),  
Descriptio nvarchar(400),  
Date_published datetime,  
Job_start_date datetime,  
No_of_vacancy int,  
-----Foreign Keys  
job_category_id int foreign key references job_category(job_category_id ),  
position_ID int foreign key references position(position_ID ),  
organization_ID int foreign key references organization(organization_ID ) ,
```

```
job_platform_ID int foreign key references job_platform(job_platform_ID ),
Process_id int foreign key references Process(Process_id )
)

create table applicant (
-----attributes
applicant_ID int primary key,
First_name Varchar(100),
last_name Varchar(100),
Email nvarchar(100),
phone nvarchar(100),
Summary nvarchar(100),
)
create table application (
-----attributes
application_Id int primary key,
date_of_application datetime,
education nvarchar(max),
Experience nvarchar(max),
Other_info nvarchar(max),
job_ID int foreign key references job(job_ID ),
applicant_ID int foreign key references applicant(applicant_ID ),
)
create table document (
-----attributes
document_ID int primary key,
Name varchar(20),
document binary(1000),
url nvarchar(200),
Last_date_update datetime,
-----Foreign Keys
)
create table Application_document (
```

-----attributes

Application\_document\_ID int primary key,

-----Foreign Keys

document\_ID int foreign key references document(document\_ID ),

application\_Id int foreign key references application(application\_Id ),

)

create table applicant\_evaluation (

-----attributes

applicant\_evaluation\_ID int primary key,

Notes nvarchar(100),

last\_name nvarchar(100),

Recruiter\_ID int foreign key references Recruiter(Recruiter\_ID),

application\_Id int foreign key references application (application\_Id)

)

create table application\_status (

application\_status\_id int primary key,

Stat nvarchar(100)

)

create table application\_status\_change (

-----attributes

application\_status\_change int primary key,

Date\_change datetime,

Discription Text,

-----Foreign Keys

application\_status\_id int foreign key references application\_status(application\_status\_id ),

application\_Id int foreign key references application(application\_Id )

)

```
create table Test (
-----attributes
Test_id int primary key,
Code nvarchar(10),
Duration int ,
Max_score int ,
)
create table application_test (
-----attributes
application_test_id int primary key,
Start_time datetime,
end_time datetime,
-----Foreign Keys
Test_id int foreign key references Test(Test_id ),
application_Id int foreign key references application(application_Id ),
)
create table answer (
-----attributes
Answer_id int primary key,
Total_grades nvarchar(10),
Pass varchar(100),
Answer_details varchar(30),
-----Foreign Keys
application_test_id int foreign key references application_test(application_test_id ),
Recruiter_ID int foreign key references Recruiter(Recruiter_ID )
)
create table interview (
-----attributes
Interview_id int primary key,
Start_time datetime,
```

```
end_time datetime,  
  
due_to Text,  
-----Foreign Keys  
Application_id int foreign key references application(Application_id )  
)  
create table interview_notes (  
-----attributes  
interview_notes_id int primary key,  
Notes nvarchar(20),  
-----Foreign Keys  
Interview_id int foreign key references Interview(Interview_id),  
Recruiter_ID int foreign key references Recruiter(Recruiter_ID)  
)
```

## Sample DDL/DML/Triggers/Stored Proc/Views/Stored Functions

Function:

Procedures

Triggers

Views

ScreenShots

Tables

Functions

Procedures

Views

CONFIDENTIAL

# Payroll Database

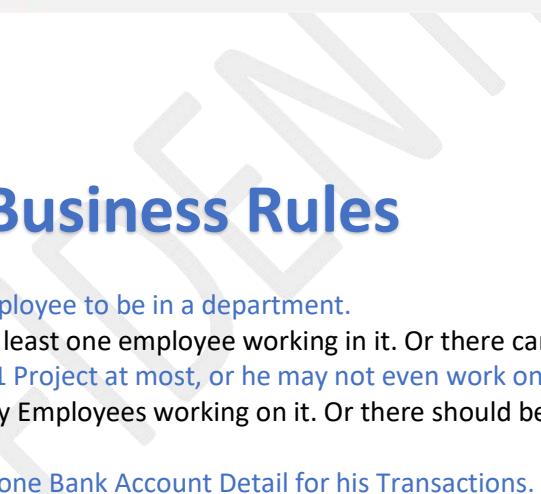
# Introduction

**Payroll** is defined as the process of paying salary to a company's employees. It starts with preparing a list of employees to be paid and ends with recording those expenses. It is a tangled process that needs different teams such as **Payroll**, **HR**, and **Finance** to work together.

# Analysis Screenshots

# Excel Template for Payroll

	A	B	C	D	E	F	G
1	Employee Name	Pay	Total Hours Worked	Overtime	Total Overtime Hours	Gross Pay	Income Tax
2	Rohan Singh	250	160	1500	10	55000	
3	Karan Grover	300	155	1000	20	66500	
4	Neha Jain	625	162	2000	30	161250	
5	Sonal Dhawan	500	140	1500	15	92500	
6	Dhruv Kohli	875	148	1200	40	177500	



A screenshot of an Excel spreadsheet titled "payroll\_monthly - Excel". The spreadsheet contains a summary of monthly payroll data for the year 2020, categorized by month (Mar 2020 to Feb 2021) and department. The columns include basic information like Employee No., Department, and various earnings and deductions. A note at the bottom of the sheet states: "On this sheet: This sheet contains a summary of all the monthly payroll data entered on the Payroll sheet. The sheet requires no user input and the data can even be filtered by department or individual employee by selecting the appropriate entries from the yellow cells at the top of the sheet." The file is shared with "Wilhelm van Noordwijk" and has a status of "Comments".

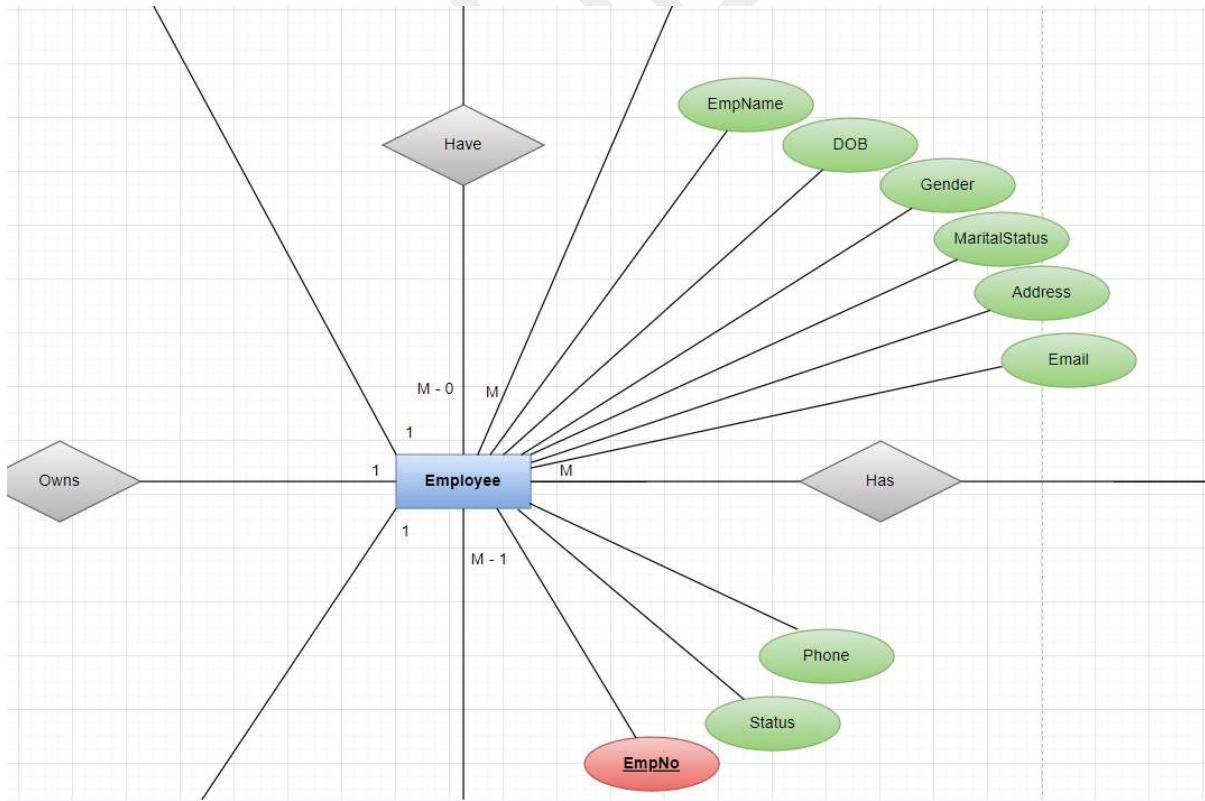
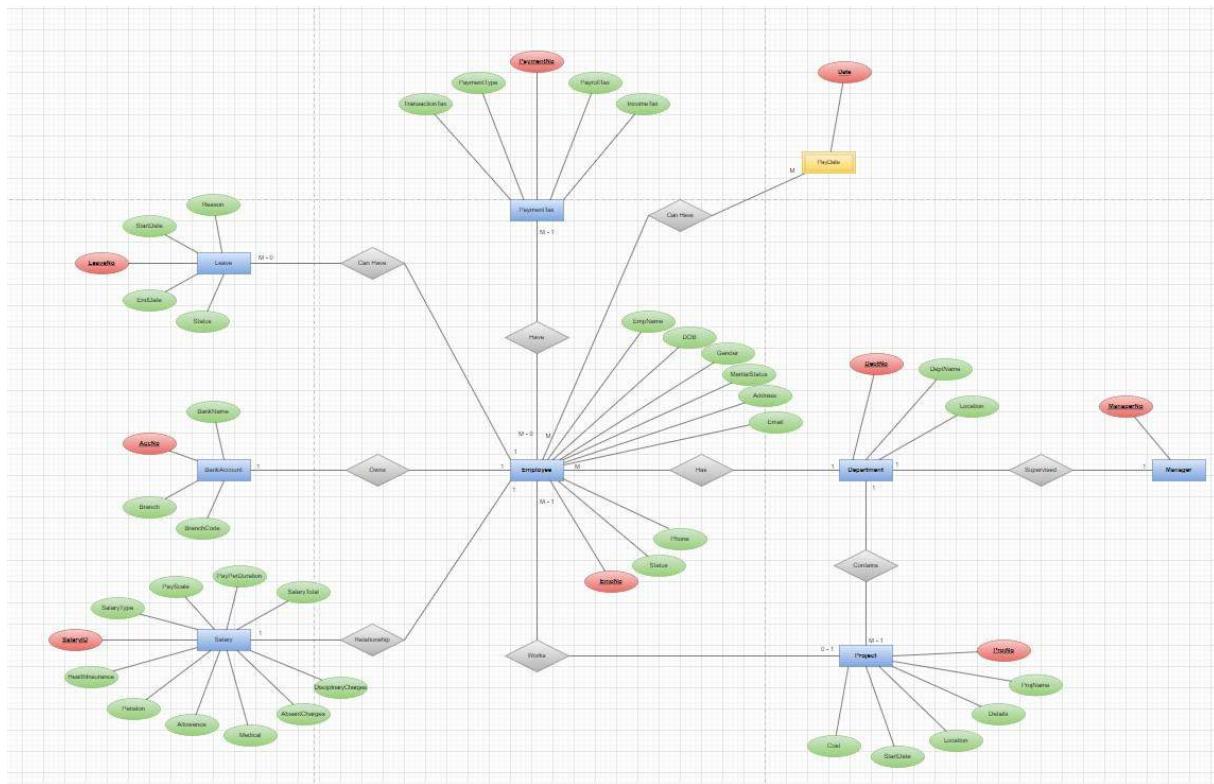
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Example (Fly) Limited			Department	Employee No.	All Employees									
2	Payroll Summary			Leave	12,600.00	324,000.00	180,230.00	571,500.00	268,650.00	5,000.00		7,460,080.00	1,649,122.70	17,583.30	912,800.70
3	Basic	Comm		Trav - Fixed			Trav - Re	Bonus - Annual	Bonus - QTR	Loan	E09	E10	Total	TAX	UF
4															PENS
5	Month	Earnings1	Earnings2	Earnings3	Earnings4	Earnings5	Earnings6	Earnings7	Earnings8	Earnings9	Earnings10	Gross Pay	Income Tax (PAYE)	Deduct11	Deduct12
6	Mar-2020	387,000.00	118,220.00	-	27,000.00	15,200.00	-	-	-	-	-	547,420.00	111,267.74	1,395.10	65,770.01
7	Apr-2020	387,000.00	111,990.00	-	27,000.00	12,100.00	-	-	-	-	-	538,090.00	109,469.74	1,395.10	65,027.51
8	May-2020	387,000.00	120,580.00	4,400.00	27,000.00	16,250.00	-	55,000.00	-	-	-	610,230.00	130,958.37	1,395.10	73,635.01
9	Jun-2020	388,500.00	115,910.00	-	27,000.00	15,300.00	-	-	-	-	-	544,710.00	111,435.42	1,395.10	64,285.01
10	Jul-2020	388,500.00	121,190.00	-	27,000.00	13,200.00	-	-	-	-	-	549,890.00	112,051.25	1,395.10	64,485.01
11	Aug-2020	388,300.00	118,370.00	-	27,000.00	13,020.00	-	64,500.00	5,000.00	-	-	616,390.00	132,325.43	1,395.10	75,848.71
12	Sep-2020	388,500.00	120,790.00	-	27,000.00	15,350.00	-	-	-	-	-	551,640.00	112,927.38	1,395.10	66,127.51
13	Oct-2020	388,500.00	107,180.00	-	27,000.00	12,100.00	-	-	-	-	-	534,780.00	107,749.68	1,395.10	65,522.51
14	Nov-2020	388,500.00	128,290.00	-	27,000.00	15,300.00	-	69,800.00	-	-	-	628,890.00	137,192.73	1,395.10	77,058.71
15	Dec-2020	388,500.00	124,850.00	8,200.00	27,000.00	14,800.00	-	571,500.00	-	-	-	1,134,850.00	324,472.03	2,220.55	148,911.21
16	Jan-2021	394,450.00	117,580.00	-	27,000.00	16,600.00	-	-	-	-	-	557,630.00	111,752.43	1,403.43	66,226.51
17	Feb-2021	394,450.00	123,730.00	-	27,000.00	19,010.00	-	79,550.00	-	-	-	643,540.00	139,120.29	1,403.43	77,903.01

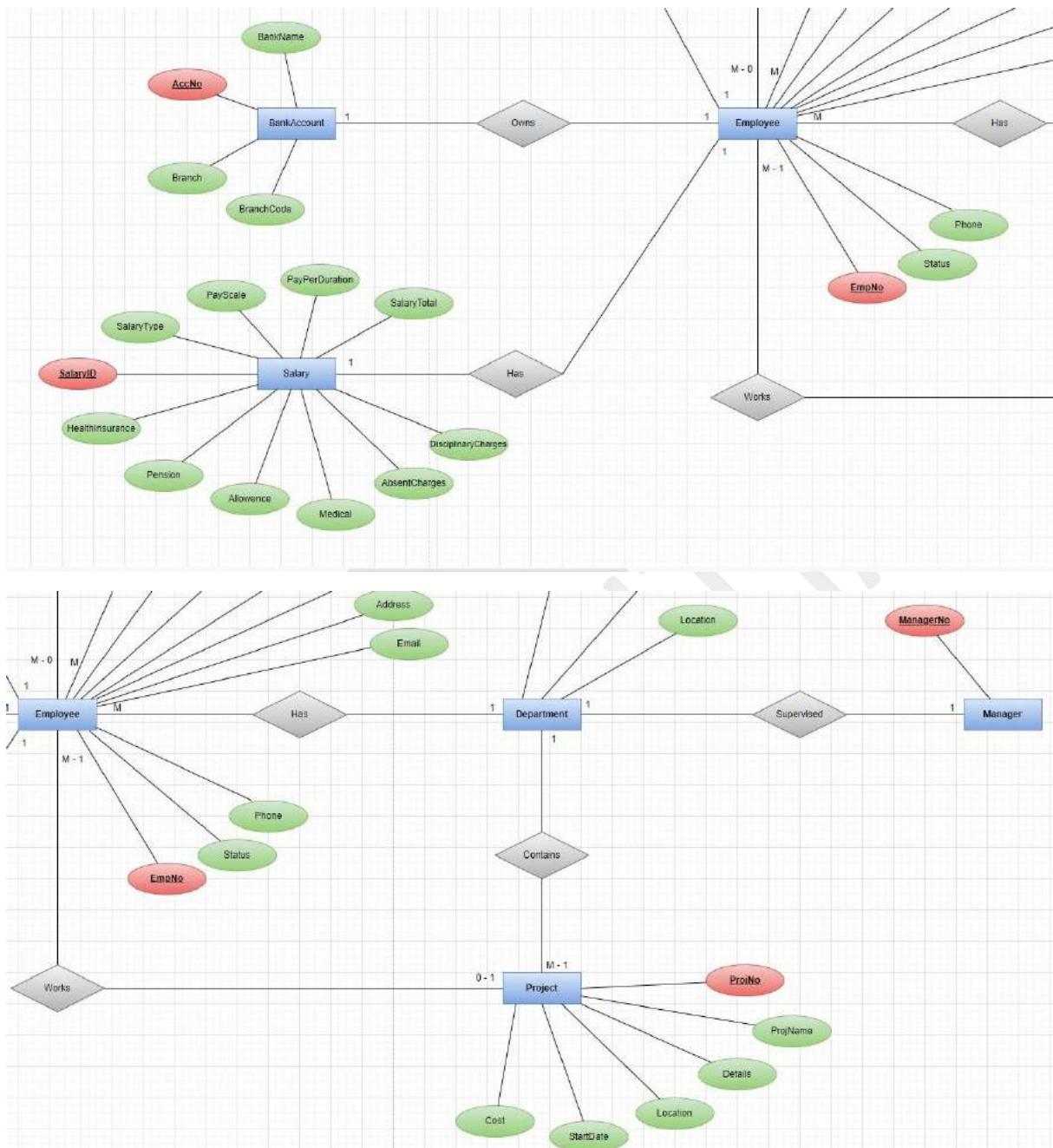
## Business Rules

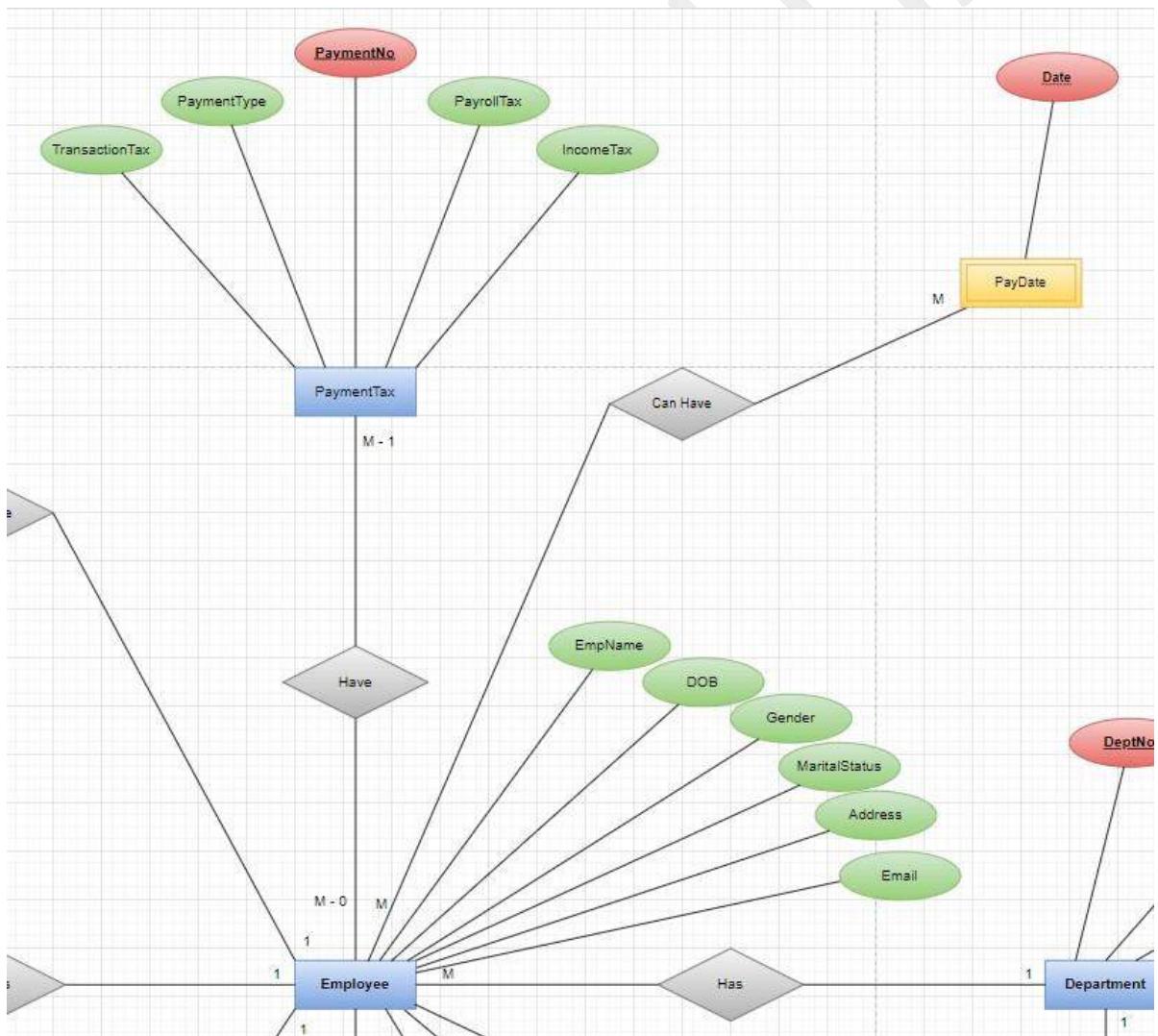
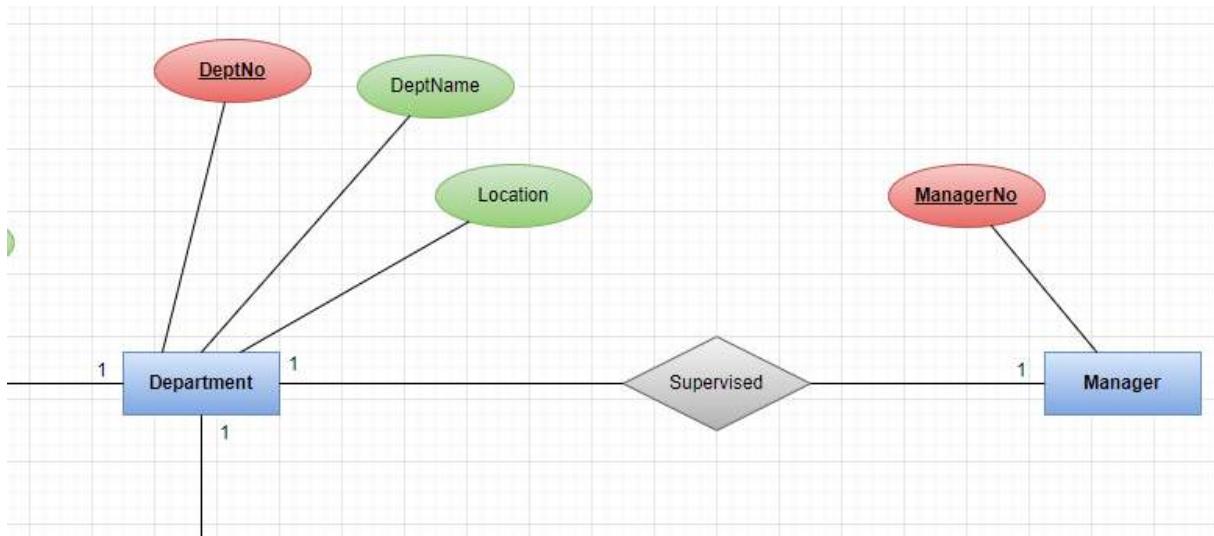
1. It is Compulsory for an Employee to be in a department.
2. A Department will have at least one employee working in it. Or there can be many more.
3. An Employee can work in 1 Project at most, or he may not even work on any.
4. A project will contain many Employees working on it. Or there should be at least 1 employee to be working on it.
5. Employee must have only one Bank Account Detail for his Transactions.
6. A single Bank Account Detail will only be owned by a single Employee.
7. Employee can have multiple Tax paying Method. It is compulsory to have at least 1 Method of Paying Tax.
8. A Tax Paying Method can be used by multiple Employees. Or there could be none to use the same payment method.
9. Employee can have Multiple Leaves during his work. It is possible he may not even go on a single leave.
10. A single Leave can only be used by one and only Employee. No other employee can have the same Leave.
11. Employee can have only one Salary. He cannot have Multiple Salaries during his work.
12. Each Salary will be owned by their respective Employee only.
13. A Department will have only one manager who will manage it.
14. Each manager will only manage only one Department. No more than that.
15. Each Department can have Multiple Projects or at least 1 project.
16. 1 Single Project can only be owned by 1 distinguished Department.
17. Salary Must Contain Bonus. Whether the bonus is 0 or Bonus can be Millions.
18. A Single Bonus will be used for a single salary
19. Salary Must also contain Charges. Whether it is empty or penalty.
20. Charges will be used for a single salary Only.

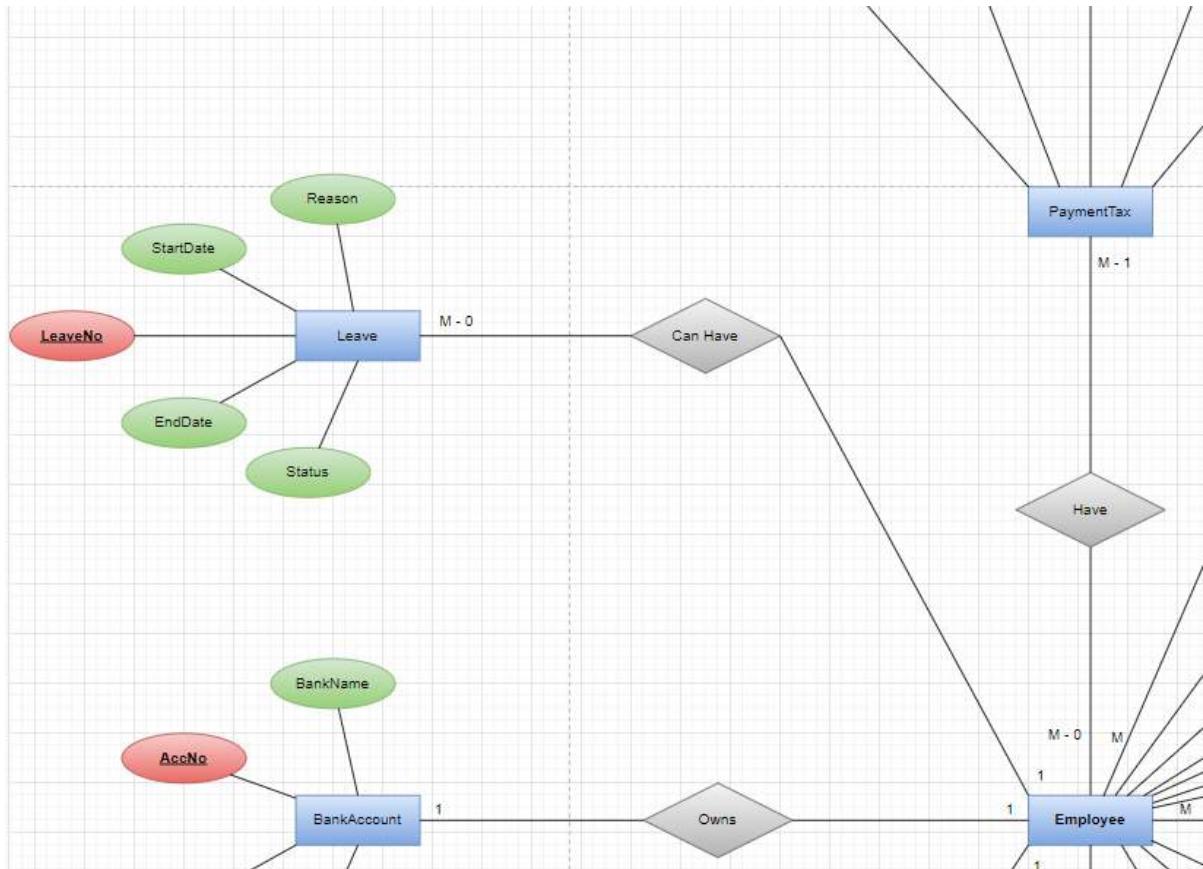
21. 1 Single Payroll will contain Detail of at least 1 employee or there could be many.
22. An Employee will only have 1 payroll at a time.
23. 1 Single Payroll will contain Detail of at least 1 department detail of an Employee or there could be many.
24. A department will have only 1 payroll at a time.
25. 1 Single Payroll will contain Detail of at least 1 working Project of a Department or there could be many.
26. 1 Single Payroll will contain Detail of at least 1 Salary of an Employee or there could be many.
27. 1 Single Payroll will contain Detail of at least 1 Leave Information of an Employee or there could be many leave info for multiple employees.
28. 1 Single Payroll will contain Detail of at least 1 Tax Payment Method of an Employee or there could be many more as much as the number of employees.
29. 1 Single Payroll will contain Detail of at least 1 Bank Details of an Employee or there could be multiple details for multiple employees.
30. A Payroll will have only 1 Paydate.
31. Paydate may contain multiple payrolls of that time or there could be none.

## Entity-Relationship Diagram Payroll









ERD Link: <https://drive.google.com/file/d/1ewI9VF-duXv-fbqbJsqWNcS6AS3XQiEK/view?usp=sharing>

Download the File and open it in <https://www.Draw.io>

## Functional Dependencies and Normalization

**Payroll** (PNo, EmpNo, EmpName, Dob, Gender, MaritalStatus, Address, Email, Phone, EmpHireDate, EmpStatus, DeptNo, DeptName, DeptLocation, ManagerNo, ManagerName, ProjNo, ProjName, ProjDetail, ProjLocation, ProjStartDate, ProjCost, AccNo, BankName, Branch, BranchCode, SalaryID, SalaryType, PayScale, PayPerDuration, SalaryTotal, ChargesID, AbsentCharges, DisciplinaryCharges, BonusCode, HealthInsurance, Pension, Allowence, Medical, PaymentNo, PaymentTax, TransactionTax, PayrollTax, IncomeTax, LeaveNo, LeaveStartDate, LeaveEndDate, Reason, LeaveStatus, HoursWorked, Date, Report, TotalAmmount)

## 1<sup>st</sup> Normalization Form

**PayrollDetail** ( PNo, EmpNo, DeptNo, ProjNo, AccNo, SalaryID, PaymentNo, LeaveNo, HoursWorked, Date, Report, TotalAmmount)

**Employee**(EmpNo, EmpName, Dob, Gender, MaritalStatus, Address, Email, Phone, EmpHireDate, EmpStatus)

**Department**(DeptNo, DeptName, DeptLocation, ManagerNo, ManagerName)

**Project**(ProjNo, ProjName, ProjDetail, ProjLocation, ProjStartDate, ProjCost)

**Bank**(AccNo, BankName, Branch, BranchCode)

**Salary**(SalaryID, SalaryType, PayScale, PayPerDuration, SalaryTotal, ChargesID , AbsentCharges, DisciplinaryCharges, BonusCode, HealthInsurance, Pension, Allowence, Medical)

**PaymentTax**(PaymentNo, PaymentTax, TransactionTax, PayrollTax, IncomeTax)

**Leave**(LeaveNo, LeaveStartDate, LeaveEndDate, Reason, LeaveStatus)

## 2<sup>nd</sup> Normalization Form

**PayrollDetail** ( PNo, EmpNo, DeptNo, ProjNo, AccNo, SalaryID, PaymentNo, LeaveNo, HoursWorked, Date, Report, TotalAmmount)

**Employee**(EmpNo, EmpName, Dob, Gender, MaritalStatus, Address, Email, Phone, EmpHireDate, EmpStatus)

**Department**(DeptNo, DeptName, DeptLocation, ManagerNo)

**Manager**(ManagerNo, ManagerName)

**Project**(ProjNo, ProjName, ProjDetail, ProjLocation, ProjStartDate, ProjCost)

**Bank**(AccNo, BankName, Branch, BranchCode)

**Salary**(SalaryID, SalaryType, PayScale, PayPerDuration, SalaryTotal, ChargesID, BonusCode)

**Charges**(ChargesID, AbsentCharges, DisciplinaryCharges)

**Bonus**(BonusCode, HealthInsurance, Pension, Allowence, Medical)

**PaymentTax**(PaymentNo, PaymentTax, TransactionTax, PayrollTax, IncomeTax)

**Leave**(LeaveNo, LeaveStartDate, LeaveEndDate, Reason, LeaveStatus)

## 3<sup>rd</sup> Normalization Form

**PayDate** ( Date, PNo)

**PayrollDetail** ( PNo, EmpNo, DeptNo, ProjNo, AccNo, SalaryID, PaymentNo, LeaveNo, HoursWorked, Report, TotalAmmount)

**Employee**(EmpNo, EmpName, Dob, Gender, MaritalStatus, Address, Email, Phone, EmpHireDate, EmpStatus, DeptNo, ProjNo)

**Department**(DeptNo, DeptName, DeptLocation, ManagerNo)

**Manager** (ManagerNo, ManagerName)

**Project**(ProjNo, ProjName, ProjDetail, ProjLocation, ProjStartDate, ProjCost, DeptNo)

**Bank**(AccNo, BankName, Branch, BranchCode, EmpNo)

**Salary**(SalaryID, SalaryType, PayScale, PayPerDuration, SalaryTotal, ChargesID, BonusCode, EmpNo)

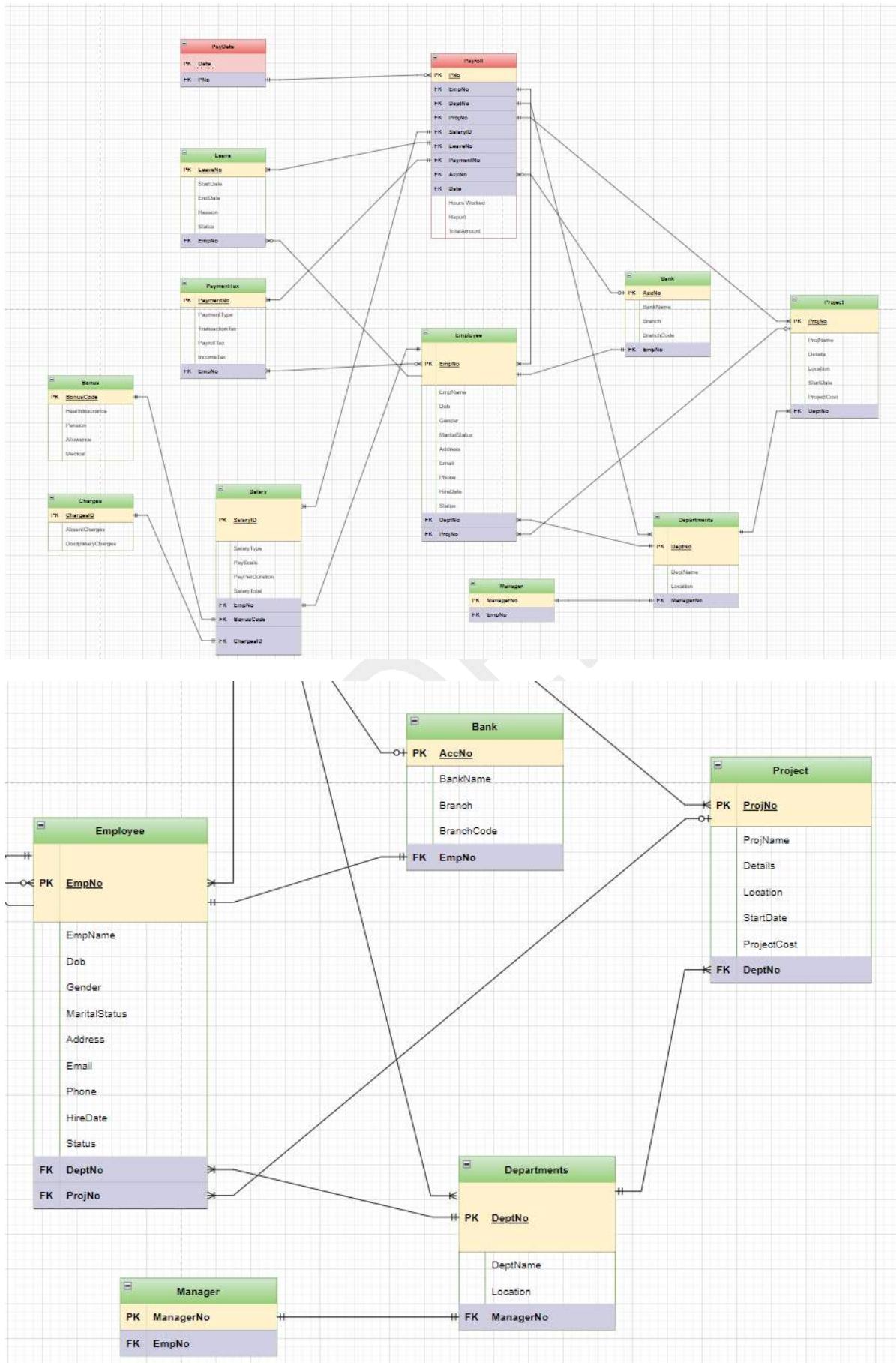
**Charges**(ChargesID, AbsentCharges, DisciplinaryCharges)

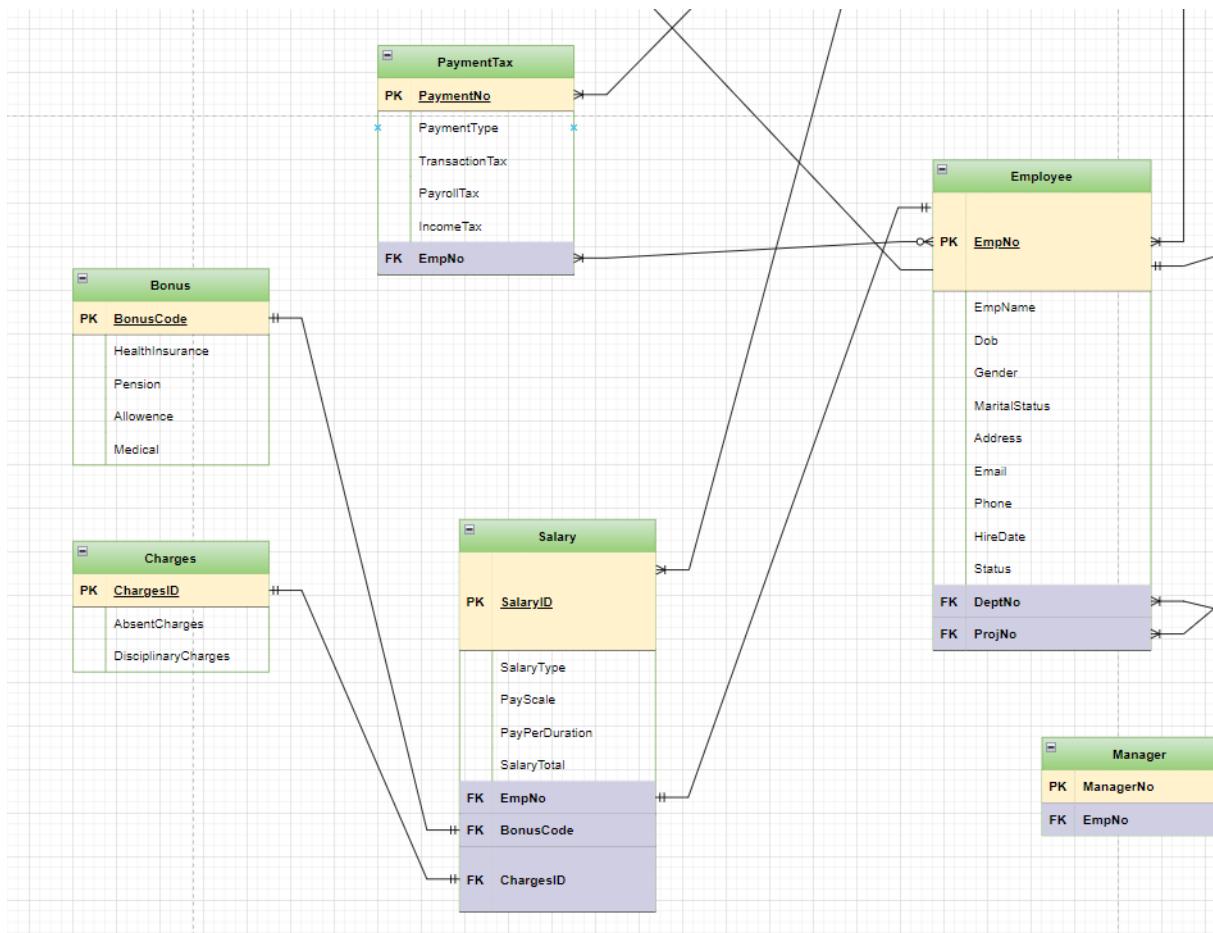
**Bonus**(BonusCode, HealthInsurance, Pension, Allowence, Medical)

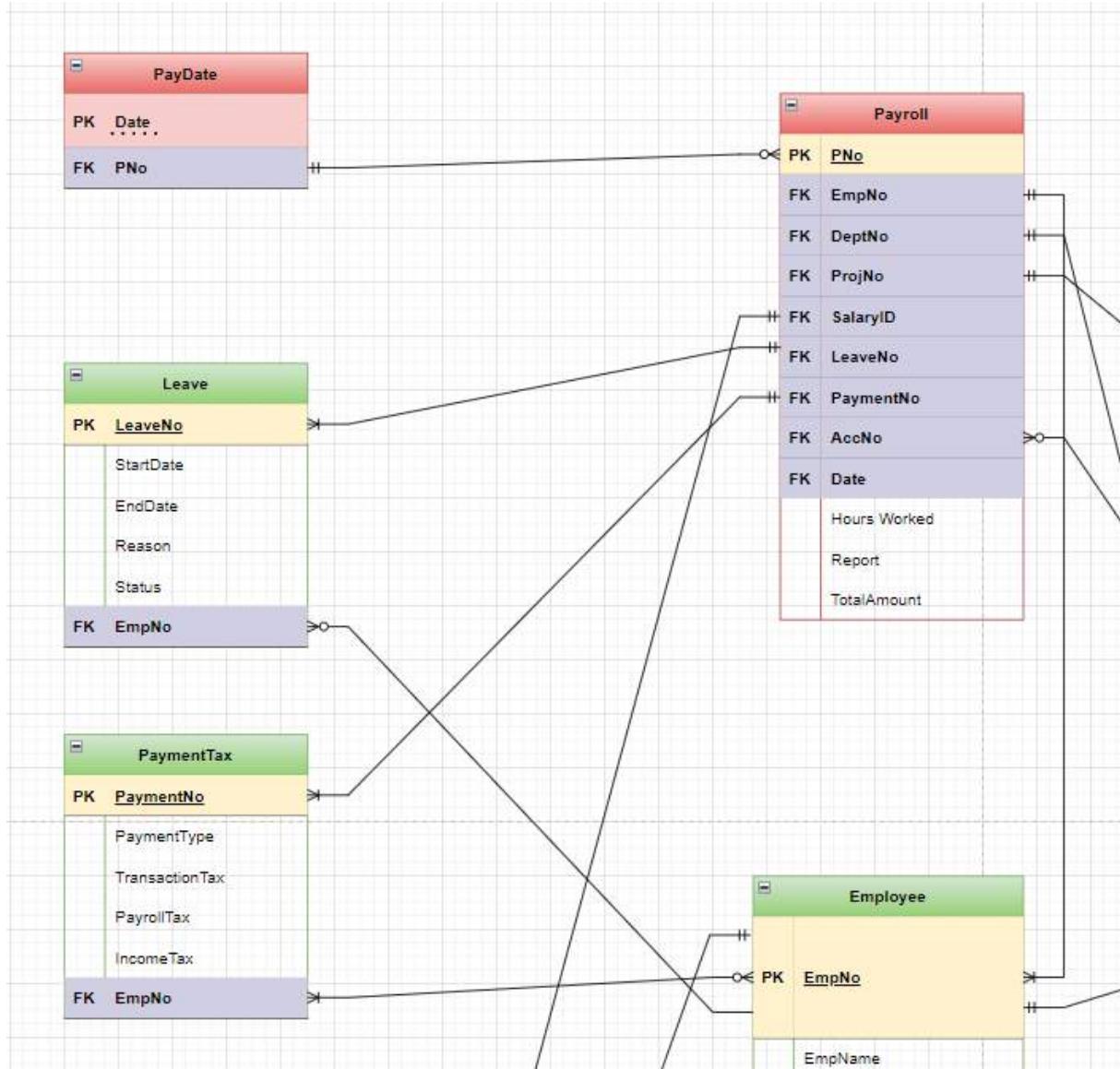
**PaymentTax**(PaymentNo, PaymentTax, TransactionTax, PayrollTax, IncomeTax, EmpNo)

**Leave**(LeaveNo, LeaveStartDate, LeaveEndDate, Reason, LeaveStatus, EmpNo)

## Logical Schema After Normalization







## Coding of Database Payroll

### Creating Database

```
create Database Payroll
```

### Table Employee:

```
Create Table Employee (
    EmpNo Int Not Null,
    EmpName Varchar(25) not null,
    DOB date,
```

```

Gender varchar(10),
MaritalStatus varchar(10),
EmpAddress varchar(30),
PhoneNo varchar(20),
HireDate Date,
Status varchar(10),
DeptNo int not null,
ProjNo int

constraint PK_EmpNo Primary Key (EmpNo)
)

```

## Changes/Upgradation of Employee

```

Alter Table Employee Add Constraint FK_EmpDeptNo Foreign Key (DeptNo) References
Department(DeptNo)
Alter Table Employee Add Constraint FK_EmpProjNo Foreign Key (ProjNo) References
Project(ProjNo)

```

### Procedure for insertion in Employee:

```

create procedure AddEmployee(@EmpID int, @Name varchar(25), @DoBb date, @Genderr
varchar(20), @Marital varchar(20), @Email varchar(30), @Phone varchar(20), @HireDate
date, @Statuss varchar(20), @Deptt int, @Projj int)
as begin
Insert into Employee(EmpNo, EmpName, DOB, Gender, MaritalStatus, EmpAddress, PhoneNo,
HireDate, Status, DeptNo, ProjNo) values (@EmpID, @Name, @DoBb, @Genderr, @Marital,
@email, @Phone, @HireDate, @Statuss, @Deptt, @Projj)
End

```

### Procedure for removal of Employee:

```

create procedure RemoveEmployee(@EmpID int)
as begin
Delete from Employee where EmpNo = @EmpID
End

```

## Insertion/Deletion of Employee

```

exec AddEmployee 5, 'Taha', '2001-01-23', 'Male', 'Unmarried',
'dadomazzoayo@yahoo.com', 41231113223, '2021-04-01', 'Inactive', 2, 2
exec RemoveEmployee 3

```

### Screenshot:

The image shows three separate SSMS windows, each displaying the results of a query against the Employee table. The first window shows the initial state of the table with four rows of data. The second window shows the execution of a stored procedure named 'RemoveEmployee' followed by a 'Select \* from Employee' command, which returns five rows of data, including the newly inserted row for employee ID 5. The third window shows the state of the table after the deletion, with only four rows remaining.

	EmpNo	EmpName	DOB	Gender	MaritalStatus	EmpAddress	PhoneNo	HireDate	Status	DeptNo	ProjNo
1	1	M.Hashim	2002-08-09	Male	Unmarried	abc@gmail.com	90078601	2022-07-02	Active	2	2
2	2	Ali Gauhar	2004-02-12	Male	Married	def@gmail.com	90078602	2022-07-01	Inactive	1	NULL
3	3	Naeem	2012-11-22	Male	Married	ghi@gmail.com	90078603	2022-02-01	Active	1	NULL
4	4	Youstra	2003-09-17	Female	Unmarried	jkl@gmail.com	90078604	2022-07-01	Active	2	NULL

	EmpNo	EmpName	DOB	Gender	MaritalStatus	EmpAddress	PhoneNo	HireDate	Status	DeptNo	ProjNo
1	1	M.Hashim	2002-08-09	Male	Unmarried	abc@gmail.com	90078601	2022-07-02	Active	2	2
2	2	Ali Gauhar	2004-02-12	Male	Married	def@gmail.com	90078602	2022-07-01	Inactive	1	NULL
3	3	Naeem	2012-11-22	Male	Married	ghi@gmail.com	90078603	2022-02-01	Active	1	NULL
4	4	Youstra	2003-09-17	Female	Unmarried	jkl@gmail.com	90078604	2022-07-01	Active	2	NULL
5	5	Taha	2001-01-23	Male	Unmarried	dedomazzoayo@yahoo.com	41231113223	2021-04-01	Inactive	2	2

	EmpNo	EmpName	DOB	Gender	MaritalStatus	EmpAddress	PhoneNo	HireDate	Status	DeptNo	ProjNo
1	1	M.Hashim	2002-08-09	Male	Unmarried	abc@gmail.com	90078601	2022-07-02	Active	2	2
2	2	Ali Gauhar	2004-02-12	Male	Married	def@gmail.com	90078602	2022-07-01	Inactive	1	NULL
3	4	Youstra	2003-09-17	Female	Unmarried	jkl@gmail.com	90078604	2022-07-01	Active	2	NULL
4	5	Taha	2001-01-23	Male	Unmarried	dedomazzoayo@yahoo.com	41231113223	2021-04-01	Inactive	2	2

## Table Department:

```
Create Table Department (
    DeptNo int not null,
    DeptName varchar (25) not null,
    DeptLocation varchar (30)

    constraint PK_DeptNo Primary Key (DeptNo)
)
```

## Changes/Upgradation of Department:

```
Alter Table Department alter Column ManagerNo int not null
Alter Table Department add constraint FK_DeptManagerNo Foreign Key (ManagerNo)
References Manager(ManagerNo)
```

### Procedure for insertion in Department:

```
create procedure AddDepartment(@DeptID int, @Name varchar(30), @Location varchar(30),
@ManagerID int)
as begin
Insert into Department(DeptNo, DeptName, DeptLocation, ManagerNo) values (@DeptID,
@Name, @Location, @ManagerID)
end
```

### Procedure for removal of Department:

```
create Procedure RemoveDepartment(@DeptID int)
```

```
as begin
Delete from Department where DeptNo = @DeptID
end
```

## Insertion/Deletion of Department:

```
exec AddDepartment 3, 'Cyber', 'Room 105', 3
exec RemoveDepartment 3
```

## Screenshot:

	DeptNo	DeptName	DeptLocation	ManagerNo
1	1	Electrical	Karachi	1
2	2	Technical	NorthNazimabad	2

	DeptNo	DeptName	DeptLocation	ManagerNo
1	1	Electrical	Karachi	1
2	2	Technical	NorthNazimabad	2
3	3	Cyber	Room 105	3

## Table Project:

```
Create Table Project (
ProjNo int not null,
ProjName varchar (25) not null,
ProjDetail varchar (30),
ProjLocation varchar (30) not null,
ProjStartDate Date not null,
ProjCost decimal,
DeptNo int not null,
constraint FK_ProjDeptNo Foreign Key (DeptNo) References Department(DeptNo),
constraint PK_ProjNo Primary Key (ProjNo)
)
```

## Changes/Upgradation of Project:

```
Alter Table Project Alter column ProjName Varchar(40) not null
```

### Procedure for insertion in Project:

```
create procedure AddProject(@ProjID int, @Name varchar(30), @Detail varchar(40),
@Location varchar(30), @Date date, @Cost decimal, @DeptID int)
```

```
as begin
Insert into Project(ProjNo, ProjName, ProjDetail, ProjLocation, ProjStartDate,
ProjCost, DeptNo) values (@ProjID , @Name , @Detail , @Location , @Date , @Cost ,
@DeptID)
end
```

### Procedure for removal of Project:

```
create Procedure RemoveProject(@ProjID int)
as begin
Delete from Project where ProjNo = @ProjID
end
```

### Insertion/Deletion of Project:

```
exec AddProject 3, 'CyberAnalysis', 'To Check for any Cyber Issue', 'CyberRoom 212',
'2020-02-1', 300000, 3
exec AddProject 4, 'CyberDefence', null, 'CyberRoom 213', '2020-01-12', 300000, 3
```

### Screenshot:

The screenshot displays two separate result sets from a SQL query execution. Both result sets are titled 'Results' and show data from a table with columns: ProjNo, ProjName, ProjDetail, ProjLocation, ProjStartDate, ProjCost, and DeptNo.

**Top Result Set (Initial Insert):**

ProjNo	ProjName	ProjDetail	ProjLocation	ProjStartDate	ProjCost	DeptNo
1	Electrical Appliances	Fix Electrical Stuff	Karachi Main Branch P-0 12439	2022-02-07	120000	1
2	Technical Work	NULL	North Nazimabad Pyara Line 212	2022-01-07	145000	2

**Bottom Result Set (Full Table):**

ProjNo	ProjName	ProjDetail	ProjLocation	ProjStartDate	ProjCost	DeptNo
1	Electrical Appliances	Fix Electrical Stuff	Karachi Main Branch P-0 12439	2022-02-07	120000	1
2	Technical Work	NULL	North Nazimabad Pyara Line 212	2022-01-07	145000	2
3	CyberAnalysis	To Check for any Cyber Issue	CyberRoom 212	2020-02-01	300000	3
4	CyberDefence	NULL	CyberRoom 213	2020-01-12	300000	3

### Table Manager:

```
Create Table Manager (
ManagerNo int not null,
EmpNo int not null unique,

constraint PK_ManagerNo Primary Key (ManagerNo),
constraint FK_ManagerEmpNo Foreign Key (EmpNo) References Employee(EmpNo),
)
```

### Insertion/Deletion of Manager:

#### Procedure for insertion in Manager:

```
create procedure AddManager(@ManagerID int, @EmpID int)
as begin
```

```
Insert into Manager(ManagerNo, EmpNo) values (@ManagerID, @EmpID)
end
```

### Procedure for removal of Manager:

```
create Procedure RemoveManager(@ManagerID int)
as begin
Delete from Manager where ManagerNo = @managerID
End
```

```
exec AddManager 3, 4
```

### Screenshot:

The screenshot shows two vertically stacked SSMS result sets. Both have tabs for 'Results' and 'Messages'. The top result set displays a table with columns 'ManagerNo' and 'EmpNo'. It contains two rows: (1, 2) and (2, 1). The bottom result set also displays a table with the same columns. It contains three rows: (1, 2), (2, 1), and (3, 4).

ManagerNo	EmpNo
1	2
2	1

ManagerNo	EmpNo
1	2
2	1
3	4

### Table Bank:

```
Create Table Bank (
AccNo int not null unique,
BankName int,
Branch varchar (20),
BranchCode int,
EmpNo int not null

constraint PK_AccNo Primary Key (AccNo),
constraint FK_BankEmpNo Foreign Key (EmpNo) References Employee(EmpNo),
)
```

### Insertion/Deletion of Bank:

#### Procedure for insertion in Bank:

```
create procedure AddBank(@AccNo int, @Name varchar(30), @Branch varchar(40),
@Branchcode varchar(30), @Emp int)
as begin
Insert into Bank(AccNo, BankName, Branch, BranchCode, EmpNo) values (@AccNo, @Name,
@Branch, @Branchcode, @Emp)
```

End

### Procedure for removal of Bank:

```
create Procedure RemoveBank(@AccNo int)
as begin
Delete from Bank where AccNo = @AccNo
end

exec AddBank 1211, 'Habib Bank', 'Sector 4/D', '209', 5
```

### Screenshot:

	AccNo	BankName	Branch	BranchCode	EmpNo
1	1711	Al Falah	Baldia	2002	1
2	1712	WomensBankLTD	NorthNazimabad	212	2

	AccNo	BankName	Branch	BranchCode	EmpNo
1	1211	Habib Bank	Sector 4/D	209	5
2	1711	Al Falah	Baldia	2002	1
3	1712	WomensBankLTD	NorthNazimabad	212	2

### Table Bonus:

```
Create Table Bonus (
BonusCode int not null,
HealthInsurance decimal,
Pension decimal,
Allowence decimal,
Medical decimal,
constraint PK_BonusCode Primary Key (BonusCode)
)
```

### Changes/Upgradation of Bonus:

```
Alter Table Bonus Add constraint DF_HealthInsurance Default 0 for HealthInsurance
Alter Table Bonus Add constraint DF_Pension Default 0 for Pension
Alter Table Bonus Add constraint DF_Allowence Default 0 for Allowence
Alter Table Bonus Add constraint DF_Medical Default 0 for Medical
```

### Insertion/Deletion of Bonus:

#### Procedure for insertion in Bonus:

```

create procedure AddBonus(@BonusCode int, @Health decimal, @pension decimal, @allow decimal, @medical decimal)
as begin
Insert into Bonus(BonusCode, HealthInsurance, Pension, Allowence, Medical) values
(@BonusCode , @Health , @pension , @allow , @medical )
end

```

### Procedure for removal of Bonus:

```

create Procedure RemoveBonus(@Bonus int)
as begin
Delete from Bonus where BonusCode = @Bonus
End

```

```
exec AddBonus 2, 0, 0, 1000, 2000
```

### Screenshot:

The screenshot shows two separate SSMS result sets. Both panes have a title bar with '100 %' and tabs for 'Results' and 'Messages'. The first pane displays a single row of data in a table:

	BonusCode	HealthInsurance	Pension	Allowence	Medical
1	1	2000	0	2000	3000

The second pane shows the state of the table after an insertion, containing two rows of data:

	BonusCode	HealthInsurance	Pension	Allowence	Medical
1	1	2000	0	2000	3000
2	2	0	0	1000	2000

### Table Charges:

```

Create Table Charges (
ChargesID int not null,
AbsentCharges decimal,
DisciplinaryCharges decimal,
constraint PK_ChargesID Primary Key (ChargesID)
)

```

### Changes/Upgradation of Charges:

```

Alter Table Charges Add constraint DF_AbsentCharges Default 0 for AbsentCharges
Alter Table Charges Add constraint DF_DisciplinaryCharges Default 0 for
DisciplinaryCharges

```

### Insertion/Deletion of Charges:

### Procedure for insertion in Charges:

```
create procedure Addcharges(@Id int, @charge1 decimal, @charge2 decimal)
as begin
Insert into Charges(ChargesID, AbsentCharges, DisciplinaryCharges) values (@id ,
@charge1 , @charge2)
end
```

### Procedure for removal of Charges:

```
create Procedure Removecharges(@id int)
as begin
Delete from charges where ChargesID = @id
end

exec AddCharges 2, 300, 2000
```

### Screenshot:

The screenshot shows two separate SSMS result sets. Both grids have columns: 'ChargesID', 'AbsentCharges', and 'DisciplinaryCharges'. The top grid has one row with values: ChargesID=1, AbsentCharges=1000, DisciplinaryCharges=5000. The bottom grid has two rows with values: ChargesID=1, AbsentCharges=1000, DisciplinaryCharges=5000; and ChargesID=2, AbsentCharges=300, DisciplinaryCharges=2000.

	ChargesID	AbsentCharges	DisciplinaryCharges
1	1	1000	5000

	ChargesID	AbsentCharges	DisciplinaryCharges
1	1	1000	5000
2	2	300	2000

### Table Salary:

```
create table Salary (
SalaryID int not null,
SalaryType varchar (25) not null,
PayScale decimal default 0,
PayPerDuration varchar (25),
SalaryTotal decimal,
EmpNo int not null unique,
BonusCode int,
ChargesID int

Constraint PK_SalaryID Primary Key (SalaryID),
Constraint FK_EmpNoSalary Foreign Key (EmpNo) References Employee(EmpNo),
Constraint FK_BonusCodeSalary Foreign Key (BonusCode) References Bonus(BonusCode),
Constraint FK_ChargesIDSALARY Foreign Key (ChargesID) References Charges(ChargesID)
)
```

### Insertion/Deletion of Salary:

### Procedure for insertion in Salary:

```

create procedure AddSalary(@SID int, @SType varchar(30), @PS decimal, @PPD
varchar(20), @emp int, @BC int, @CID int)
as begin
Insert Into Salary(SalaryID, SalaryType, PayScale, PayPerDuration, EmpNo, BonusCode,
ChargesID) values (@SID, @SType, @PS, @PPD, @emp, @BC, @CID)
Update Salary Set SalaryTotal = (Select HealthInsurance+Pension+Allowence+Medical-
AbsentCharges-DisciplinaryCharges from Bonus, Charges where BonusCode = @BC AND
ChargesID = @CID)+@PS where SalaryID = @SID
End

```

### Procedure for removal of Salary:

```

create procedure RemoveSalary(@Emp int)
as begin
Delete from Salary where EmpNo = @emp
End

```

**Salary Will be removed on EmployeeNo**

```
exec AddSalary 2, 'Hourly', 800, '1 Hour', 2, 1, 1
```

### Screenshot:

	SalaryID	SalaryType	PayScale	PayPerDuration	SalaryTotal	EmpNo	BonusCode	ChargesID
1	1	Monthly	120000	1 Month	121000	1	1	1

	SalaryID	SalaryType	PayScale	PayPerDuration	SalaryTotal	EmpNo	BonusCode	ChargesID
1	1	Monthly	120000	1 Month	121000	1	1	1
2	2	Hourly	800	1 Hour	1800	2	1	1

### Table PaymentTax:

```

Create Table PaymentTax (
PaymentNo int not null,
PaymentType varchar (25),
TransactionTax decimal not null,
PayrollTax decimal,
IncomeTax decimal,
EmpNo int

constraint PK_PaymentNo Primary Key (PaymentNo),
constraint FK_PaymentTaxEmpNo Foreign Key (EmpNo) references Employee(EmpNo)
)

```

## Changes/Upgradation of PaymentTax:

```
Alter Table PaymentTax Add constraint DF_TransactionTax Default 0 for TransactionTax
Alter Table PaymentTax Add constraint DF_PayrollTax Default 0 for PayrollTax
Alter Table PaymentTax Add constraint DF_IncomeTax Default 0 for IncomeTax
```

## Insertion/Deletion of PaymentTax:

```
Insert Into PaymentTax(PaymentNo,PaymentType, TransactionTax, PayrollTax, IncomeTax, EmpNo) Values (1, 'Bank Transaction', 250, 50, 300,1)
```

### Procedure for insertion in PaymentTax:

```
create procedure AddPaymentTax(@Pno int, @Ptype varchar(30), @Tt decimal, @Pt Decimal, @It decimal, @emp int)
as begin
Insert Into
PaymentTax(PaymentNo,PaymentType,TransactionTax,PayrollTax,IncomeTax,EmpNo) values
(@Pno, @Ptype, @Tt, @Pt, @It, @emp)
end
```

### Procedure for removal of PaymentTax:

```
create procedure RemovePaymentTax(@Pno int)
as begin
Delete From PaymentTax where PaymentNo = @Pno
End

exec AddPaymentTax 2, 'Cash', 0, 50, 350, 4
```

## Screenshot:

The screenshot displays two separate result sets from a SQL query. The first result set shows a single record with PaymentNo 1, PaymentType 'Bank Transaction', TransactionTax 250, PayrollTax 50, IncomeTax 300, and EmpNo 1. The second result set shows two records: one with PaymentNo 1, PaymentType 'Bank Transaction', TransactionTax 250, PayrollTax 50, IncomeTax 300, and EmpNo 1; and another with PaymentNo 2, PaymentType 'Cash', TransactionTax 0, PayrollTax 50, IncomeTax 350, and EmpNo 4.

	PaymentNo	PaymentType	TransactionTax	PayrollTax	IncomeTax	EmpNo
1	1	Bank Transaction	250	50	300	1

	PaymentNo	PaymentType	TransactionTax	PayrollTax	IncomeTax	EmpNo
1	1	Bank Transaction	250	50	300	1
2	2	Cash	0	50	350	4

## Table Leave:

```
Create Table Leave (
```

```

LeaveNo int not null,
StartDate Date,
EndDate Date,
Reason varchar (25),
Status varchar (10),
EmpNo int

constraint PK_LeaveNo Primary Key (LeaveNo),
constraint FK_LeaveEmpNo Foreign Key (EmpNo) references Employee(EmpNo)
)

```

## Insertion/Deletion of Leave:

### Procedure for insertion in Leave:

```

create procedure AddLeave(@Lno int, @Sdate date, @eDate date, @reason varchar(30),
@status varchar(20), @emp int)
as begin
Insert Into Leave(LeaveNo,StartDate,EndDate,Reason,Status,EmpNo) values (@Lno, @Sdate,
@eDate, @reason, @status, @emp)
end

```

### Procedure for removal of Leave:

```

create procedure RemoveLeave(@Lno int)
as begin
Delete From Leave where LeaveNo = @Lno
End

exec AddLeave 1, '2022-07-05', '2022-07-20', 'Sick Leave', 'Active', 2

```

## Screenshot:

The screenshot shows two separate result sets from SSMS. The top result set displays the schema of the 'Leave' table with columns: LeaveNo, StartDate, EndDate, Reason, Status, and EmpNo. The bottom result set shows the execution of the 'AddLeave' stored procedure, which has inserted a new row into the 'Leave' table with the values: LeaveNo = 1, StartDate = 2022-07-05, EndDate = 2022-07-20, Reason = Sick Leave, Status = Active, and EmpNo = 2.

LeaveNo	StartDate	EndDate	Reason	Status	EmpNo
1	2022-07-05	2022-07-20	Sick Leave	Active	2

## Table PayrollDetail:

```

Create Table PayrollDetail (
PNo int not null unique,

```

```

EmpNo int not null,
DeptNo int not null,
ProjNo int,
SalaryID int not Null,
LeaveNo int,
PaymentNo int not null,
AccNo int,
Date Date not null,
HoursWorked int,
Report varchar (40),
TotalAmmount decimal not null

constraint PK_PNo Primary Key (PNo),
constraint FK_PayrollEmpNo Foreign Key (EmpNo) references Employee(EmpNo),
constraint FK_PayrollDeptNo Foreign Key (DeptNo) references Department(DeptNo),
constraint FK_PayrollProjNo Foreign Key (ProjNo) references Project(ProjNo),
constraint FK_PayrollLeaveNo Foreign Key (LeaveNo) references Leave(LeaveNo),
constraint FK_PayrollPaymentNo Foreign Key (PaymentNo) references
PaymentTax(PaymentNo),
constraint FK_PayrollAccNo Foreign Key (AccNo) references Bank(AccNo)
)

```

## Changes/Upgradation of PayrollDetail:

```
Alter Table PayrollDetail Add Constraint FK_PayrollDetailSalaryID Foreign Key
(SalaryID) References Salary(SalaryID)
```

## Insertion/Deletion of PayrollDetail:

```

Insert Into PayrollDetail(PNo, EmpNo, DeptNo, ProjNo, SalaryID, PaymentNo, AccNo,
Date, HoursWorked, Report, TotalAmmount) Values (1011, 1, (Select DeptNo from Employee
where EmpNo = 1), (Select ProjNo from Employee where EmpNo = 1), (Select SalaryID from
Salary where EmpNo = 1), (Select PaymentNo from PaymentTax where EmpNo = 1), (Select
AccNo from Bank where EmpNo = 1), '07-04-2022',12, 'Good Code of Conduct', (Select
SalaryTotal from Salary where EmpNo = 1)-(Select TransactionTax+PayrollTax+IncomeTax
from PaymentTax where EmpNo = 1))

```

```
Update PayrollDetail Set ProjNo = (Select ProjNo from Employee where EmpNo = 1)
```

## Procedure for insertion in Payroll Monthly:

```

create procedure AddPayrollMonthly(@pnoo int, @eno int, @d date, @hw int, @rprt
varchar (50))
as begin
Insert Into PayrollDetail(PNo, EmpNo, DeptNo, ProjNo, SalaryID, LeaveNo, PaymentNo,
AccNo, Date, HoursWorked, Report, TotalAmmount) Values (@pnoo, @eno,(Select DeptNo
from Employee where EmpNo = @eno),(Select ProjNo from Employee where EmpNo =
@eno),(Select SalaryID from Salary where EmpNo = @eno),(Select LeaveNo from Leave
where EmpNo = @eno),(Select PaymentNo from PaymentTax where EmpNo = @eno),(Select
AccNo from Bank where EmpNo = @eno),@d, @hw, @rprt, (Select SalaryTotal from Salary
where EmpNo = @eno)-(Select TransactionTax+PayrollTax+IncomeTax from PaymentTax where
EmpNo = @eno))
end

```

### Procedure for insertion in Payroll Weekly:

```
create procedure AddPayrollWeekly(@pnoo int, @eno int, @d date, @hw int, @rppt varchar(50))
as begin
Insert Into PayrollDetail(PNo, EmpNo, DeptNo, ProjNo, SalaryID, LeaveNo, PaymentNo, AccNo, Date, HoursWorked, Report, TotalAmmount) Values (@pnoo, @eno,(Select DeptNo from Employee where EmpNo = @eno),(Select ProjNo from Employee where EmpNo = @eno),(Select SalaryID from Salary where EmpNo = @eno),(Select LeaveNo from Leave where EmpNo = @eno),(Select PaymentNo from PaymentTax where EmpNo = @eno),(Select AccNo from Bank where EmpNo = @eno),@d, @hw, @rppt, (Select SalaryTotal from Salary where EmpNo = @eno)*7-(Select TransactionTax+PayrollTax+IncomeTax from PaymentTax where EmpNo = @eno))
end
```

### Procedure for insertion in Payroll Hourly:

```
create procedure AddPayrollHourly(@pnoo int, @eno int, @d date, @hw int, @rppt varchar(50))
as begin
Insert Into PayrollDetail(PNo, EmpNo, DeptNo, ProjNo, SalaryID, LeaveNo, PaymentNo, AccNo, Date, HoursWorked, Report, TotalAmmount) Values (@pnoo, @eno,(Select DeptNo from Employee where EmpNo = @eno),(Select ProjNo from Employee where EmpNo = @eno),(Select SalaryID from Salary where EmpNo = @eno),(Select LeaveNo from Leave where EmpNo = @eno),(Select PaymentNo from PaymentTax where EmpNo = @eno),(Select AccNo from Bank where EmpNo = @eno),@d, @hw, @rppt, ((Select SalaryTotal from Salary where EmpNo = @eno)-(Select TransactionTax+PayrollTax+IncomeTax from PaymentTax where EmpNo = @eno))*@Hw)
end
```

### Procedure for removal of Payroll:

```
create procedure RemovePayrollDetail (@pno int)
as begin
delete from PayrollDetail where Pno = @Pno
end
```

### Screenshot:

The screenshot displays two separate result sets from a SQL query execution.

**Top Result Set (Payroll Weekly Insertion):**

PNo	EmpNo	DeptNo	ProjNo	SalaryID	LeaveNo	PaymentNo	AccNo	Date	HoursWorked	Report	TotalAmmount
1011	1	2	2	1	NULL	1	1711	2022-07-04	12	Good Code of Conduct	120400

**Bottom Result Set (Payroll Hourly Insertion):**

PNo	EmpNo	DeptNo	ProjNo	SalaryID	LeaveNo	PaymentNo	AccNo	Date	HoursWorked	Report	TotalAmmount
1011	1	2	2	1	NULL	1	1711	2022-07-04	12	Good Code of Conduct	120400
2032	2	1	NULL	2	1	3	1712	2022-07-05	16	Average	18400

```
exec AddPayrollWeekly 2032, 2, '2022-07-05', 16, 'Average'
Select * from PayrollDetail
```

100 %

PNo	EmpNo	DeptNo	ProjNo	SalaryID	LeaveNo	PaymentNo	AccNo	Date	HoursWorked	Report	TotalAmmount
1011	1	2	2	1	NULL	1	1711	2022-07-04	12	Good Code of Conduct	120400
2032	2	1	NULL	2	1	3	1712	2022-07-05	16	Average	11950

```
exec AddPayrollMonthly 2032, 1, '2022-07-05', 16, 'Average'
Select * from PayrollDetail
```

100 %

PNo	EmpNo	DeptNo	ProjNo	SalaryID	LeaveNo	PaymentNo	AccNo	Date	HoursWorked	Report	TotalAmmount
1011	1	2	2	1	NULL	1	1711	2022-07-04	12	Good Code of Conduct	120400
2032	1	2	2	1	NULL	1	1711	2022-07-05	16	Average	120400

## Table PayDate:

```
create table PayDate (
Date Date,
PNo int

Constraint PK_DatePayDate Primary Key(Date),
Constraint FK_PNoPayDate Foreign Key(PNo) References PayrollDetail(PNo)
)
```

## Insertion/Deletion of PayDate:

```
Insert Into PayDate(Date, PNo) Values ('2022-07-04', 1011)
```

### Procedure for insertion in PayDate:

```
create procedure AddPayDate(@D date, @Pno int)
as begin
Insert Into PayDate(Date, PNo) values (@D, @Pno)
end
```

### Procedure for removal of PayDate:

```
create procedure RemovePayDate(@Pno int)
as begin
Delete From Paydate where Pno = @Pno
end
```

### Function for retrieving PayDate using Employee No:

```
create function ShowPayDateEmployee(@Emp int)
returns table as
return
(
Select pd.Date, p.PNo, e.EmpNo, EmpName, TotalAmmount
from PayDate pd, PayrollDetail p, Employee e
where pd.Date = p.Date and p.EmpNo = e.EmpNo AND p.EmpNo = @emp
)
```

```
SELECT * FROM ShowPayDateEmployee(1)
```

#### Function for retrieving PayDate using Date:

```
create function ShowPayDate(@D date)
returns table as
return
(
Select pd.Date, p.PNo, e.EmpNo, EmpName, TotalAmmount
from PayDate pd, PayrollDetail p, Employee e
where pd.Date = p.Date and p.EmpNo = e.EmpNo AND pd.Date = @D
)

SELECT * FROM ShowPayDate('2022-07-04')
```

#### Screenshot:

	Date	PNo
1	2022-07-04	1011

## Queries Tested on Database

#### Testing Employee, Manager Relationship

```
Select M.ManagerNo, M.EmpNo, E.EmpName
from Manager M, Employee E
where E.EmpNo = M.EmpNo
```

SQLQuery1.sql - LA...2SDCGJ\thund (54)\* ✎ X

```
Select M.ManagerNo, M.EmpNo, E.EmpName
from Manager M, Employee E
where E.EmpNo = M.EmpNo
```

100 %

Results Messages

	ManagerNo	EmpNo	EmpName
1	1	2	Ali Gauhar
2	2	1	M.Hashim

### Testing Manager, Employee, Department and Project Relationship:

```
Select E.EmpNo, E.EmpName, B.AccNo, D.DeptName, D.DeptNo, P.ProjNo
from Employee E, Bank B, Department D, Project P
where E.EmpNo = B.EmpNo AND E.DeptNo = D.DeptNo AND P.DeptNo = D.DeptNo
```

SQLQuery1.sql - LA...2SDCGJ\thund (54)\* X

```

Select E.EmpNo, E.EmpName, B.AccNo, D.DeptName, D.DeptNo, P.ProjNo
from Employee E, Bank B, Department D, Project P
where E.EmpNo = B.EmpNo AND E.DeptNo = D.DeptNo AND P.DeptNo = D.DeptNo

Select * from Manager
Select * from Employee
Select * from Department
Select * from Project

```

100 % ▾

Results Messages

	EmpNo	EmpName	AccNo	DeptName	DeptNo	ProjNo
1	1	M.Hashim	1711	Technical	2	2
2	2	Ali Gauhar	1712	Electrical	1	1

	ManagerNo	EmpNo	DeptNo
1	1	2	1
2	2	1	2

	EmpNo	EmpName	DOB	Gender	MaritalStatus	EmpAddress	PhoneNo	HireDate	Status	DeptNo	ProjNo
1	1	M.Hashim	2002-08-09	Male	Unmarried	abc@gmail.com	90078601	2022-07-02	Active	2	2
2	2	Ali Gauhar	2004-02-12	Male	Married	def@gmail.com	90078602	2022-07-01	Inactive	1	NULL
3	3	Naeem	2012-11-22	Male	Married	ghi@gmail.com	90078603	2022-02-01	Active	1	NULL
4	4	Youstra	2003-09-17	Female	Unmarried	jkl@gmail.com	90078604	2022-07-01	Active	2	NULL

	DeptNo	DeptName	DeptLocation	ManagerNo
1	1	Electrical	Karachi	1
2	2	Technical	NorthNazimabad	2

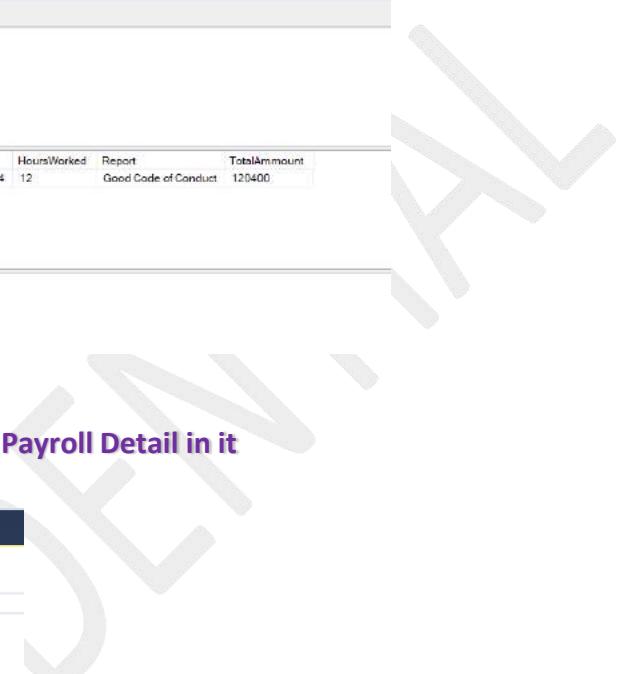
	ProjNo	ProjName	ProjDetail	ProjLocation	ProjStartDate	ProjCost	DeptNo
1	1	Electrical Appliances	Fix Electrical Stuff	Karachi Main Branch P-0 12439	2022-02-07	120000	1
2	2	Technical Work	NULL	North Nazimabad Pyara Line 212	2022-01-07	145000	2

### Testing PayDate and Payroll Relationship

```

Select p.PNo, d.Date, p.EmpNo
from PayDate d, PayrollDetail p
where d.Date = '2022-07-04' AND EmpNo = 1

```



SQLQuery1.sql - LA...2SDCG\thund (54)\* - X

```
Select p.PNo, d.Date, p.EmpNo  
from PayDate d, PayrollDetail p  
where d.Date = '2022-07-04' AND EmpNo = 1  
  
Select * from PayrollDetail  
Select * from PayDate
```

100 %

Results Messages

PNo	Date	EmpNo
1011	2022-07-04	1

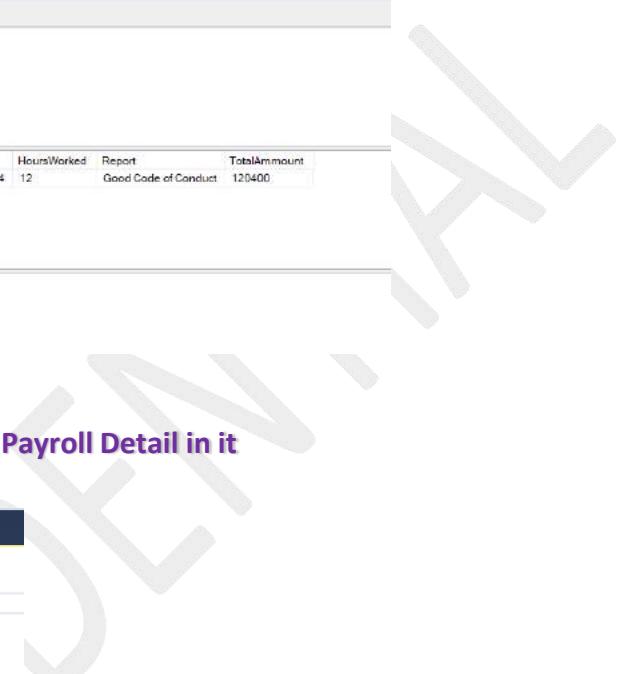
  

PNo	EmpNo	DeptNo	ProjNo	SalaryID	LeaveNo	PaymentNo	AccID	Date	HoursWorked	Report	TotalAmount
1011	1	2	2	1	NULL	1	1	2022-07-04	12	Good Code of Conduct	120400

Date	PNo
2022-07-04	1011

### Testing PayDate with a date that has No Payroll Detail in it



SQLQuery1.sql - LA...2SDCG\thund (54)\* - X

```
Select * from PayDate where Date = '2022-07-06'  
Select * from PayDate
```

100 %

Results Messages

Date	PNo
2022-07-04	1011