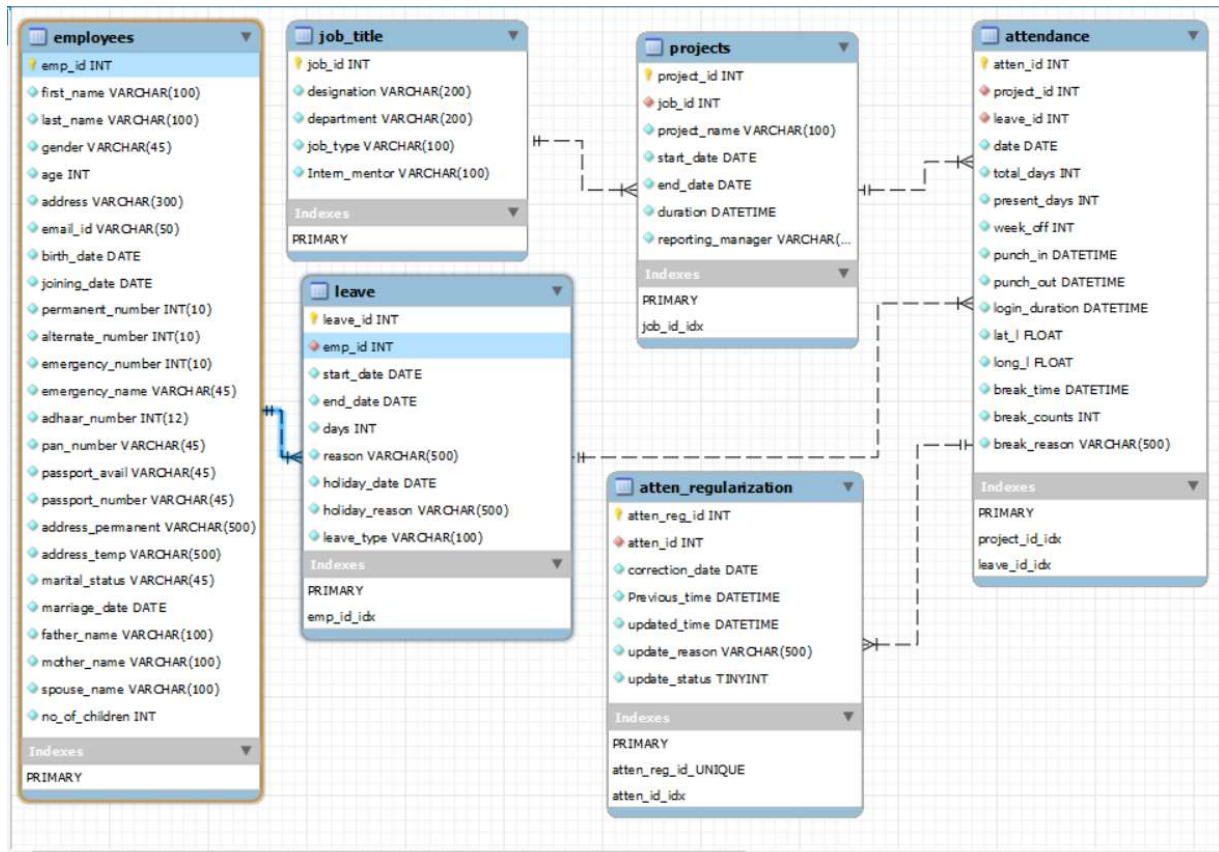


HRMS database document description

Version:- 1.0

Author:- Abhishek Patel



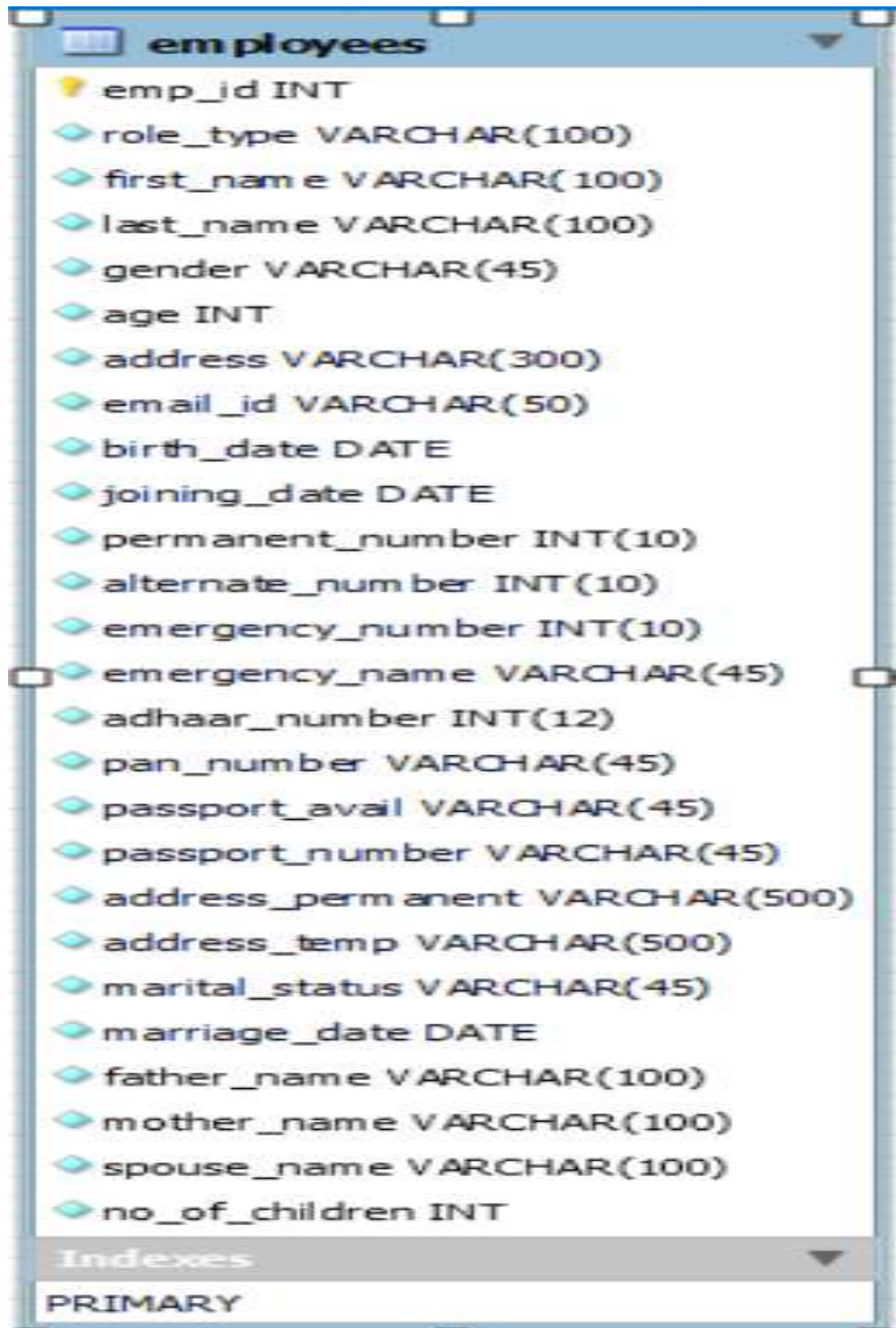
ER Diagram for the HRMS:-

There are tables in the database for the HRMS project.

1. **Employees**:- contains the data for the employee.
2. **Job_title**:- contains the data for the job-related title.
3. **Projects**:-contains the data for the project related details.
4. **Attendance**:- contains the data for the attendance such as punch in, punch out.
5. **Leaves**:- Leave related data such as several leaves applied, total leaves.
6. **Atten_regularization**:- contains the data for the attendance correction details.

Tables:-

Employees:-



The image shows a screenshot of a database management tool displaying the structure of a table named 'employees'. The table has 28 columns with various data types and lengths. The columns are listed in a scrollable area, and the 'Indexes' section at the bottom shows a 'PRIMARY' index.

Column Name	Data Type
emp_id	INT
role_type	VARCHAR(100)
first_name	VARCHAR(100)
last_name	VARCHAR(100)
gender	VARCHAR(45)
age	INT
address	VARCHAR(300)
email_id	VARCHAR(50)
birth_date	DATE
joining_date	DATE
permanent_number	INT(10)
alternate_number	INT(10)
emergency_number	INT(10)
emergency_name	VARCHAR(45)
adhaar_number	INT(12)
pan_number	VARCHAR(45)
passport_avail	VARCHAR(45)
passport_number	VARCHAR(45)
address_permanent	VARCHAR(500)
address_temp	VARCHAR(500)
marital_status	VARCHAR(45)
marriage_date	DATE
father_name	VARCHAR(100)
mother_name	VARCHAR(100)
spouse_name	VARCHAR(100)
no_of_children	INT

Indexes

PRIMARY

The Employees table contains the data for the employee and the admin. There are different types of column(Attribute) such as:-

Columns:-

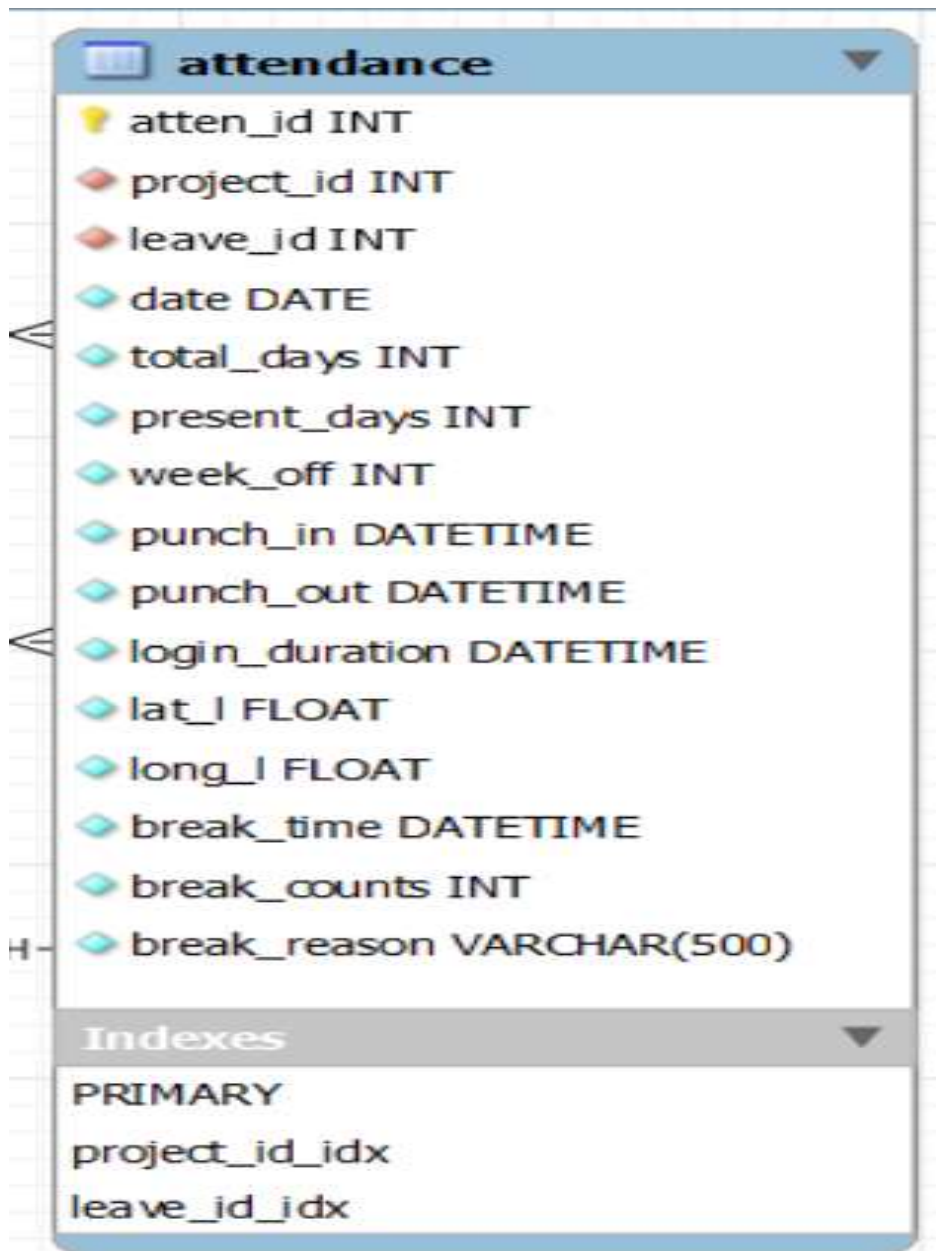
1. **emp_id**:- Data type for the emp_id is INT. This is also the primary key for the table and this can not be null. This column will be auto-incremented.
2. **role_type**:- The data type for this column is VARCHAR(100). It will store the type of employee whether the employee is admin/HR or normal employee. This column can not be null as the backend team will show the data based on the value of this column.
3. **first_name**:- This will store the first name of the employee and this attribute can not be null. The data type for this column is VARCHAR(100).
4. **last_name**:- This will store the last_name of the employee. This attribute can be null but only in one condition if the employee does not have the last name, in that case, the backend team will have to show the space instead of "null" from the database. The data type for this column is VARCHAR(100).
5. **gender**:- This will store the gender of the employee such as male, female, other. This column can not be null. The data type for this column is VARCHAR(100).
6. **age**:- This will store the age of the employee and this can not be null. Either the employee will enter the age or we will calculate the age as a subtraction value of the current date and the date of birth. The data type for this column is INT.
7. **email_id**:- This will store the active email id for the employee. The backend team will have to identify whether the email id is valid or invalid. This field can not be null and the data type for this field is VARCHAR(50).
8. **birth_date**:- This field will store the date of birth for the employee. The data type for this will be DATE and this field can not be null.
9. **joining_date**:- This field will store the joining date in which the employee has joined the organization. The data type for this field will be DATE and that can not be null.
10. **permanent_number**:- there will be three categories for the mobile number in the employee table, the first field will be the permanent number. This field will store the permanent mobile number of the employee and this field can not be null. The data type for this one will be INT(10).
11. **alternate_number**:- This will be used to store the alternative contact number for the employee. The data type for this one is INT(10) and this field can not be null. This can be null only in one condition if the employee does not have the alternative contact number but we will ask the employee to get the alternative contact number ASAP.
12. **emergency_number**:- This will store the emergency contact number for the employee. If there is any kind of emergency and the employee's mobile number will be out of reach, we will be using this contact number and this field can not be null. The data type for this field will be INT(10).
13. **emergency_name**:- This will store the name of the person whose mobile number will be provided by the employee as the emergency contact number. The data type for this column will be VARCHAR(50) and this field can not be null.

14. **adhaar_number**:- This will store the aadhaar card number of the employee for verification purposes. This field can not be null and the data type for this one is INT(12).
15. **pan_number**:- This will store the Pan card number of the employee for verification purposes. This field can not be null and the data type for this one is VARCHAR(45).
16. **passport_avail**:- This will store the passport availability option for the employee in the form of “yes” or “no” and if it is available then it is mandatory to enter the passport number in the next column. The data type for this field is VARCHAR(45) and it can not be null.
17. **passport_number**:- This will store the passport number of the employee. The data type for this field is VARCHAR(45) and it can not be null. It can only be null if the passport is not available for the employee and the backend team will have to show “passport is not available” instead of null.
18. **address_permanent**:- This will store the permanent address of the employee. The data type for this field is VARCHAR(500) and this field can not be null.
19. **address_temp**:- This will store the temporary or the present address of the employee. The data type for this field is VARCHAR(500) and this field can not be null. If the temporary address is not available for the employee, in that case, we will ask the employee to submit the address details ASAP and the backend team will have to show “Please update the address ASAP”.
20. **marital_status**:- This will store the marital status of the employee in the form of “married” or “unmarried”. The data type for this field is VARCHAR(45) and this cannot be null.
21. **marriage_date**:- This will store the marriage date of the employee. If his/her status is “married” in the marital_status column then the employee must have entered the marriage date. The data type for this field is DATE and this can not be null if the employee is unmarried then the backend team will have to show “<employee name> is unmarried” instead of a null value.
22. **father_name**:- This will store the father name of the employee. The data type for this field is VARCHAR(100) and this field can not be null.
23. **mother_name**:- This will store the mother name of the employee. The data type for this field is VARCHAR(100) and this field can not be null.
24. **spouse_name**:- This will store the spouse name of the employee. If the employee is unmarried then the backend team need not show the data from this column. The data type for this field is VARCHAR(100).
25. **no_of_children**:- This will store the children count of the employee. The data type for this field is INT and if the employee does not have children then the backend team need not show the data from this attribute.

Attendance:-

This table will contain the data for attendance such as punch in, punch out, breaks, login durations. Based on this table, we will calculate the performance of the employee and Count the payroll days.

Columns:-



attendance	
⚡	atten_id INT
🔴	project_id INT
🔴	leave_id INT
🔵	date DATE
🔵	total_days INT
🔵	present_days INT
🔵	week_off INT
🔵	punch_in DATETIME
🔵	punch_out DATETIME
🔵	login_duration DATETIME
🔵	lat_l FLOAT
🔵	long_l FLOAT
🔵	break_time DATETIME
🔵	break_counts INT
🔵	break_reason VARCHAR(500)
Indexes	
PRIMARY	
project_id_idx	
leave_id_idx	

1. **atten_id:-** This is the primary key for this table and this can not be null. It will be auto-incremented with each entry in the table. The data type for this column is INT.
2. **project_id:-** This is the foreign key in the attendance table which connects the attendance table with the projects table. The data type for this column is INT.
3. **leave_id:-** This is the foreign key in the attendance table which connects attendance with the leaves table. The data type for this column is INT.

4. **date:-** This column will store the current local date. The data type for this one is DATE and this can not be null. The backend team will have to store the current date dynamically which should change according to the current local date.
5. **total_days:-** Total days will store the total number of days available in the current month. The data type for this column will be INT and this can not be null. The backend team will have to store the data dynamically in this field according to current months because not every month has the same number of days.
eg. If the month is February then the days will be either 28 or 29 according to the leap year.
6. **present_days:-** This field will record the data regarding the total number of present days of the employee in the current month. The data type for this one will be INT and this can not be null. The backend team will have to populate the data according to total days, week offs and the number of leaves taken by the employee.
Eg:- $\text{present_days} = [\text{total_days}(\text{attendance}) - \text{week_off}(\text{attendance}) + \text{days}(\text{leave})]$
7. **week_off:-** This field will record the data for the number of weeks offs provided by the company to the employee in the running month, week_offs can be Sunday or additional off according to the policy. The data type for this field will be INT and this can not be null. The backend team have to populate the data by matching the day, for example, if the day is "Sunday" then it will be counted as week_off.
8. **punch_in:-** This field will record the punch in time of the employee. Punch in time is the time when the employee will come to the office and press the punch in button in the application. The data type for this field is DATETIME and this can not be null. It can be null only in one condition if the employee is on leave or absent. In that case, the backend team will have to show the message "<employee name> is on leave or absent".
9. **punch_out:-** This field will record the punch-out time of the employee. Punch out time is the time when the employee will leave the office after completing the office hours and press the punch out button in the application. The data type for this field is DATETIME and this can not be null. It can be null only in one condition if the employee is on leave or absent. In that case, the backend team will have to show the message "<employee name> is on leave or absent".
10. **login_duration:-** login_duration will store the total login hours of the employee for a day. It will be calculated based on punch_in, punch_out and break_time. The data type for this field is DATETIME and this field can not be null. It can be null only in one condition if the employee is on leave or absent. In that case, the backend team will have to show the message "<employee name> is on leave or absent". The backend team will have to populate the data for this column by using the formula that is -
 $\text{login_duration} = [\text{punch_out}(\text{attendance}) - \text{punch_in}(\text{attendance}) - \text{break_time}(\text{attendance})]$.
11. **lat_l:-** This field will store the latitude value of the location at the time of punch in. The data type for this column is FLOAT and it can not be null if the employee is in the office and the employee is not on leave.
12. **long_l:-** This field will store the longitude value of the location at the time of punch in. The data type for this column is FLOAT and it can not be null if the employee is in the office and the employee is not on leave.

13. **break_time**:- This field will store the total time of break taken by the employee in a single day. The data type for this one is DATETIME and it can not be null if the employee has joined the office on the same day and is not on leave.
14. **break_count**:- It will store the total number of breaks taken by the employee in a day. The backend team will have to count the break and store the same in this field. The data type for this field is INT and it can not be null if the employee has joined the office on the same day and is not on leave.
15. **break_reason**:- It will record the reason for the breaks taken by the employee in a single day. The data type for this field is VARCHAR(500) and it can not be null if the employee is available in the office and is not on leave.

Job_title:-

This table will contain the data regarding designation, department, job_type and their(intern) mentor.

There are multiple columns containing the data according to the data type for the employee.



The image shows a screenshot of a database table definition for a table named 'job_title'. The table has five columns: 'job_id' (INT), 'designation' (VARCHAR(200)), 'department' (VARCHAR(200)), 'job_type' (VARCHAR(100)), and 'Intern_mentor' (VARCHAR(100)). The 'job_id' column is marked as the primary key with a yellow lightbulb icon. Below the column list, there is a section for 'Indexes' which shows a 'PRIMARY' index.

job_title	
job_id	INT
designation	VARCHAR(200)
department	VARCHAR(200)
job_type	VARCHAR(100)
Intern_mentor	VARCHAR(100)

Indexes
PRIMARY

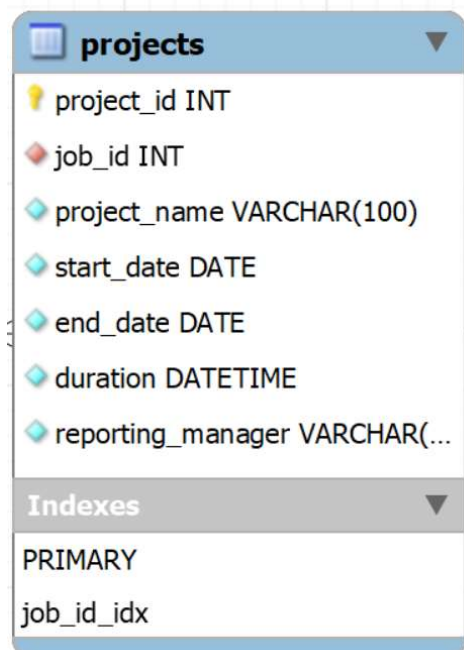
Columns:-

1. **job_id**:- This is the primary key for this table which is the combination of unique key and not null constraints. This is also set as auto-increment. The data type for this field is INT.
2. **designation**:- This field will contain the data for the current designation of the employee in the company. The data type for this column is VARCHAR(200) and this column can not be null.
3. **department**:- This field contains the data for the department in which employee is working in the company. It will store the data according to emp_id. Designations can be Intern, frontend developer, backend developer, analyst, QA. The data type for this column is VARCHAR(200) and it can not be null.
4. **job_type**:- This field will store the data related to job type which can be intern, part-time employee, full-time employee. The data type for this field is VARCHAR(100) and it can not be null.
5. **Intern_mentor**:- This field will store the name of the mentor who is assigned to the intern as a supervisor to take care of the progress of the intern and the projects allotted to the interns. The data type for this field is VARCHAR(100) and it can not be null.

Projects:-

This table contains the data related to the project details such as project name, its duration, reporting manager.

There are multiple columns available to store the data for project details.



projects	
project_id	INT
job_id	INT
project_name	VARCHAR(100)
start_date	DATE
end_date	DATE
duration	DATETIME
reporting_manager	VARCHAR(...)
Indexes	
PRIMARY	
job_id_idx	

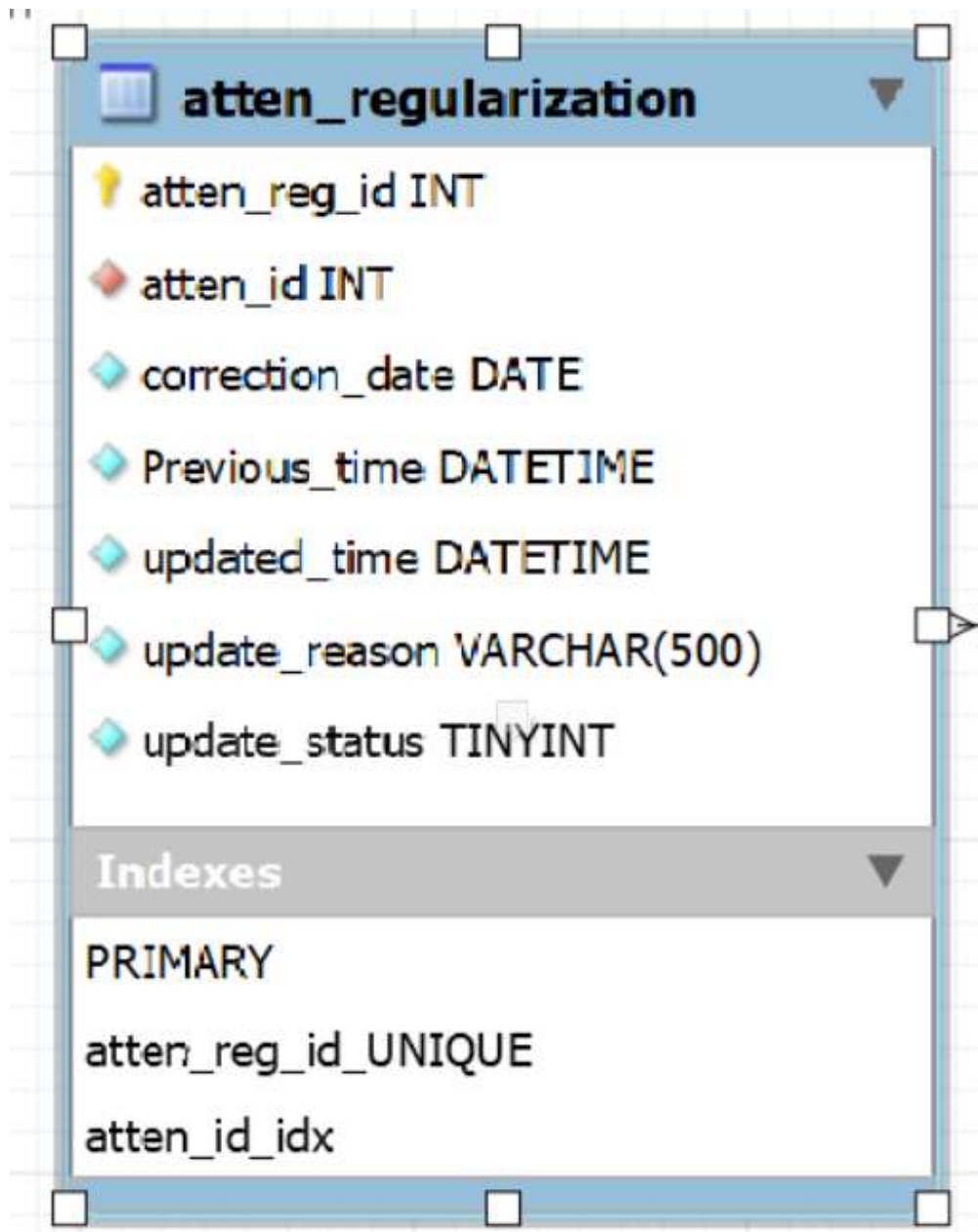
Columns:-

1. **project_id**:- This is the primary key for this table which is a combination of unique key and not null constraints. This is also set as auto-increment and the data type for this column is INT.
2. **job_id**:- This is the foreign key for the projects table. This column connects the projects table with the job table as job_id is also the primary key of the job_title table. The data type for this column is INT.
3. **project_name**:- This column will contain the data regarding the project's name which is currently available in the company and also which has completed their period in the company. The data type for this field is VARCHAR(100) and this can not be null.
4. **start_date**:- This column will store the start date of the project. The start date is the onboarding date of the project. The data type for this column is DATE and this can not be null if the project name is available and the employee is working on the project.
5. **end_date**:- This field will store the end date of the project. The end date of the project is the date on which the project has completed its period in the company. The data type for this field is DATE and this can not be null if the project has been completed.
6. **duration**:- This will record the duration of the project. Duration will be calculated based on the start_date and the end_date of the project. The data type for this field will be DATETIME and it can not be null if project_name, start_date, end_date is available in the table.
Eg. duration = [end_date(projects) - start_date(projects)]
7. **reporting_manager**:- This attribute will be store the name of the project manager/reporting manager of the running project and the completed project. The data type for this one is VARCHAR(100) and it can not be null if the project is in the running phase in the company.

Atten_regularization:-

This table will contain the data regarding the attendance correction and the reasons for the correction.

There are multiple columns available to store the data for correction details.



Columns:-

1. **atten_reg_id**:- This field will store the attendance regularization id of this table. This is the primary key for the table which is a combination of unique key and the not null constraints. The data type for this field is INT and it is set as auto-increment.
2. **atten_id**:- This field is the foreign key for this table which connects the atten_regularization table with the attendance table as atten_id is also the primary key of the attendance table. The data type for the atten_id is INT and it can not be null.
3. **correction_date**:- This field will store the date of the attendance correction for which the employee has requested the corrections to be done. The data type for this column is DATE and it can not be null if the employee is requesting the correction in attendance.
4. **previous_time**:- This field will store the previous time(punch-in/punch-out/breaks) of the attendance for the particular day on which correction is to be done. The data type for this column is DATETIME and this can not be null if the employee is requested for the correction.
5. **updated_time**:- This field will store the updated time(punch-in/punch-out/breaks) of the attendance for the particular day after the correction has been done by the HR/admin team. The data type for this column is DATETIME and this can not be null if the employee is requested for the correction and it has been done after reviewing the request. It is also important because on this basis only the Hr/admin team will review the details for the correction before approving the request.
6. **update_reason**:- This field will store the reason for the attendance correction, attendance correction can be for a punch-in time, punch-out time and breaks timings. The data type for this column is VARCHAR(500) and that can not be null as the employee must submit the reason and without it, the request will not be approved.
7. **update_status**:- This column will show the data for the status of the request, it can be either approved or rejected. The backend team will have to show only two options for the HR/admin team to update the status from their end. The data type for this column is TINYINT and it can be null only in one condition if the request is still pending or has not been reviewed yet.

Leave:-

This table will contain the data for the leaves of the employee such as total leaves, types of leaves, used leaves, leave duration, reasons for leaves.

There are multiple columns to store the data related to leaves.

leave

leave_id INT

emp_id INT

start_date DATE

end_date DATE

days INT

reason VARCHAR(500)

holiday_date DATE

holiday_reason VARCHAR(500)

leave_type VARCHAR(100)

Indexes

PRIMARY

emp_id_idx

Columns:-

1. **leave_id**:- This field will store the leave id of this table. This is the primary key for the table which is a combination of unique key and the not null constraints. The data type for this field is INT and it is set as auto-increment.
2. **emp_id**:- This field is the foreign key for this table which connects the leave table with the employee table as emp_id is also the primary key of the employee table. The data type for the atten_id is INT and it can not be null.

3. **start_date**:- This column will store the start date of the leaves. The start date is the date from which the employee will be on leave. The data type for this column is DATE and it can not be null if the employee is applying for the leave.
4. **end_date**:- This column will store the end date of the leaves. The end date is the date from which the employee will join the office after the completion of the leaves. The data type for this column is DATE and it can not be null if the employee has applied for the leaves.
5. **days**:- This column will store the total number of leaves taken by the employee in a month. The data type for this column is INT and it can not be null if the employee has applied for the leaves. The backend team will have to populate the data based on end_date and the start_date.

Eg. **days** = [end_date(leave) - start_date(leave)]

6. **reason**:- This column will record the reasons for the leaves taken by the employees. The user must enter the reason as without it request will not be approved. The data type for this column is VARCHAR(500) and it can not be null if the user has applied for the leaves.
7. **holiday_date**:- This column will store the holidays' dates according to company policy. Holidays dates are the dates on which the company is allowing employees to take the leaves on different occasions such as Holi, Diwali, independence day etc. and these leaves will be the paid leaves. The data type for this column will be DATE and it can not be null.
8. **holiday_reason**:- This column will store the reasons for the holidays provided by the company to the employees. These reasons can be Holi, Diwali, independence day etc. The data type for this column will be VARCHAR(500) and it can not be null.
9. **leave_type**:- This column will store the type of leaves which can be availed by the employees. These types can be paid leaves, unpaid leaves, sick leaves, casual leaves etc. backend team will have to restrict the users so that users will enter the specific name of the leaves and it will also be useful at the time of data fetching.