TOPIC: Payroll Management System

Contents

INTRODUCTION

Abstract:

" Payroll Management System" is designed to make the existing manual system automatic with the help of computerised equipment and full-edged computer software, fulfilling their requirements, so that their valuable data and information can

be stored for a longer period with easy access and manipulation of the same. The required software is easily available and easy to work with. This web application can maintain and view computerised records without getting redundant entries. The project describes how to manage user data for good performance and provide better services for the client

INTRODUCTION:

The proposed project “Employee Database and Payroll Management System” has been developed to overcome the problems faced in the practice of manual systems. This software is built to eliminate and in some cases reduce the hardships faced by the existing system. Moreover this system is designed for particular needs of the company to carry out its operations in a smooth and effective manner. This web application is reduced as much as possible to avoid errors while entering data. It also provides error messages while entering invalid data. It is user-friendly as no formal knowledge is required to use the system

Purpose

The purpose of this document is to describe the functionality and specifications of the design of a web application for Managing Employees and their payroll. The expected audiences of this document are the developers and the admin of the web application. Now with the help of this system the admin has the information on his finger tips and can easily prepare a good record based on their requirements. Finally, we can say that this system will not only automate the process but save the valuable time of the manager or

the admin, which can be well utilized buy his institute. This will be an additional advantage and management of power based on their free time from his normal duty.

Admin

The Admin gets logged in by valid username and password. Admin can add a new Employee, add a new Department, add a new Pay Grade for the employees. Admin can set the 'from' and 'to' date worked by an employee in a department with specific pay grade. The Admin can generate an automated monthly salary of an employee. The admin can view all the past records of any recorded employee.

ENTITIES LIST:

1.Company

2.Department

3.Employee

4.Pay grade

5.Pay roll

ENTITY COMPANY

Every one company can have N departments

ATTRIBUTES:

PRIMARY KEY (comp_id)

Phone _no

Comp_address

Comp_name

ENTITY DEPARTMENT:

ATTRIBUTES:

Dept_name

dept_id

ENTITY EMPLOYEE:

ATTRIBUTES:

Emp_name

Emp_id

Emp_address

Emp_dob

emp_dateof joining

Emp_phno

ENTITY PAY GRADE:

ATTRIBUTES:

Payg_id

Payg_name

Payg_basic

ENTITY PAYROLL:

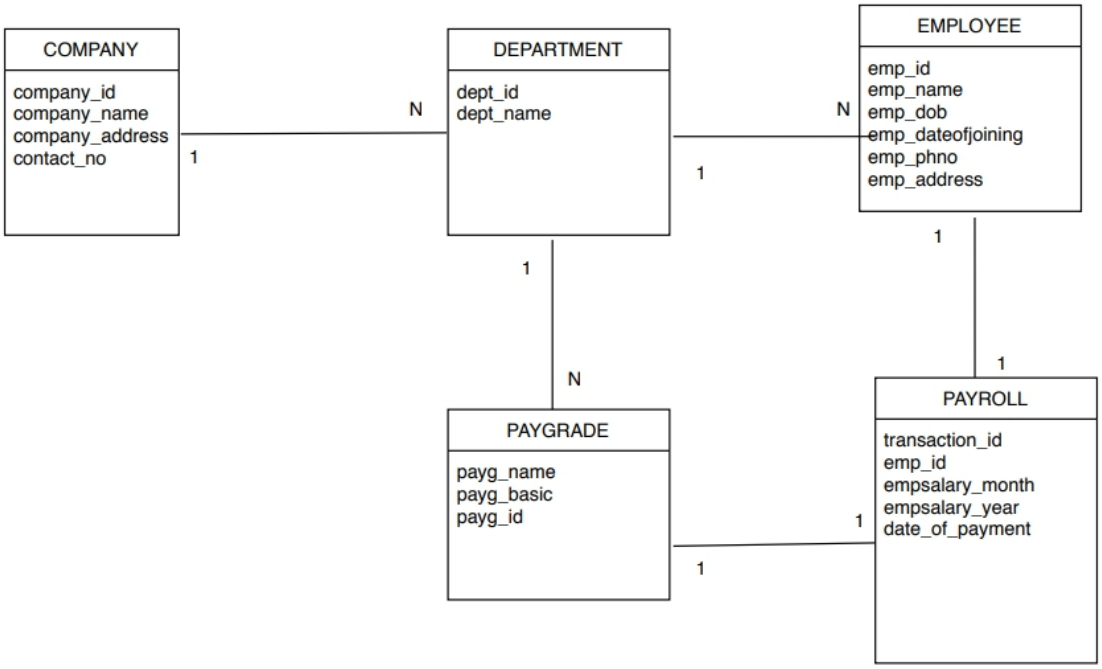
ATTRIBUTES:

Transaction_id

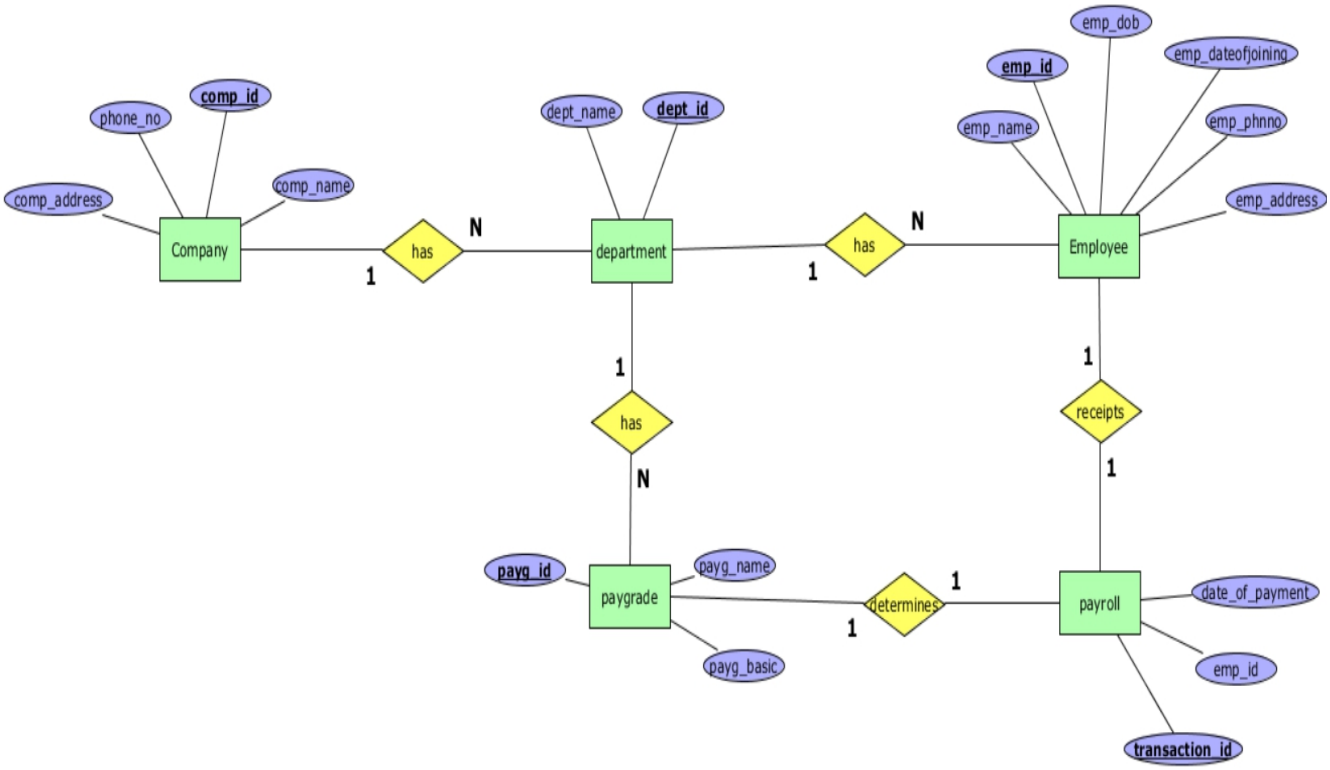
Emp_id

Dateofpayement

SCHEMA



ER-DIAGRAM



CARDINALITY RATIO:

Company to department(one to many)

Department to employee(one to many)

Department to paygrade(one to many)

pay grade to payroll(one to one)

Employee to payroll(one to one)

CONVERTING ER MODEL TO TABLES

Entities:

1.Employee:

Create:

create table employee(emp_id varchar(20),emp_name varchar(20),DOB date,DOJ date,mobile_no number(20), address varchar(50), PRIMARY KEY (emp_id));

INSERT:

insert into employee values('101','Kavya',to_date('2001-10-07','yyyy-mm-dd'),to_date('2010-05-03','yyyy-mm-dd'),9974156280,'Bazaar street,Nellore');
insert into employee values('102','Akhila',to_date('2000-01-10','yyyy-mm-dd'),to_date('2005-11-10','yyyy-mm-dd'),9974156281,'gandhinagar,VIZAG,AP');
insert into employee values('103','Sanjeev',to_date('1999-10-01','yyyy-mm-dd'),to_date('2006-09-06','yyyy-mm-dd'),9974156282,' Near Railway Station,TIRUPATI,AP');
insert into employee values('104','Sathish',to_date('1998-02-12','yyyy-mm-dd'),to_date('2000-02-15','yyyy-mm-dd'),9974156283,'Shivajinagar, block5, VIZAG,AP');

SELECT COMMAND:

SELECT* from employee;

OUTPUT:

EMP_ID	EMP_NAME	DOB	DOJ	MOBILE_NO	ADDRESS

101	Kavya	07-OCT-01	03-MAY-10	9974156280	Bazaar street,Nellore
103	Sanjeev	01-OCT-99	06-SEP-06	9974156282	Near Railway Station,TIRUPATI,AP
104	Sathish	12-FEB-98	15-FEB-00	9974156283	Shivajinagar, block5, VIZAG,AP
102	Akhila	10-JAN-00	10-NOV-05	9974156281	gandhinagar,VIZAG,AP

Download CSV

4 rows selected.

2.PayRoll

Create:

create table payroll(transaction_id varchar(20),emp_id varchar(20),date_of_payment DATE,PRIMARY KEY (transaction_id),foreign key(emp_id) references employee);

INSERT:

insert into payroll1 values('t1','E01',to_date('2000-11-11','yyyy-mm-dd'));

insert into payroll1 values('t2','E02',to_date('1999-07-05','yyyy-mm-dd'));

insert into payroll1 values('t3','E03',to_date('2001-12-12','yyyy-mm-dd'));

insert into payroll1 values('t4','E04',to_date('2000-08-02','yyyy-mm-dd'));

SELECT COMMAND:

SELECT * FROM payroll

OUTPUT:

TRANSACTION_ID	EMP_ID	DATE_OF_PAYMENT
t1	101	11-NOV-00
t2	102	05-JUL-99
t3	103	12-DEC-01
t4	104	02-AUG-00

3.Pay grade

Create:

create table paygrade1(payg_id varchar2(20),payg_name varchar2(20),payg_basic number(20),transaction_id varchar2(20),primary key(transaction_id),foreign

key(transaction_id) references payroll1(transaction_id));

Insert:

```
insert into paygrade1 values('01','g4',2000,'t1');
insert into paygrade1 values('02','g3',5000,'t3');
insert into paygrade1 values('03','g2',6000,'t4');
insert into paygrade1 values('04','g1',7000,'t6');
```

SELECT COMMAND:

```
SELECT* FROM paygrade;
```

OUTPUT:

payg_ID	payg_NAME	payg_BASIC	transaction_ID
01	G1	2000	t1
02	G2	5000	t3
03	G3	6000	t4
04	G4	7000	t6

4 rows selected

4.Department

Create:

```
create table department3(dept_id
varchar2(20),emp_id
varchar2(20),dept_name
varchar2(50),payg_id varchar2(20),primary
key (dept_id),foreign key(emp_id)
references employee1(emp_id),foreign
key(payg_id) references paygrade1);
```

Insert:

```
insert into department3
values('D1','101','Business','01');
insert into department3
values('D2','102','finance','02');
insert into department3
values('D3','103','resource
management','03');
insert into department3
values('D4','104','R&D','04');
```

Select command:

SELECT * FROM department3;

OUTPUT:

DEPT_ID	EMP_ID	DEPT_NAME	GRADE_ID
D1	101	Business	01
D2	102	Finance	02
D3	103	Resource Management	03
D4	104	R&D	04

Download CSV

4 rows selected.

4.Company

CREATE:

create table comp(comp_id varchar2(20),name varchar2(20),phone_no number(10),address varchar2(40),dept_id varchar2(20),primary key(comp_id),foreign key(dept_id) references department3);

INSERT:

insert into comp values('C1','CISCO',8856471239,'GANDHI NAGAR PUNE','D1');

insert into comp values('C2','MRF',6856454239,' SAI NAGAR VIZAG','D2');

insert into comp values('C3','TIZUM',9956471239,'GANDHI NAGAR PUNE','D3');

insert into comp values('C4','TATA',9546471239,' HCL COLONY MUMBAI','D4')

SELECT COMMAND:

SELECT* FROM comp ;

output:

COMPANY_ID	NAME	CONTACT	ADDRESS	DEPT_ID
C1	CISCO	8856471239	GANDHI NAGAR PUNE	D1
C2	MRF	6856454239	SAI NAGAR VIZAG	D2
C3	TIZUM	9956471239	GANDHI NAGAR PUNE	D3
C4	TATA	9546471239	HCL COLONY MUMBAI	D4

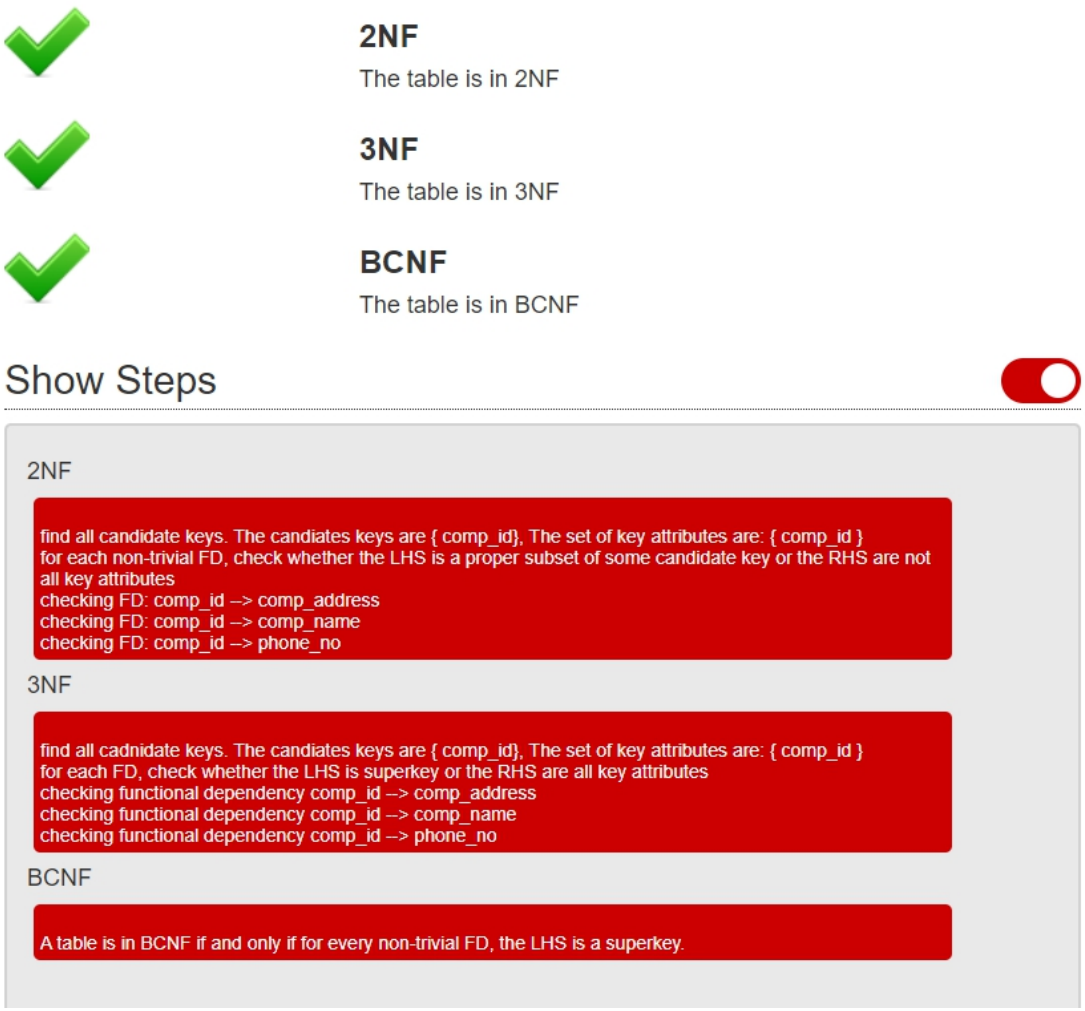
Download CSV

NORMALIZATION:

Company table:

Functional dependencies:

comp_id ->comp_name, comp_address, phone_no



Payroll Table:

Functional dependencies:

Transaction_id -> emp_id, date of payment

2NF

The table is in 2NF

3NF

The table is in 3NF

BCNF

The table is in BCNF

Show Steps

2NF

find all candidate keys. The candiates keys are { transaction_id}, The set of key attributes are: { transaction_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: transaction_id –> emp_id
checking FD: transaction_id –> date_of_payment

3NF

find all cadnidate keys. The candiates keys are { transaction_id}, The set of key attributes are: { transaction_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency transaction_id –> emp_id
checking functional dependency transaction_id –> date_of_payment

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Paygrade Table:

Functional dependencies :

payg_id-> payg_name, payg_basic

Check Normal Form

2NF

The table is in 2NF

3NF

The table is in 3NF

BCNF

The table is in BCNF

Show Steps

2NF

find all candidate keys. The candiates keys are { payg_id}, The set of key attributes are: { payg_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: payg_id --> payg_name
checking FD: payg_id --> payg_basic

3NF

find all cadnidate keys. The candiates keys are { payg_id}, The set of key attributes are: { payg_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency payg_id --> payg_name
checking functional dependency payg_id --> payg_basic

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Department table:

Functional dependencies:

dept_id -> dept_name

2NF

The table is in 2NF

3NF

The table is in 3NF

BCNF

The table is in BCNF

Show Steps

2NF

find all candidate keys. The candiates keys are { dept_id}, The set of key attributes are: { dept_id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: dept_id –> dept_name

3NF

find all cadnidate keys. The candiates keys are { dept_id}, The set of key attributes are: { dept_id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency dept_id –> dept_name

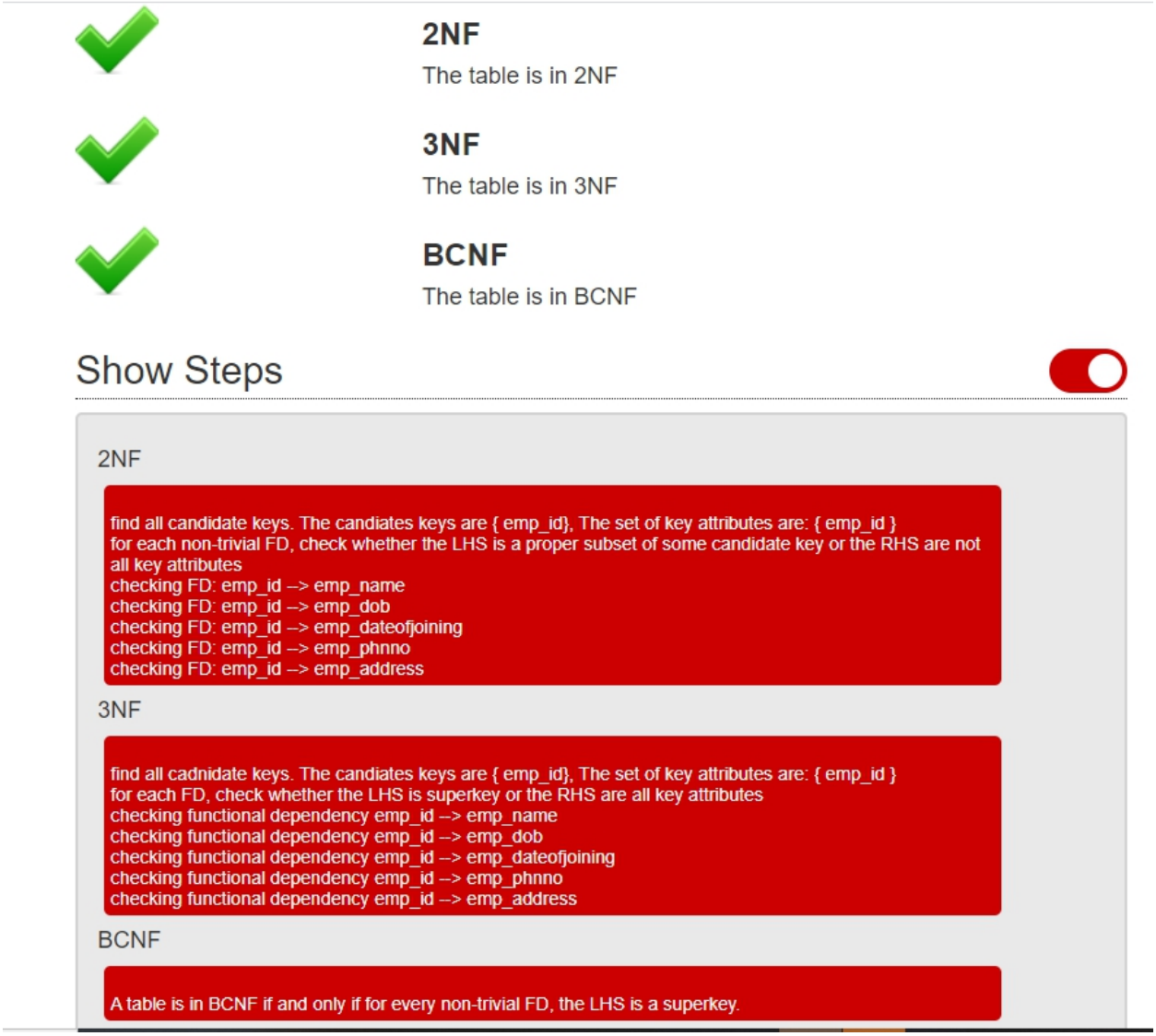
BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Employee table:

Functional dependencies:

Emp_id -> emp_dob, emp_address, emp_name, emp_phnno, emp_dateofjoining



IMPLEMENTATION AND RESULT:

DDL COMMANDS

- **CREATE**
- **ALTER**
- **DROP**
- **TRUNCATE**

ALTER TABLE command to add a New Column

SELECT* from employee;

Alter table employee add city char(12);

OUTPUT:

EMP_ID	EMP_NAME	DOB	DOJ	MOBILE_NO	ADDRESS	AGE
101	Kavya	07-OCT-01	03-MAY-10	9974156280	Bazaar street,Nellore	-
103	Sanjeev	01-OCT-99	06-SEP-06	9974156282	Near Railway Station,TIRUPATI,AP	-
104	Sathish	12-FEB-98	15-FEB-00	9974156283	Shivajinagar, block5, VIZAG,AP	-
102	Akhila	10-JAN-00	10-NOV-05	9974156281	gandhinagar,VIZAG,AP	-

Download CSV

4 rows selected.

Table altered.

ALTER TABLE command to DROP COLUMN :

ALTER TABLE employee DROP COLUMN city;

OUTPUT:

EMP_ID	EMP_NAME	DOB	DOJ	MOBILE_NO	ADDRESS
101	Kavya	07-OCT-01	03-MAY-10	9974156280	Bazaar street,Nellore
102	Akhila	10-JAN-00	10-NOV-05	9974156281	gandhinagar,VIZAG,AP
103	Sanjeev	01-OCT-99	06-SEP-06	9974156282	Near Railway Station,TIRUPATI,AP
104	Sathish	12-FEB-98	15-FEB-00	9974156283	Shivajinagar, block5, VIZAG,AP

TRUNCATE TABLE command:

truncate table employee;

OUTPUT:

Table truncated.

Data Manipulation Language

- **INSERT**
- **UPDATE**
- **DELETE**

Update command:

```
UPDATE  employee

set  emp_name='Akhila varma'

where  emp_id=102;

SELECT emp_id,emp_name from employee;
```

OUTPUT:

1 row(s) updated.

EMP_ID	EMP_NAME
101	Kavya
102	Akhila varma
103	Sanjeev

104	Sathish

DELETE COMMAND

```
DELETE FROM employee

WHERE emp_name ='Kavya';

SELECT emp_id,emp_name from employee;
```

OUTPUT:

0 row(s) deleted.

EMP_ID	EMP_NAME
102	Akhila varma
103	Sanjeev
104	Sathish

Download CSV

3 rows selected.

SELECT COMMANDS

SQL SELECT DISTINCT Statement

```
select distinct address
```

from employee;

OUTPUT

ADDRESS
Bazaar street,Nellore
Near Railway Station,TIRUPATI,AP
gandhinagar,VIZAG,AP
Shivajinagar, block5, VIZAG,AP

Download CSV

4 rows selected.

Where clause using sql

SELECT * FROM employee

WHERE emp_name='Akhila';

Output:

EMP_ID	EMP_NAME	DOB	DOJ	MOBILE_NO	ADDRESS
102	Akhila	10-JAN-00	10-NOV-05	9974156281	gandhinagar,VIZAG,AP

Download CSV

Queries:

1. Write an sql command to find the employee name starting with k

select*

from employee

where emp_name like 'K%'

OUTPUT:

EMP_ID	EMP_NAME	DOB	DOJ	MOBILE_NO	ADDRESS
101	Kavya	07-OCT-01	03-MAY-10	9974156280	Bazaar street,Nellore

Download CSV

2.Write a sql command to find the employee name and id in order by clause

SELECT * FROM employee

ORDER BY emp_name,emp_id;

OUTPUT:

EMP_ID	EMP_NAME	DOB	DOJ	MOBILE_NO	ADDRESS
102	Akhila	10-JAN-00	10-NOV-05	9974156281	gandhinagar,VIZAG,AP
101	Kavya	07-OCT-01	03-MAY-10	9974156280	Bazaar street,Nellore
103	Sanjeev	01-OCT-99	06-SEP-06	9974156282	Near Railway Station,TIRUPATI,AP

104	Sathish	12-FEB-98	15-FEB-00	9974156283	Shivajinagar, block5, VIZAG,AP

Download CSV

4 rows selected

3. Write an sql command to find min of emp_id in employee table

```
select min(emp_id)
from employee;
```

Output:

MIN(EMP_ID)
101

Download CSV

4.write an sql command between emp_id in employee table

```
SELECT emp_id
FROM employee
WHERE emp_id BETWEEN 101 AND 104;
```

Output:

EMP_ID
101
102
103

104

Download CSV

4 rows selected.

5.write an sql command using in operator

SELECT * FROM employee

WHERE address IN ('gandhinagar,VIZAG,AP', 'Near Railway Station,TIRUPATI,AP');

output:

EMP_ID	EMP_NAME	DOB	DOJ	MOBILE_NO	ADDRESS
102	Akhila	10-JAN-00	10-NOV-05	9974156281	gandhinagar,VIZAG,AP

Download CSV

Conclusion:

This project is built keeping in mind that it is to be used by only one user that is the admin. It is built for use in small scale organizations where the number of employees is limited. According to the requested requirement the admin can add, manipulate, update and delete all employee data in his organization. The admin can add new departments and delete them. The

Admin can also add predefined pay grades for the employees. The required records can be easily viewed by the admin anytime time he wants in an instant. The payment of the employee is based on monthly basis. Numerous validations implemented would enable the admin to enter accurate data. The main objective of this framework is to save time, make the system cost effective and management records efficiently.

THANK YOU