Normalization and Schema Refinement

ASSUMPTION:-All contact numbers are unique.

Relations	Schema
Employee	Emp_id(pk),Emp_name,Contact,email_id
Hourly_employee	Emp_ID(pk), Emp_Name, Contact Hourly_Wage, Hours_worked,EMAIL_ID
Intern	Emp_ID (pk),Emp_Name, Contact,EMAIL_ID ,Stipend
FullTime_employee	Emp_ID(pk) ,Emp_Name, Contact, Experience, Salary,email_id.
Department	dept_id,manager_name,dept_budget
NON-HR Department	Dept_ID(pk) ,Manager_Name, Dept_Budget, Dept_Name
HR_Department	Dept_ID(pk) ,Manager_Name, Dept_Budget, Num_HR's
Candidate	Name ,DOB ,Cand_ID(pk) ,EMAIL_ID, Interview_Status, Dept_Applied.
Company	Name(pk) ,grade,PINCODE,CITY,STATE
Location	PINCODE(pk),CITY,STATE Employee_Capacity
Works_in	Emp_id(fk),dept_id(fk),DEPT_ID,

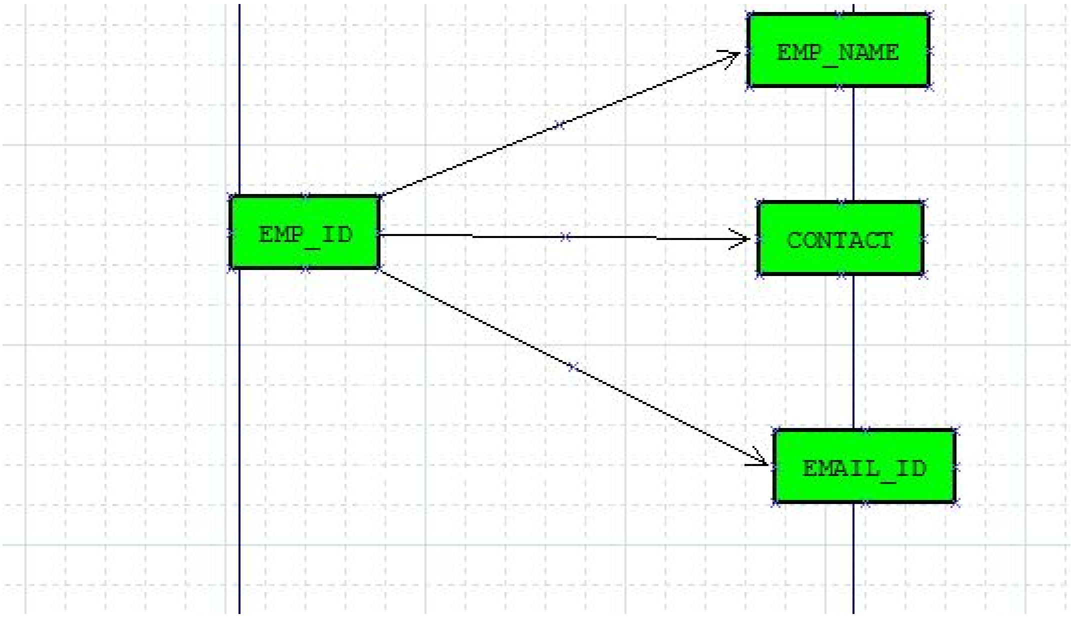
	EMP_ID,PINCODE(fk)
Monitors	DEPT_ID,C_NAME(PK,FK)
Applies_to	C_NAME,C_ID(PK),C_NAME(FK),CAND_ID(FK)
Interviews	DEPT_ID,CAND_ID,EMP_ID(PK,FK)
Manages	DEPT_ID(pk,fk),EMP_ID(fk)

ATTRIBUTE: Num_HR's==Num_HRS

Relation Employee:-

Emp_id	Emp_name	email_id	Contact

FDD->



CREATE TABLE EMPLOYEE
(

```
     EMP_ID INTEGER,
    EMP_NAME,
   CONTACT[10],
   EMAIL_ID CHAR[20]
PRIMARY KEY (EMP_ID)
)
```

EXPLANATION:-

pk dependencies:-
emp_id->emp_name,contact,emial_id

candidate key:-
(emp_id,emp_name)

REDUNDANCY:- There is no redundancy.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

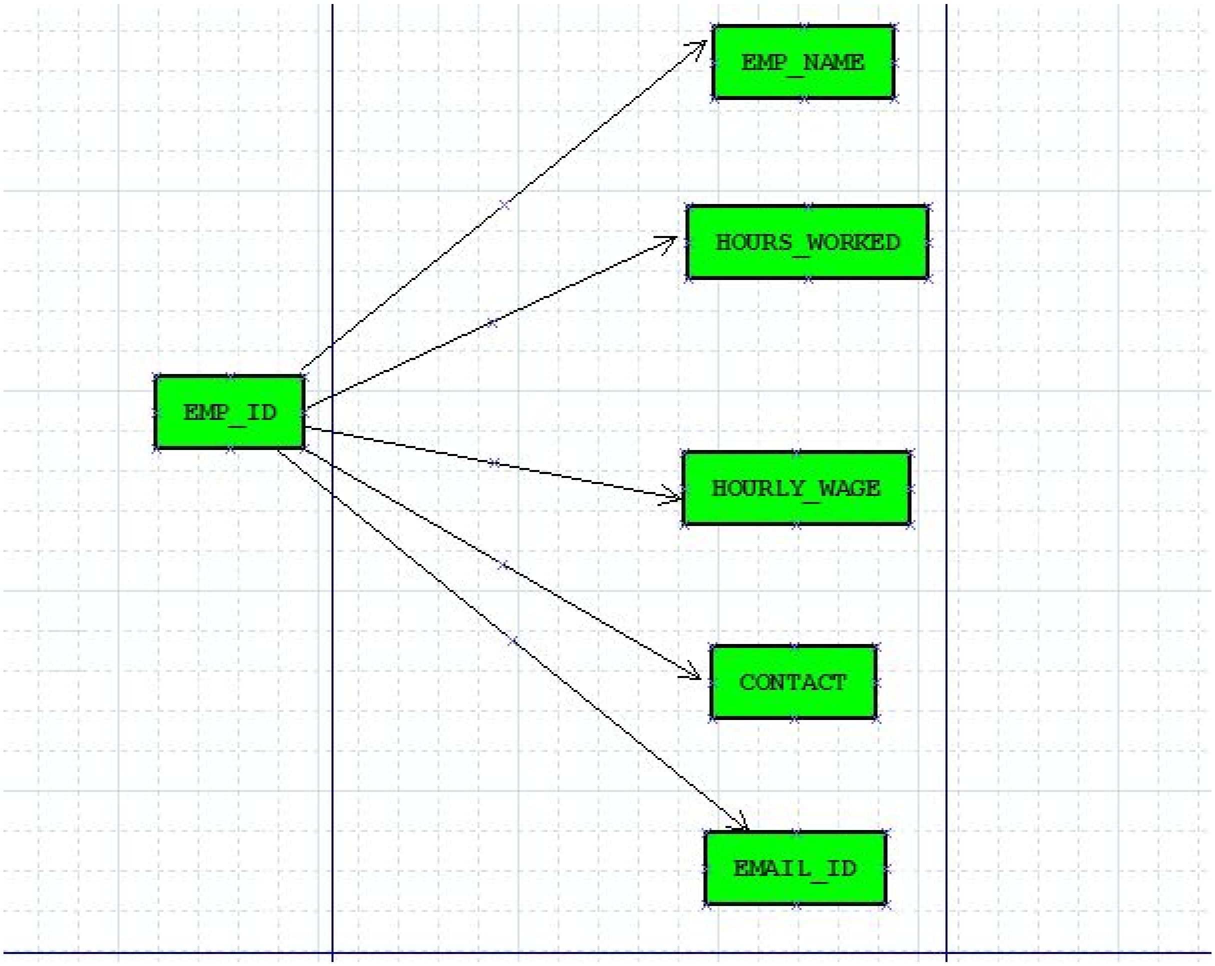
NORMALIZATION

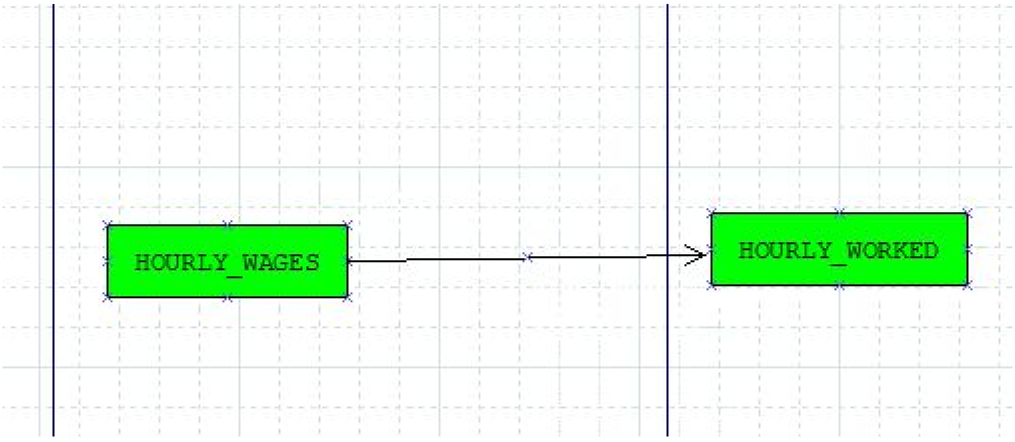
All the attributes here are atomic. So it is in 1NF form.
There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form
There is no transitive FD. So, it is in 3NF form.

Relation:HOURLY_EMPLOYEE

Emp_ID	Emp_name	contact	Hourly_Wage	Hours_worked	email_id

FDD->





Before decomposition:-
CREATE TABLE HOURLY EMPLOYEES
(EMP_ID INTEGER,

```
 EMP_NAME CHAR(15),
 CONTACT INTEGER,
  Email_id CHAR(20),
HOURS_WORKED INTEGER,
HOURLY_WAGE INTEGER,
 PRIMARY KEY(EMP_ID) )
```

EXPLANATION:-

pk dependencies:-
Emp_id->emp_name,contact,Hourly_wage,Hours_worked,email_id.

CANDIDATE KEY:-
(EMP_ID,EMP_NAME)

REDUNDANCY:Redundancy is repetition of data,Here we store information about each Hourly_wages. For each Hours_worked there is particular Hourly_wage assigned to it.So where ever there is hours_worked there will be repetetion of this information . So here there is redundancy.

UPDATE, DELETE and INSERT anomalies :- If the Hourly_wages is updated, we must correspondingly update the Hours_worked.We can't insert a Hourly_wages tuple unless we know the Hours_Worked. If we delete the last tuple corresponding to a particular Hourly_wages, we lose information regarding the Hours_worked.Hence, we have update, insert and deletion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is   no transitive FD. So, it is in 3NF form.

Decomposition into two tables:-

- **HOURLY_EMPLOYEE**

```
         (
          EMP_ID INTEGER NOT NULL,
        EMP_NAME CHAR(15),
       CONTACT INTEGER,
         EMAIL_ID CHAR(20),
          HOURLY_WAGE INTEGER,
         PRIMARY KEY (EMP_ID)
         FOREIGN KEY HOURLY_WAGE REFERENCES Hourly_wages


          )
```

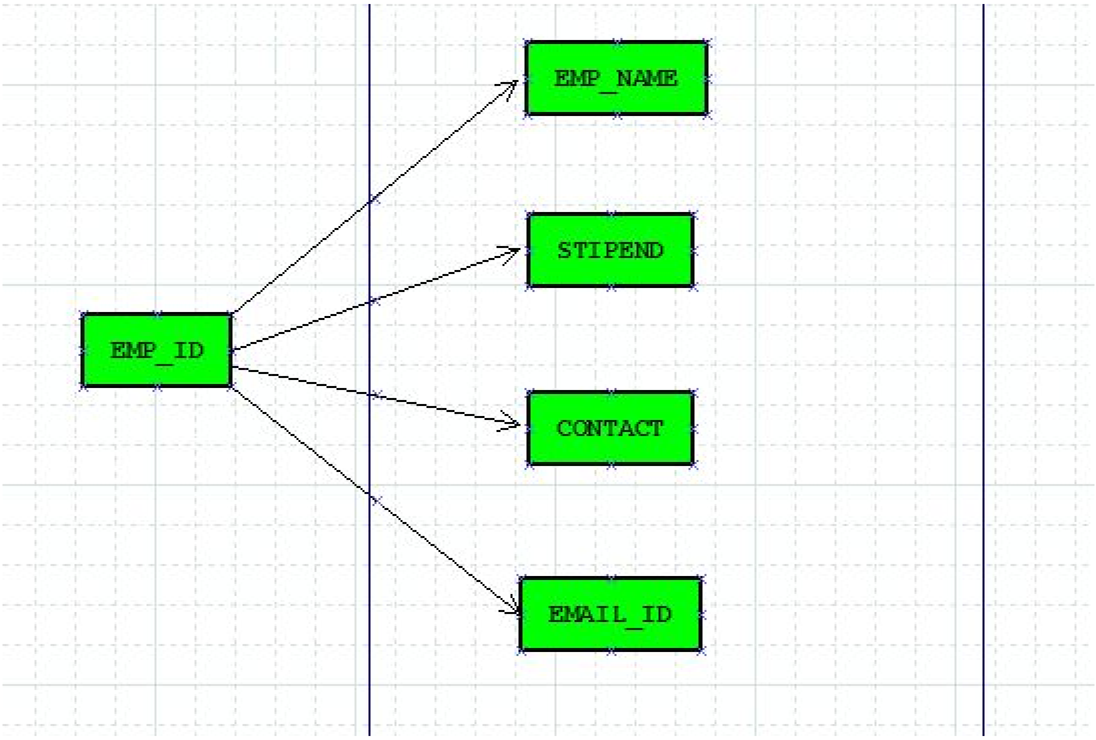
- **HOURLY_WAGE**

```
         (
            HOURS_WORKED INTEGER,
           HOURLY_WAGES INTEGER,
          PRIMARY KEY HOURLY_WAGES,
       )
```

Relation:Intern

Emp_id	Emp_name	email_id	contact	stipend

FDD->



CREATE TABLE FOR INTERN:
(STIPEND FLOAT,
EMP_ID INTEGER,
EMAIL_ID CHAR(15)**,**
EMP_NAME CHAR(15),
CONTACT INTEGER,
PRIMARY KEY(EMP_ID))

EXPLANATION:-
functional dependencies:-
Emp_ID (pk)->Emp_Name, Contact,stipend,email_id.
Emp_name->contact,stipend,email_id

pk dependencies:-
Emp_ID (pk)->Emp_Name, Contact,stipend,email_id.

REDUNDANCY:Redundancy is repetition of data,Here we store information about each Intern. Each Intern has a unique address, contact and employee id. So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

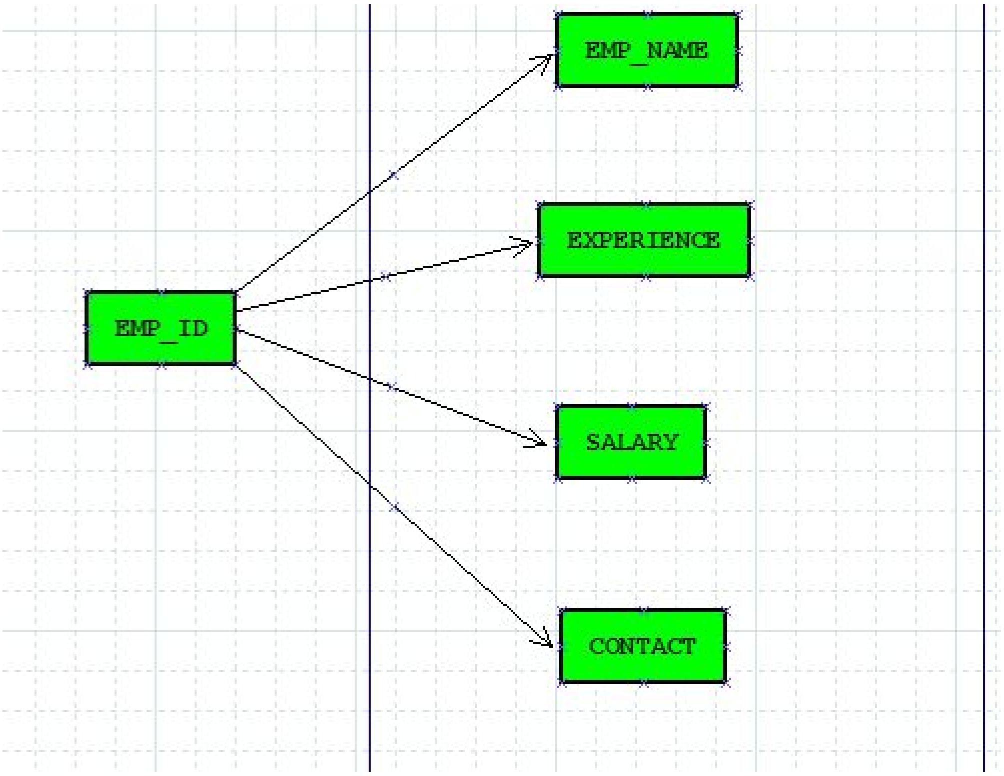
There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is no transitive FD. So, it is in 3NF form.

Relation:FullTime_employee

Emp_id	Emp_name	contact	Experience	salary	email_id

FDD->



CREATE TABLE FULL TIME EMPLOYEE:
(EMP_NAME CHAR(20).
EXPERIENCE CHAR(30),
SALARY FLOAT,
EMP_ID INTEGER,
EMAIL_ID CHAR(15),
CONTACT INTEGER,
PRIMARY KEY(EMP_ID))

EXPLANATION:-
functional dependencies:-
Emp_ID(pk) ->Emp_Name, Contact, Experience,email_id
Emp_name->contact,email_id,experience

pk dependencies:-
Emp_ID(pk) ->Emp_Name, Contact, Experience,ad

REDUNDANCY:Redundancy is repetition of data,Here we store information about each FullTime_Employee. Each Employee has a unique address, contact and employee id. So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

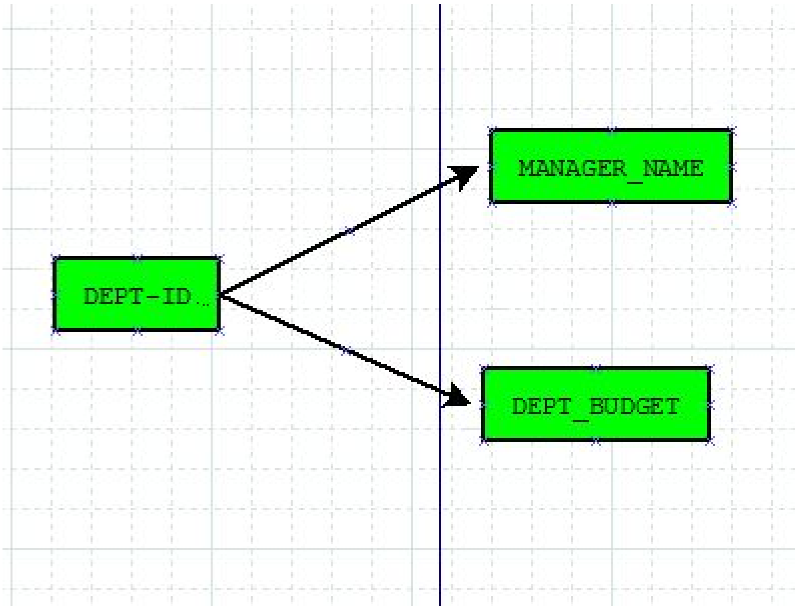
There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is no transitive FD. So, it is in 3NF form.

Relation:Department

DEPT_ID	MANAGER_NAME	DEPT_BUDGET

FDD->



```
CREATE TABLE DEPARTMENT:
( DEPT_ID INTEGER,
MANAGER_NAME CHAR(20),
DEPT_BUDGET INTEGER,
PRIMARY KEY(DEPT_ID)
)
```

EXPLANATION:-

PK DEPENDENCIES:-
dept_id->manager_name,dept_budget.

REDUNDANCY:Redundancy is repetition of data,Here we store information about each DEPARTMENT. Each DEPARTMENT has a unique DEPT_ID. So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

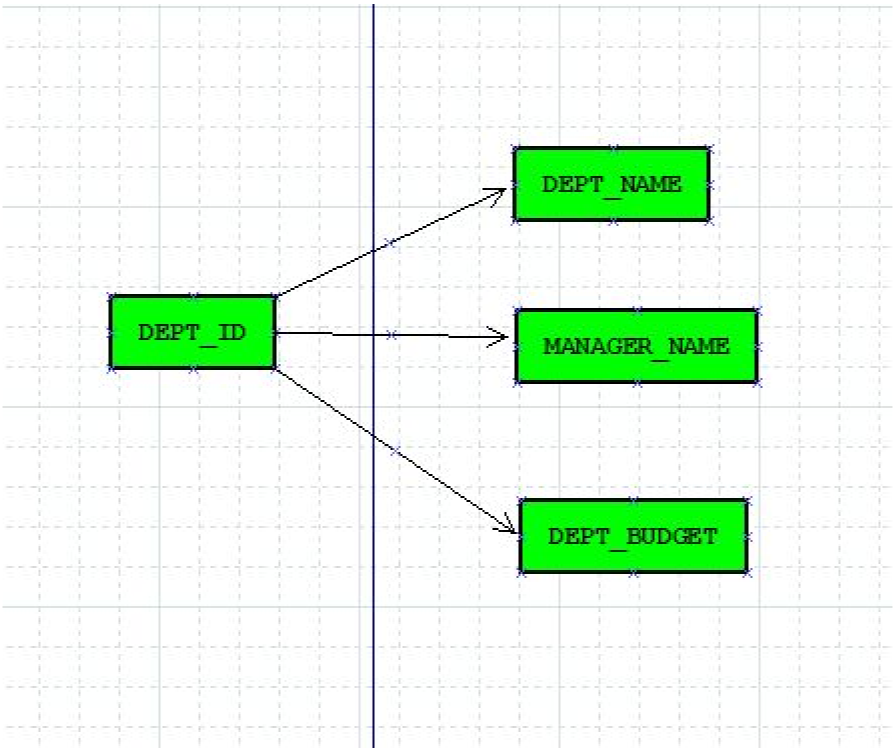
There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is no transitive FD. So, it is in 3NF form.

Relation:NON HR department

Dept_id	Manager_name	Dept_budget	Dept_name

FDD->



CREATE TABLE NON HR- DEPARTMENT:
(DEPT_ID INTEGER,
MANAGER_NAME CHAR(20),
DEPT_NAME CHAR(20),
DEPT_BUDGET INTEGER,
PRIMARY KEY(DEPT_ID))

EXPLANATION:-

pk dependencies:-
dept_id->manger_name,Dept_budget,address,Dept_name.

REDUNDANCY:Redundancy is repetition of data,Here we store information about each Non HR-Department. Each Department has a unique address, Department-name , Department-id and different Department-budget . So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

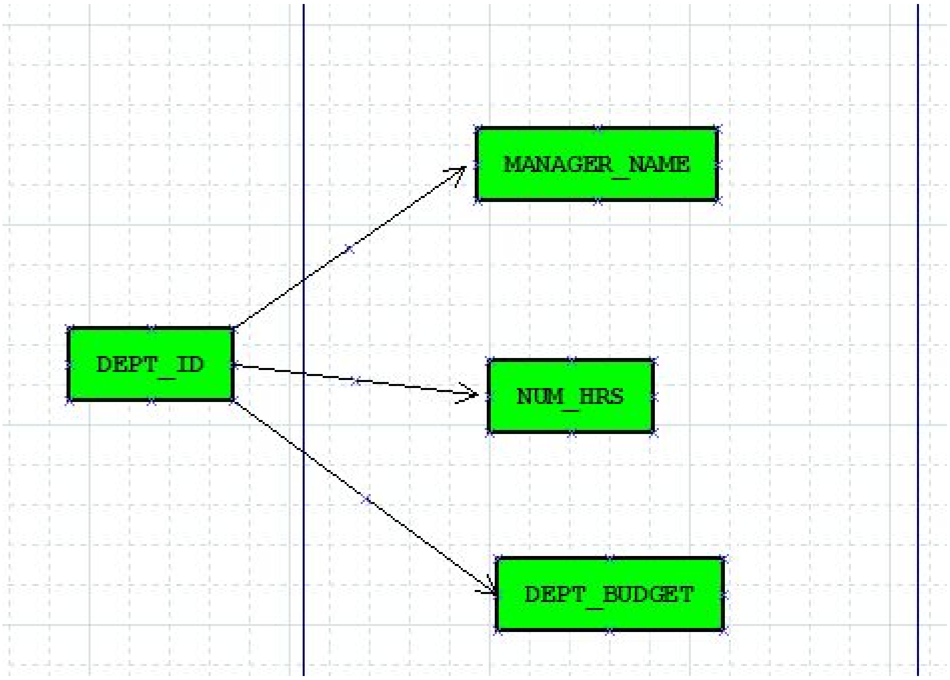
NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.
There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form
There is no transitive FD. So, it is in 3NF form.

Relation:HR department

Dept_ID(pk)	Manager_Name	Dept_Budget	Num_HR's

FDD->



CREATE TABLE HR- DEPARTMENT:
(DEPT_ID INTEGER,
NUM_HR'S INTEGER,
MANAGER_NAME CHAR(20),
DEPT_BUDGET INTEGER,
PRIMARY KEY(DEPT_ID))

EXPLANATION:-
pk dependencies:-
Dept_id->manger_name,Dept_budget,address,Num_Hr’s

REDUNDANCY:Redundancy is repetition of data,Here we store information about each HR-Department. Each Department has a unique address, contact and Department- id. So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies
There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

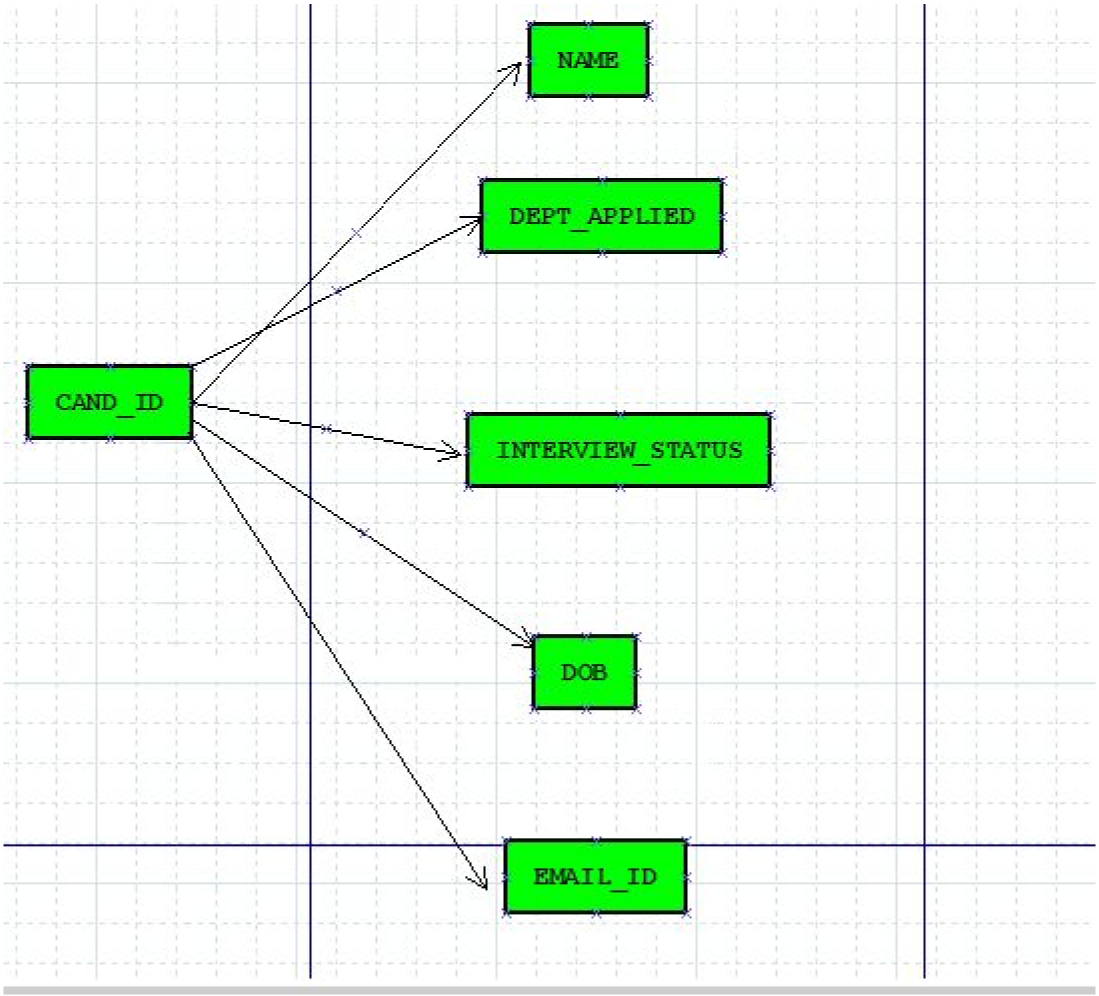
There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is no transitive FD. So, it is in 3NF form.

Relation:Candidate

Name	DOB	Cand_id	email_id	Interview_status	Dept_applied

FDD->



ASSUMING THERE CAN BE MULTIPLE CANDIDATES WITH SAME NAME

```
CREATE TABLE CANDIDATE:
(CAND_ID INTEGER,
CAND_NAME CHAR(20),
DOB INTEGER,
INTERVIEW_STATUS CHAR(10),
EMAIL_ID CHAR(20),
DEPT_APPLIED CHAR(20),
PRIMARY KEY(CAND_ID) )
```

EXPLANATION:-

functional dependencies:-

Cand_ID(pk)->Name ,DOB ,email_id, Interview_Status, Dept_Applied.

Cand_id,name->DOB,email_id,interview_status,Dept_applied.

pk dependencies:-
Cand_ID(pk)->Name ,DOB ,email_id, Interview_Status, Dept_Applied.

REDUNDANCY:Redundancy is repetition of data,Here we store information about each Candidate. Each Candidate has a unique email_id, contact and candidate- id. So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

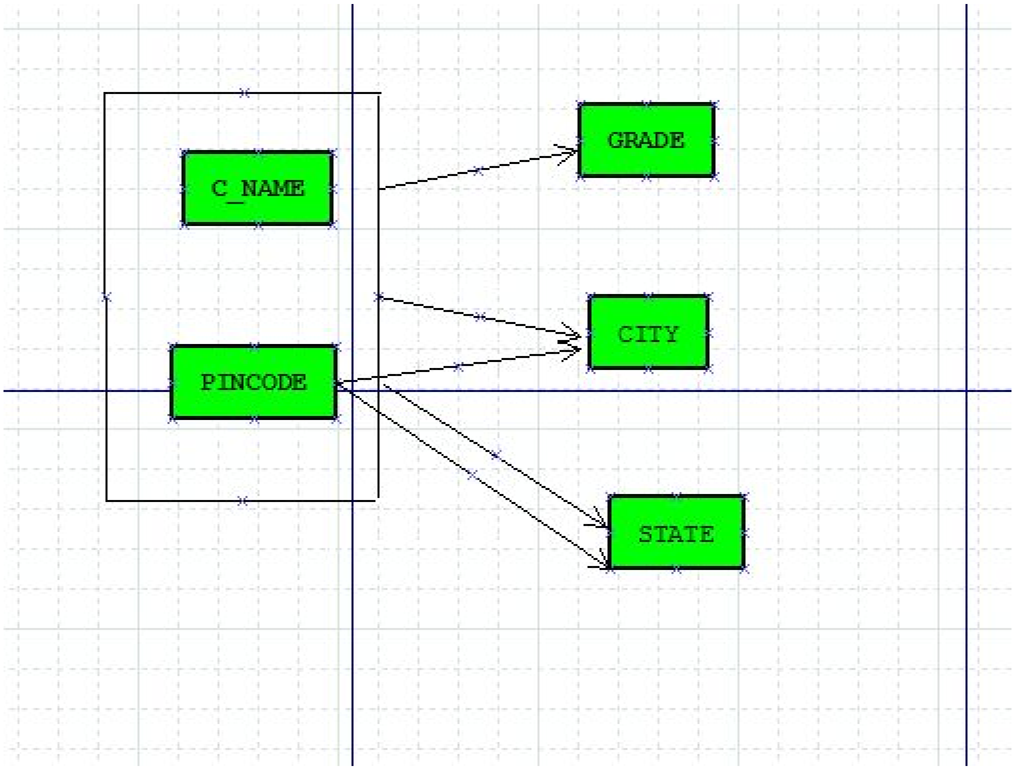
There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is no transitive FD. So, it is in 3NF form.

Relation:Company

c_Name	**grade**	**pincode**	**city**	**state**

FDD->



```
CREATE TABLE COMPANY
(C_NAME CHAR(20),
C_GRADE CHAR(20),
PINCODE INTEGER,
CITY CHAR(20),
STATE CHAR(20),
PRIMARY KEY(C_NAME,PINCODE))
```

EXPLANATION:-

PRIMARY KEY dependencies:-
C_NAME,PINCODE->GRADE,CITY,STATE.

Partial key dependency:-
pincode->city,state.

REDUNDANCY:There is redundancy in data here. A single company can have many branches in same state.So c_name and state repeat.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

There is partial key dependencies.City and state can be determined from pincode itself.So there is partial key dependency.

There is no transitive FD. So, it is in 3NF form.

Decomposition into two tables (through 2NF):-

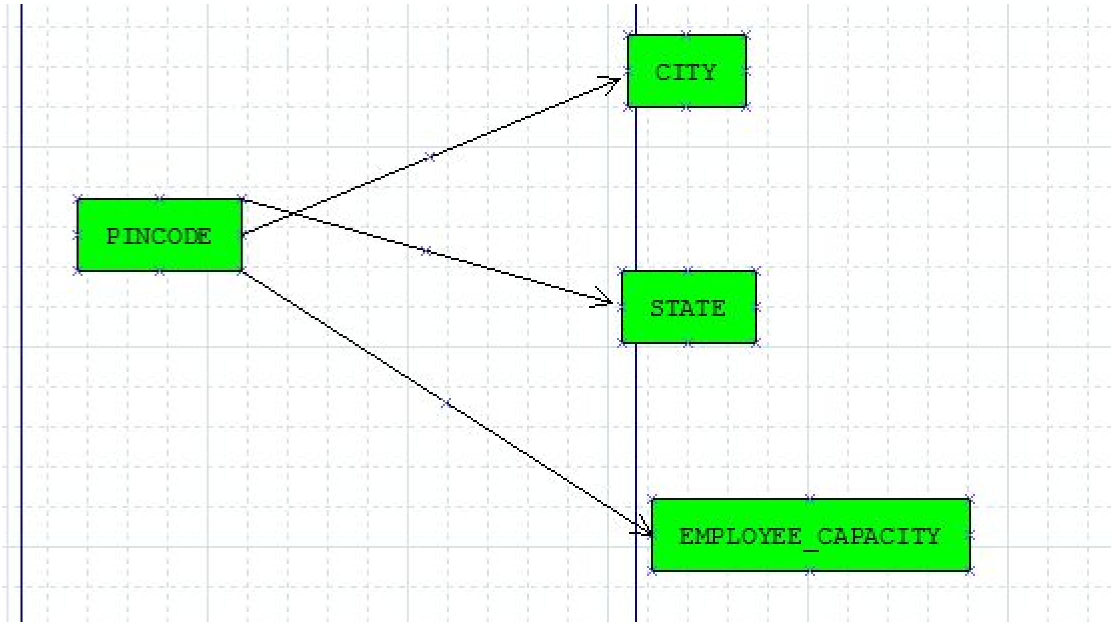
```
CREATE TABLE COMPANY
( C_NAME CHAR(20),
C_GRADE CHAR(20),
PINCODE INTEGER
PRIMARY KEY (C_NAME,PINODE)

)
```

```
CREATE TABLE ADDRESS
(PINCODE INTEGER,
CITY CHAR(20),
STATE CHAR(20),
FOREIGN KEY PINCODE REFERENCE COMPANY
)
```

Relation:location

Pincode	city	State	Employee_Capacity

FDD->



CREATE TABLE LOCATION
PINCODE INTEGER,
CITY CHAR(20),
STATE CHAR(20),
EMPLOYEE_CAPACITY INTEGER,
PRIMARY KEY(PINCODE))

EXPLANATION:-
Primary key dependency:-
Pincode(pk)->Employee_Capacity

REDUNDANCY:Redundancy is repetition of data,Here we store information about each Location. Each Location has a unique pincode and employee_Capacity. So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies
There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is no transitive FD. So, it is in 3NF form.

Nothing to decompose.

Relation:Works_in

Emp_id	Dept_id	PINCODE

```
CREATE TABLE WORKS_IN
( DEPT_ID INTEGER,
EMP_ID INTEGER,
PINCODE INTEGER,
PRIMARY KEY(DEPT_ID,EMP_ID,PINCODE),
FOREIGN KEY (EMP_ID) REFERENCES TO EMPLOYEE,
FOREIGN KEY (DEPT_ID) REFERENCES TO DEPARTMENT,
FOREIGN KEY(PINCODE) REFERENCES TO LOCATION,
ON DELETE CASCADE)
```

EXPLANATION:-

functional dependencies:-

Emp_id,dept_id->Pincode.

REDUNDANCY:Redundancy is repetition of data,Here we store information about each Employee,Department. Each Employee and Department has a unique pincode, contact and id. So here there is no redundancy.

UPDATE, DELETE and INSERT anomalies

There are no update, delete and insertion anomalies.

NORMALIZATION

All the attributes here are atomic. So it is in 1NF form.

There are no partial key dependencies(all candidate keys have only one attribute). So, it is in 2NF form

There is no transitive FD. So, it is in 3NF form.

Relation Manages

DEPT_ID	EMP_ID

CREATE TABLE MANAGES

```
( DEPT_ID INTEGER,
EMP_ID INTEGER,
PRIMARY KEY(DEPT_ID),
FOREIGN KEY (EMP_ID) REFERENCES  EMPLOYEE,
FOREIGN KEY(DEPT_ID) REFERENCES  DEPARTMENT
)
```

Relation Monitors

DEPT_ID	C_NAME

```
CREATE TABLE MONITORS
(DEPT_ID INTEGER,
C_NAME CHAR(20),
PRIMARY KEY(DEPT_ID,C_NAME),
FOREIGN KEY(C_NAME) REFERENCES COMPANY,
FOREIGN KEY(DEPT_ID) REFERENCES DEPARTMENT
)
```

Relation Interviews:-

DEPT_ID	CAND_ID	EMP_ID

```
CREATE TABLE INTERVIEWS
(DEPT_ID INTEGER,
CAND_ID INTEGER,
EMP_ID INTEGER,
PRIMARY KEY(DEPT_ID,CAND_ID,EMP_ID),
FOREIGN KEY(DEPT_ID) REFERENCES  DEPARTMENT,
FOREIGN KEY(CAND_ID) REFERENCES CANDIDATE,
FOREIGN KEY(EMP_ID) REFERENCES EMPLOYEE
)
```

Relation Applies_To

C_NAME	CAND_ID

```
CREATE TABLE APPLIES_TO
(C_NAME CHAR(20),
CAND_ID INTEGER,
PRIMARY KEY(C_NAME,CAND_ID),
FOREIGN KEY(C_NAME) REFERENCES  COMPANY,
FOREIGN KEY(CAND_ID) REFERENCES  CANDIDATE)
```