Introduction

The system we are working on is basically the HR management system. The main entities that are part of the HR management system are as follows:

* HR Manager
* Employee
* Clients
* Performance reviews
* Salaries
* Increments
* Deduction

### Dataset:

We have not used any kind of dataset available online. We gave priority to make our own database of the System mentioned above and we were guided several times by our Professor Dr. Abid throughout the project.

### How the System will Work?

The system is basically the HR management system in which there are thousands of employees work at the same time. We will generate the data of almost 10 years. Each of the main entities mentioned above has its own importance and plays a significant role in the whole system. Following are the entities along with their some of the main functionalities.

### HR Manager:

HR manager is also an employee. HR manager will have all the attributes like a normal employee. The Salary of the HR Manager is mentioned in the Job Titles as there are total of 139 job titles for different employees. An HR manager can see the comments and reviews given by the clients and give his own comments too.

### Employee:

An employee has different attributes like name (can be first name and last name together), address, basic salary, job titles etc. Employee is of some type defined in the “Employee Type “column. Employee has a working status that can be either 0 or 1 and this will reflect whether the employee is still working or left the job. Employee has a salary and after each job assignment, employees are given reviews by the clients and on the basis of these reviews, employee gets an increment if it meets the requirement.

### Performance Reviews:

After every job assignment, employee is given a review and these reviews are annually calculated according to a decided formula. If the review’s ratio is 50% or above, that specific employee will get an increment in one’s salary.

### Clients:

Client is a simple entity having some common attributes like name, address, phone number, email etc. Client is capable of giving at most one rating and one review per job assignment.

### Salary:

Salary is one of the most important entity of the whole system as it contains the numeric data that can be aggregated. So, this will be our first and main transactional entity. Salary is given to employees on monthly basis employees and there is a possibility of salary increment after exactly one year if the rating’s ratio of a particular employee meets the target which is at least 50%.

### Deduction:

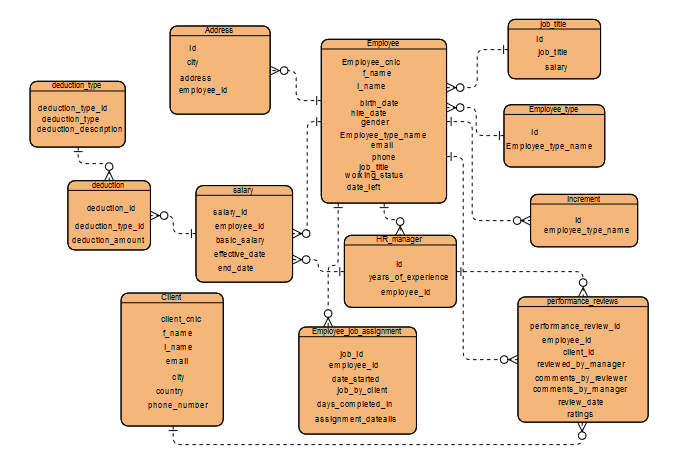
Two types of tax deductions are set. One is of 10% of the basic salary and second one is of 2% of the basic salary. Both of the deductions are on monthly basis. Salary comes from salary table and then deduction is applied.

### Increments:

Increment table calculates the increment amount of a specific employee that is working for one year. The criteria for incrementing in a salary is simple. If the annual ratings of the employee have a ratio of 50% or above, employee will have an increment in the basic salary. Below is the criteria table according to which increments will be calculated.

|  |  |
| --- | --- |
| **Percentage of Ratings Calculated Annually** | **Increment in the Basic Salary (Percentage)** |
| 50-59% | 5-5.9% |
| 60-69% | 6-6.9% |
| 70-79% | 7-7.9% |
| 80-89% | 8-8.9% |
| 90-100% | 9-10% |

ERD (Entity Relation Diagram)



# Employee\_Type

create table employee\_type(

id int not null identity primary key,

employee\_type\_name varchar(15);

# Employee\_Job\_Assignment

create table employee\_job\_assignment(

job\_id int not null identity primary key,

employee\_id varchar(25) foreign key references employee(employee\_cnic),

date\_started date,

days\_completed\_in int,

assignment\_details varchar(300),

job\_by\_client varchar(25) foreign key references client(client\_cnic)

);

insert into employee\_job\_assignment (employee\_id,date\_started,assignment\_details,job\_by\_client,days\_completed\_in) select employee\_id, review\_date,comments\_by\_reviewer,client\_id,RAND()\*(90-30)+30 from performance\_reviews ;



# Employee

create table employee (

employee\_cnic varchar(25) not null primary key,

f\_name varchar(10),

l\_name varchar(10),

birth\_date date,

hire\_date date,

gendre varchar(6),

employee\_type\_name int foreign key references employee\_type(id)

);



alter table employee add job\_title int;

alter table employee add foreign key (job\_title) references job\_title(id);

GO

CREATE PROCEDURE selectEmployee @employee\_cnic nvarchar(30)

AS

SELECT \*

FROM employee

WHERE employee\_cnic = @employee\_cnic

GO

exec selectEmployee @employee\_cnic='00009-6482451-2';

alter table employee alter column f\_name varchar(25);

alter table employee alter column l\_name varchar(25);

alter table employee drop column job\_title;

alter table employee add email varchar(45);

alter table employee add phone varchar(20);



# HR\_Manager

create table Hr\_Manager (

id int not null identity primary key,

years\_of\_experience float(6),

employee\_id varchar(25) foreign key references employee(employee\_cnic)

);

alter table Hr\_Manager drop column job\_titles;

select \* from hr\_manager;

alter table Hr\_manager add Job\_titles varchar(30);

alter table employee add Job\_titles varchar(30);

insert into Hr\_Manager (employee\_id) select employee\_cnic from employee where job\_title=83;

# Performance\_Reviews

create table performance\_reviews(

performance\_review\_id int not null identity primary key,

employee\_id varchar(25) foreign key references employee(employee\_cnic),

client\_id varchar(25) foreign key references client(client\_cnic),

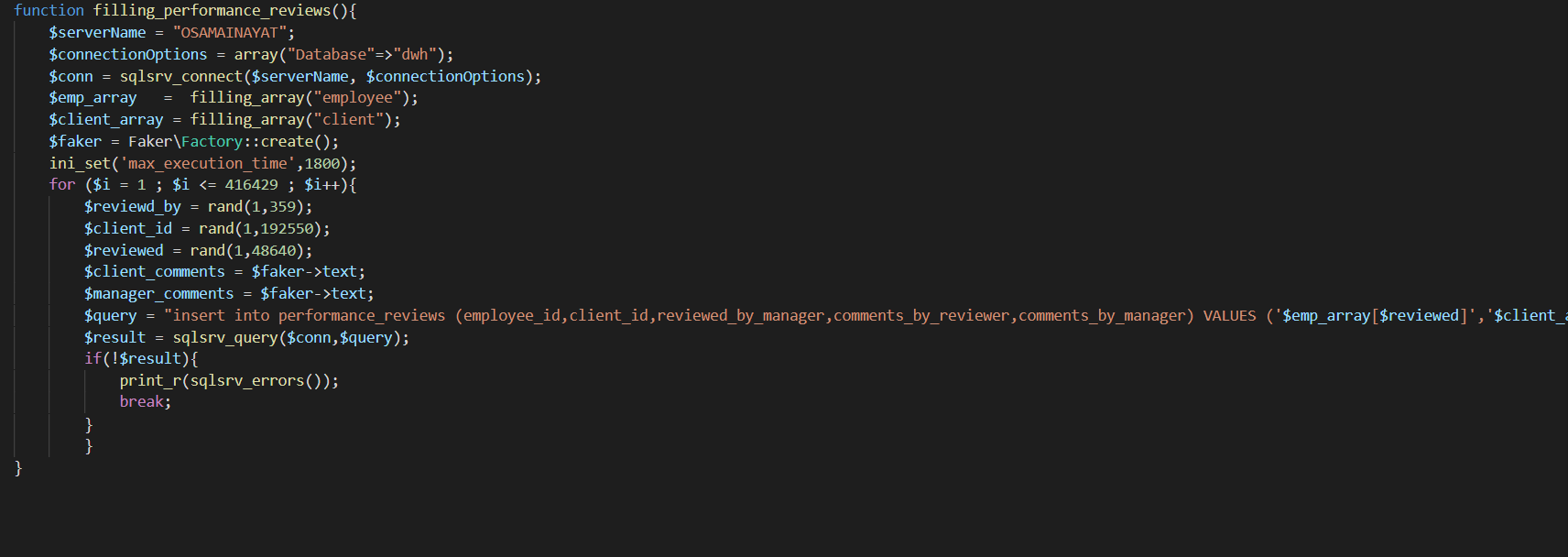
reviewed\_by\_manager int foreign key references hr\_manager(id),

comments\_by\_reviewer varchar(300),

comments\_by\_manager varchar(300)

);

alter table performance\_reviews add review\_date date;



# Deduction\_Type

create table deduction\_type(

deduction\_type\_id int not null identity primary key,

deduction\_type varchar(50),

deduction\_description varchar(200)

);

insert into deduction\_type (deduction\_type,deduction\_description) values('General TAX','MONTHLY INCOME TAX');

insert into deduction\_type (deduction\_type,deduction\_description) values('GP TAX','GP TAX');

# Deduction

create table deduction(

deduction\_id int not null primary key identity,

deduction\_type\_id int foreign key references deduction\_type(deduction\_type\_id),

deduction\_amount float,

salary\_id int foreign key references salary(salary\_id)

);

insert into deduction(salary\_id,deduction\_amount) select salary\_id,basic\_salary from salary;

select \* from salary;

SELECT \* FROM increments;

select \* from deduction\_type;

select \* from deduction;

update deduction set deduction\_type\_id=2 where deduction\_type\_id IS NULL ;

select \* from deduction

update deduction set deduction\_amount = (0.01\*deduction\_amount) where deduction\_type\_id = 2;

# Client

create table client(

client\_cnic varchar(25) primary key,

f\_name varchar(15) not null,

l\_name varchar(15) not null,

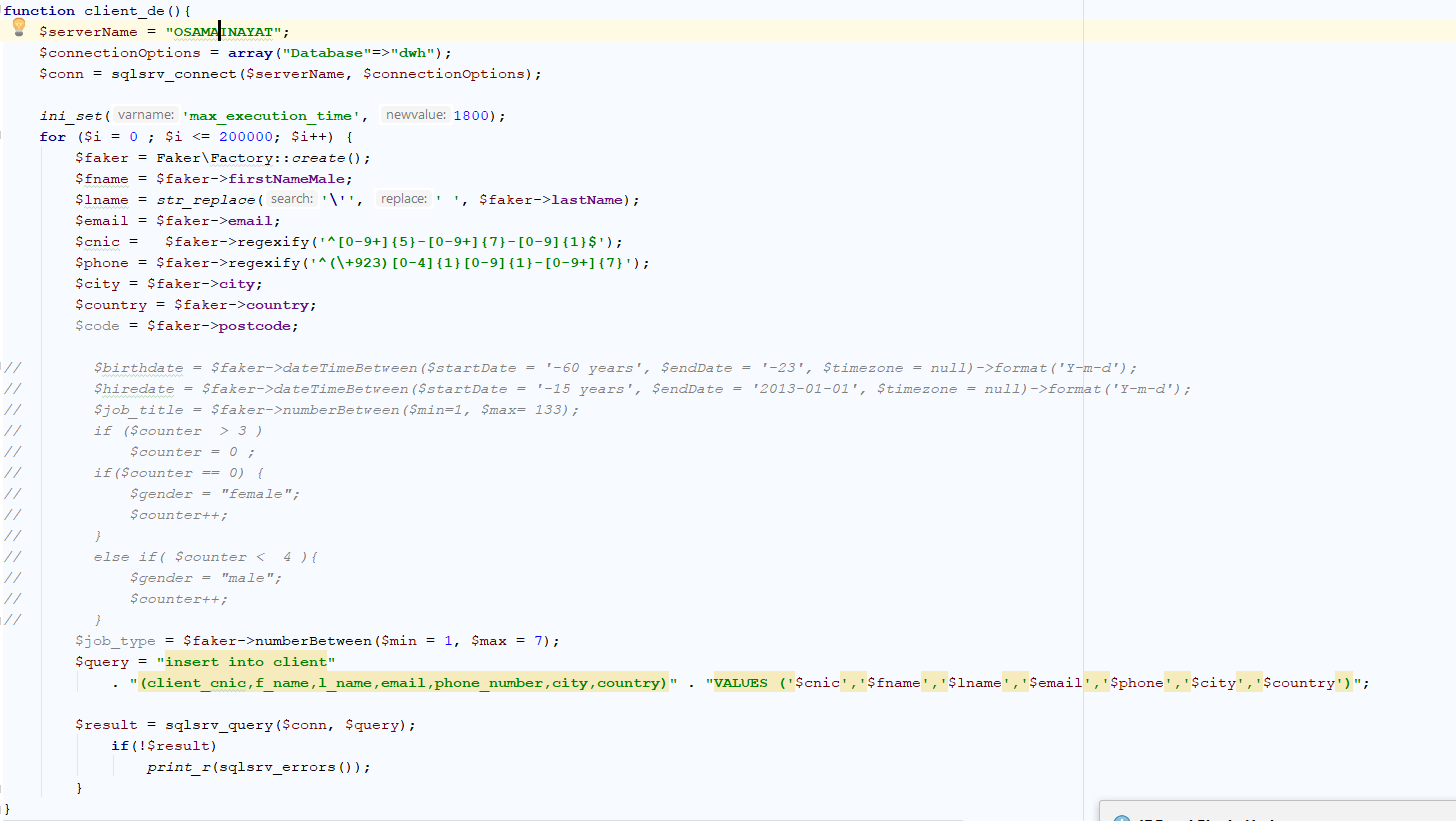
email varchar(35) not null,

city varchar(30),

country varchar(30),

phone\_number varchar(30)

);



# Job\_Title

create table job\_title(

id int not null identity primary key,

job\_title varchar(60)

);

# Salary

create table salary(

salary\_id int not null identity primary key,

employee\_id varchar(25) foreign key references employee(employee\_cnic),

basic\_salary float not null,

effective\_date date,

end\_date date,

approved\_by\_manager int foreign key references Hr\_Manager(id)

);

GO

Create procedure adding\_salaries

@hire\_date date,@temp\_year int,@emp\_cnic varchar(25),@salary float,

/\*INCREMENT ARGUMENTS\*/

@number\_of\_ratings int,@sum\_of\_ratings int,@manager int

AS

DECLARE @ending\_date date;

DECLARE @i int = 1;

WHILE (@i <= 12)

BEGIN

SET @ending\_date = DATEADD(MONTH,1,@hire\_date);

insert into salary (employee\_id,basic\_salary,effective\_date,end\_date,approved\_by\_manager)

values(@emp\_cnic,@salary,@hire\_date,@ending\_date,@manager);

SET @hire\_date = DATEADD(DAY,1,@ending\_date);

SET @i = @i+1;

END

IF(@number\_of\_ratings > 0 )

BEGIN

DECLARE @increment\_percentage float;

SET @increment\_percentage = (CAST(@sum\_of\_ratings as float)/CAST((@number\_of\_ratings\*10 ) as float))\*100;

IF(@increment\_percentage > 50 )

BEGIN

DECLARE @increment\_year int = YEAR(@hire\_date);

DECLARE @increment float = (Cast((@increment\_percentage/1000) as float))\*CAST(@salary as float);

SET @salary = @salary+@increment;

insert into increments(employee\_cnic,increment\_year,increment,salary\_after\_increment) values (@emp\_cnic,@increment\_year,@increment,@salary);

END

END

**GO**

**exec adding\_salaries @hire\_date='2008-08-21',@temp\_year='2008',@emp\_cnic='00012-9027189-6',@salary='75000',@number\_of\_ratings='1',@sum\_of\_ratings='9';**

**GO**

**create procedure updating @basic\_salary float, @employee\_id varchar (25) , @effective\_date date, @end\_date date**

**AS**

**update salary set basic\_salary=@basic\_salary where employee\_id=@employee\_id and effective\_date=@effective\_date**

**and end\_date=@end\_date;**

**GO**

**CREATE PROCEDURE updating\_salary @year int ,@employee\_cnic nvarchar(30)**

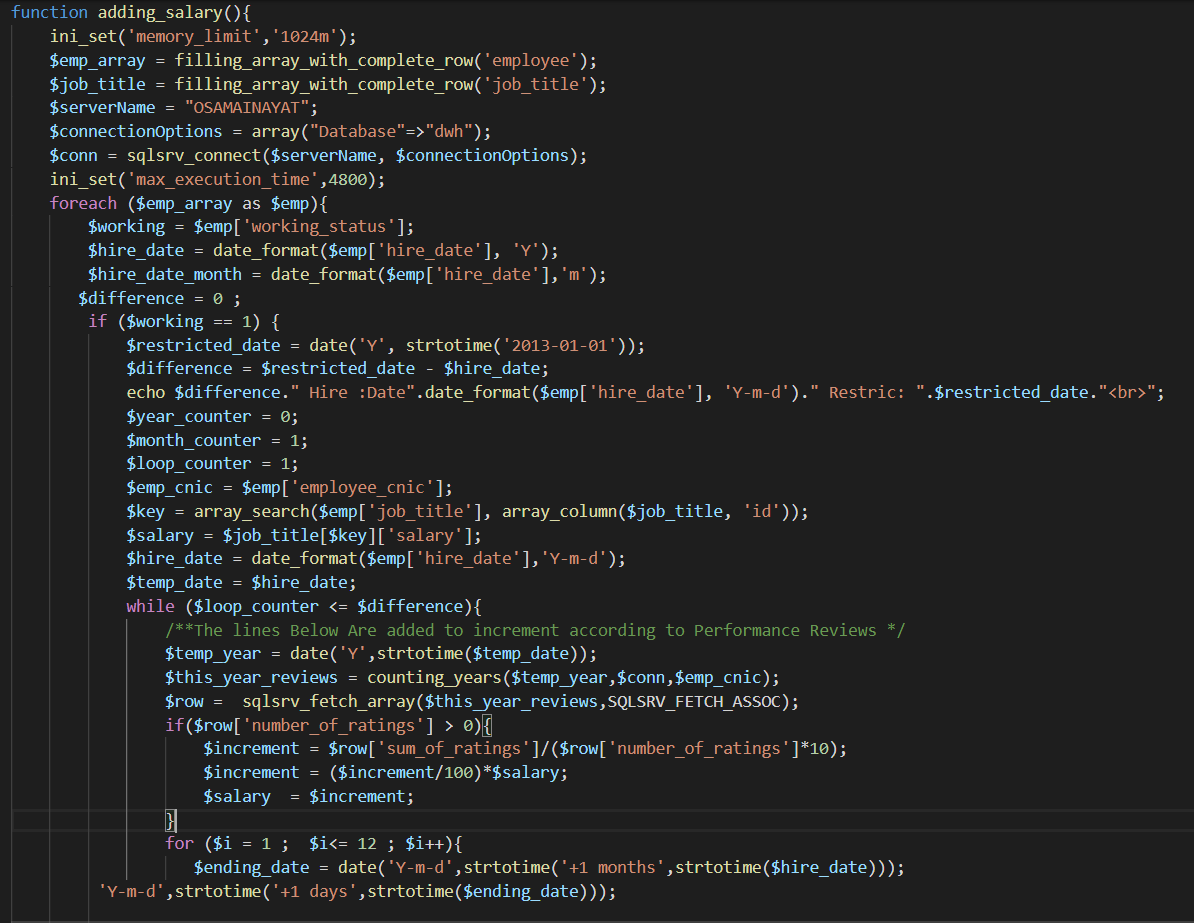
**AS**

**select count(ratings) as number\_of\_ratings,sum(ratings) as sum\_of\_ratings from performance\_reviews**

**where year(review\_date)=@year and employee\_id=@employee\_cnic;**

**GO**

**exec updating\_salary @year=2004 , @employee\_cnic='00019-8012201-7'**



# Increments

create table increments

(id int not null identity primary key,

employee\_cnic varchar(25) foreign key references employee(employee\_cnic),

increment\_year int, increment float, salary\_after\_increment float );

select \* from performance\_reviews;

Update increments set effective\_date = (select TOP(1) effective\_date from salary where basic\_salary= increments.salary\_after\_increment and increments.employee\_cnic = salary.employee\_id)

GO

Create procedure adding\_salaries

@hire\_date date,@temp\_year int,@emp\_cnic varchar(25),@salary float,

/\*INCREMENT ARGUMENTS\*/

@number\_of\_ratings int,@sum\_of\_ratings int,@manager int

AS

DECLARE @ending\_date date;

DECLARE @i int = 1;

WHILE (@i <= 12)

BEGIN

SET @ending\_date = DATEADD(MONTH,1,@hire\_date);

insert into salary (employee\_id,basic\_salary,effective\_date,end\_date,approved\_by\_manager)

values(@emp\_cnic,@salary,@hire\_date,@ending\_date,@manager);

SET @hire\_date = DATEADD(DAY,1,@ending\_date);

SET @i = @i+1;

END

IF(@number\_of\_ratings > 0 )

BEGIN

DECLARE @increment\_percentage float;

SET @increment\_percentage = (CAST(@sum\_of\_ratings as float)/CAST((@number\_of\_ratings\*10 ) as float))\*100;

IF(@increment\_percentage > 50 )

BEGIN

DECLARE @increment\_year int = YEAR(@hire\_date);

DECLARE @increment float = (Cast((@increment\_percentage/1000) as float))\*CAST(@salary as float);

SET @salary = @salary+@increment;

insert into increments(employee\_cnic,increment\_year,increment,salary\_after\_increment) values (@emp\_cnic,@increment\_year,@increment,@salary);

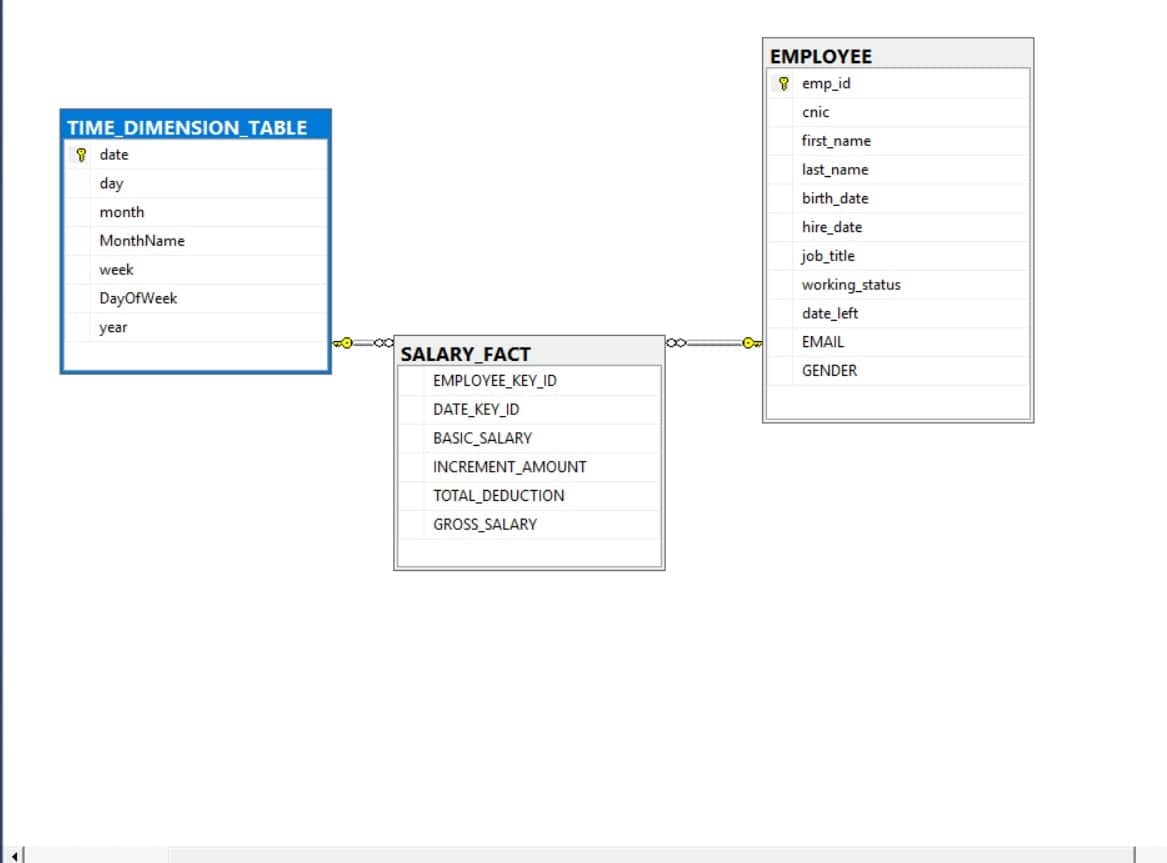
END

END

GO

Schema Modeling

# Star Schema



# WHY WE CHOOSE STAR SCHEMA FOR CREATING DATA WAREHOUSE ?

As we know that a **star schema** is used as a basic implementation of an OLAP cube. If the fact table contains a 1 to many relationships to each of the dimensions in the data warehouse schema then we know that it is appropriate to **use** a **star schema.** Considering our system, the dimensions are in a 1-to-many relation with the fact table. That is why we choose star schema.

The reason we are using the Star schema is due to its advantage in the decision support environment. The star schema is faster in query processing unlike others such as OLTP system.

The number of tables in a star schema are fewer and it makes it easy to understand the join paths as they are clear.

Star schema performs well in loading and administration is quite easy compare to others.

Another main reason for star schema is the built-in referential integrity.

The relations and the schema modeling can be easily understood by anyone who is not much familiar with the System.

In computing, the **star schema** is the simplest style of data mart **schema** and is the approach most widely **used** to develop data warehouses and dimensional data marts.

The **star schema** is an important special case of the snowflake **schema**, and is more effective for handling simpler queries.

Hierarchies are split into different tables (Sub Dimensions). The snowflake schemais normalized. So, the data access latency is less in **star schema** in comparison to snowflake schema. As the **star schema** is denormalized, the size of the **data warehouse** will be larger than that of snowflake schema.

Business Process Model

**Statement:**

Once an employee has completed its job assignment. Client has an option of giving ratings to an employee on the basis of job assignment which is done by employee. Each employee has a rating chart which is updated after every job assignment. If the year is completed (it’s the last assignment of the year) Increment is added to the employee’s salary on the basis of rating’s ratio after every year.

**Process model of Salary Increment**

