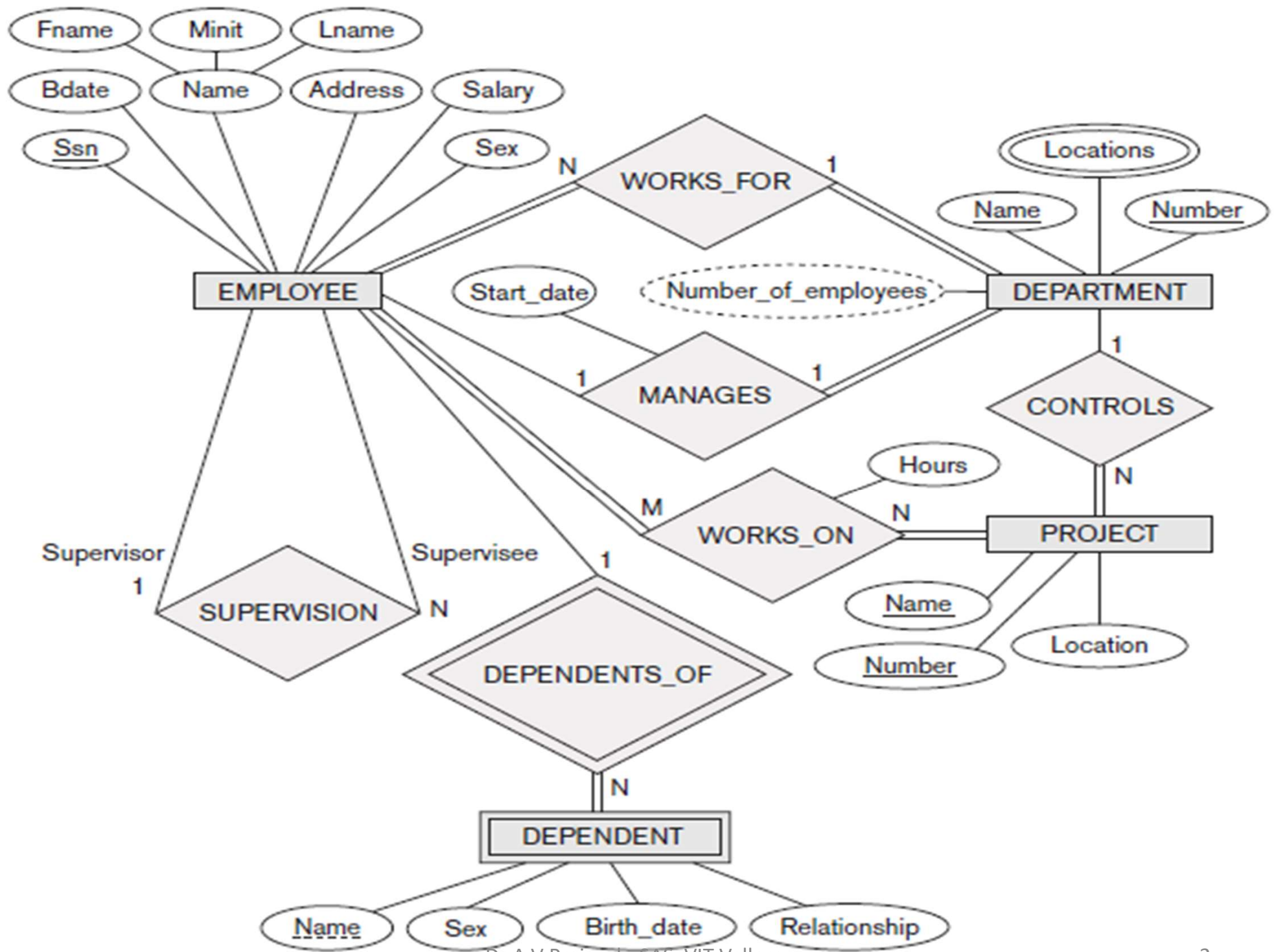


# Mapping ER model to a Relational Schema

ER Model, when conceptualized into diagrams, gives a good overview of entity-relationship, which is easier to understand. ER diagrams can be mapped to relational schema, that is, it is possible to create relational schema using ER diagram.

We cannot import all the ER constraints into relational model, but an approximate schema can be generated.

There are several processes and algorithms available to convert ER Diagrams into Relational Schema. Some of them are automated and some of them are manual. We may focus here on the mapping diagram contents to relational basics.



## EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

## DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

## DEPT\_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

## PROJECT

Pname	<u>Pnumber</u>	<u>Plocation</u>	Dnum
-------	----------------	------------------	------

## WORKS\_ON

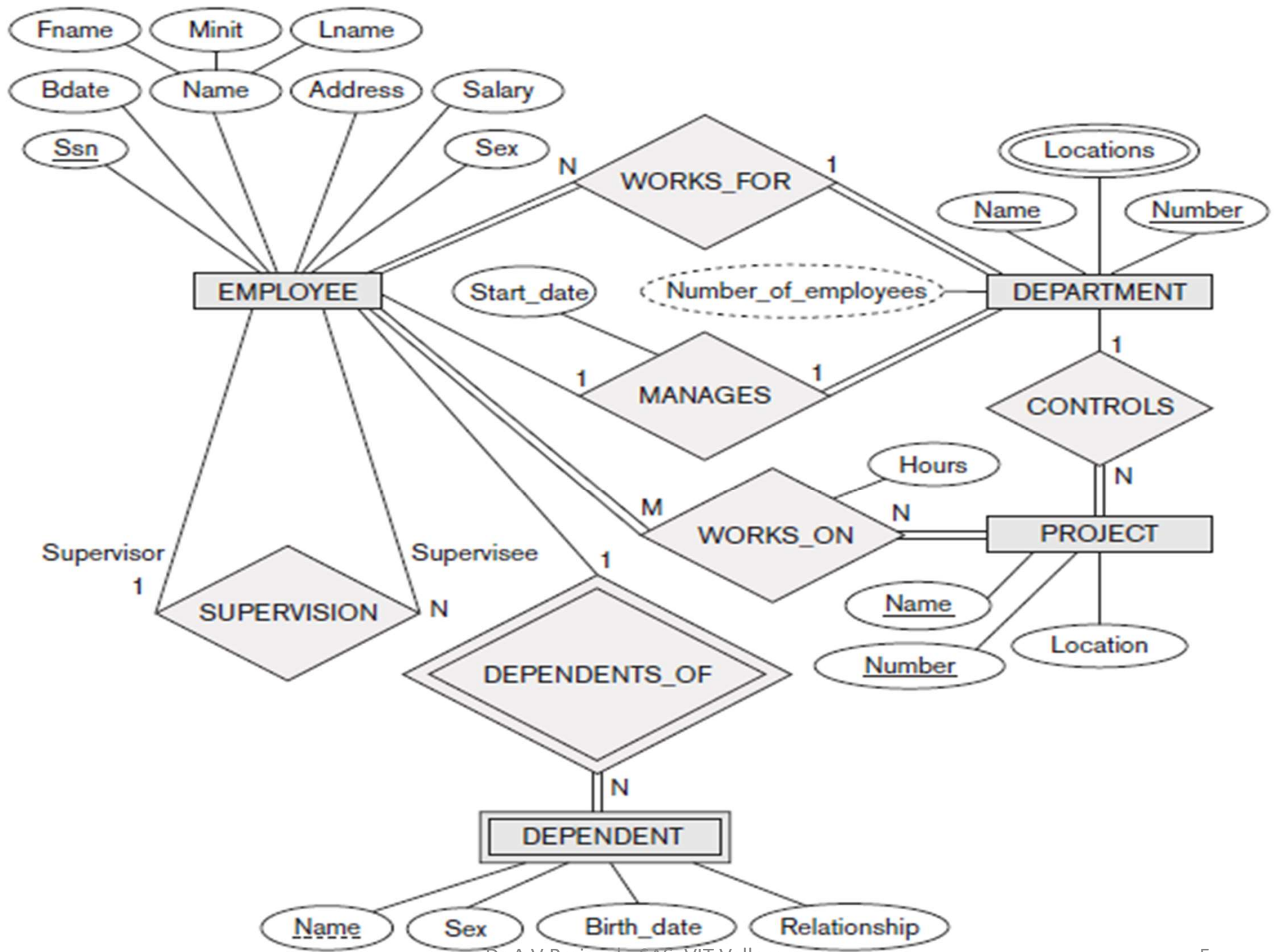
<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

## DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------

# Converting Strong entity types

- Each **entity type** becomes a **table**
- Each **single-valued attribute** becomes a **column**
- **Derived attributes** are **ignored**
- **Composite attributes** are represented by **components**
- **Multi-valued attributes** are represented by a **separate table**
- The **key attribute** of the entity type becomes the **primary key** of the table



Strong Entity types with key attributes as primary key

### EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary
-------	-------	-------	------------	-------	---------	-----	--------

### DEPARTMENT

Dname	<u>Dnumber</u>
-------	----------------

### PROJECT

Pname	<u>Pnumber</u>	<u>Plocation</u>
-------	----------------	------------------

## Multi valued attributes

### DEPT\_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

The primary key of strong entity becomes the foreign key in the multivalued attribute table. Here department table is taken as the foreign key in Dept\_locations table.

### DEPARTMENT

Dname	<u>Dnumber</u>
-------	----------------

### DEPT\_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

## Weak Entity types

### DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------

The primary key of strong entity becomes the foreign key in weak entity table

The primary key of weak entity tables is a combination of respective strong entity is a combination of strong entity primary key and weak entity partial key.



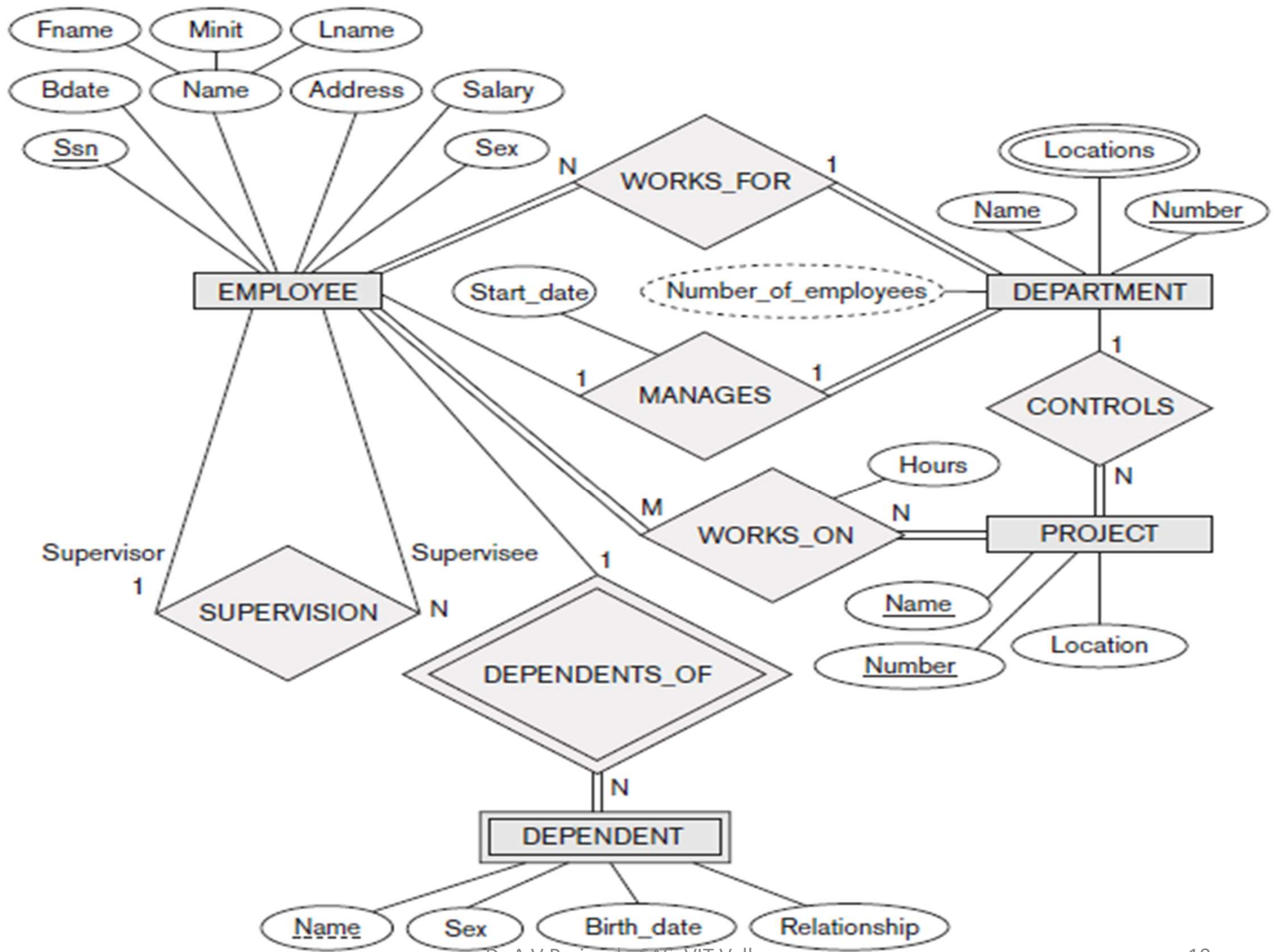
## EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary
-------	-------	-------	------------	-------	---------	-----	--------

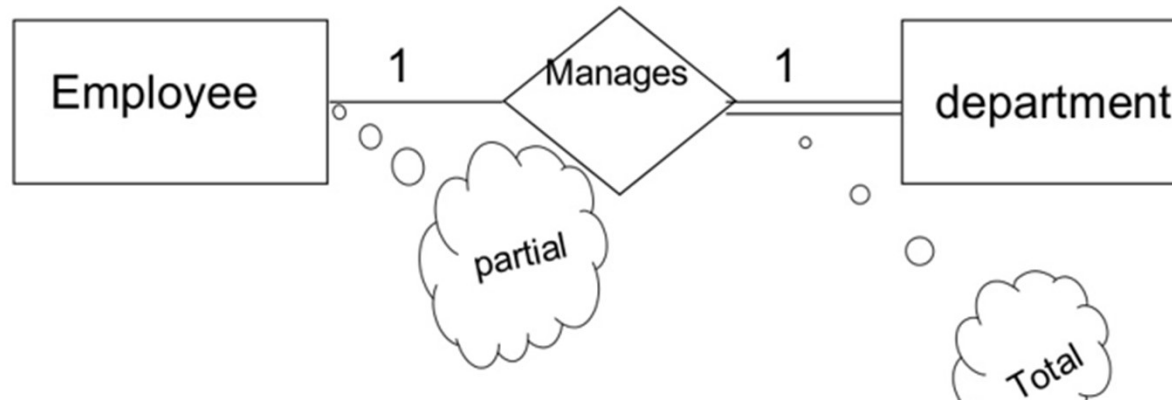
## DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------





## Binary 1:1 relationship



The primary key of the partial participant will become the foreign key of the total participant

### EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary
-------	-------	-------	------------	-------	---------	-----	--------

### DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

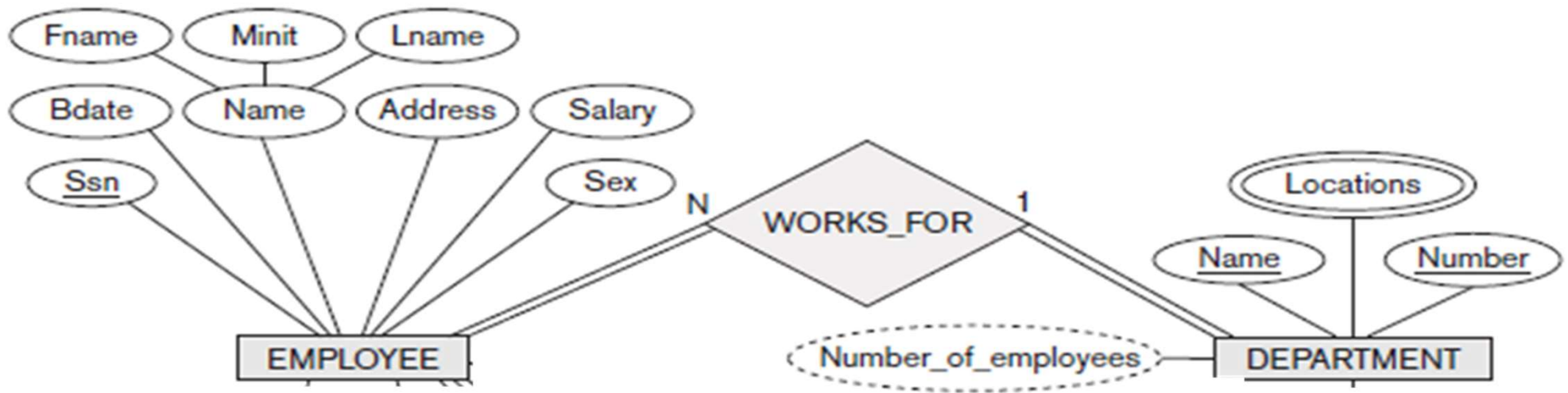
## **Binary 1:1 relationship with both partial participation**

1:1 relationship with partial participation from two entity set S and R. In that case either place the primary key of S and describing attribute of the relationship to the R relation as foreign key, or place the primary key of the R and describing attribute of the relationship to the S relation as foreign key.

## **Binary 1:1 relationship with both total participation**

In this case combine both the relations S and R and include all the attributes of S and attributes of R in the same table. Primary key of new table is the primary keys of S and R together.

## Binary 1:N relationship



The primary key of the relation on the “1” side of the relationship becomes a foreign key in the relation on the “N” side

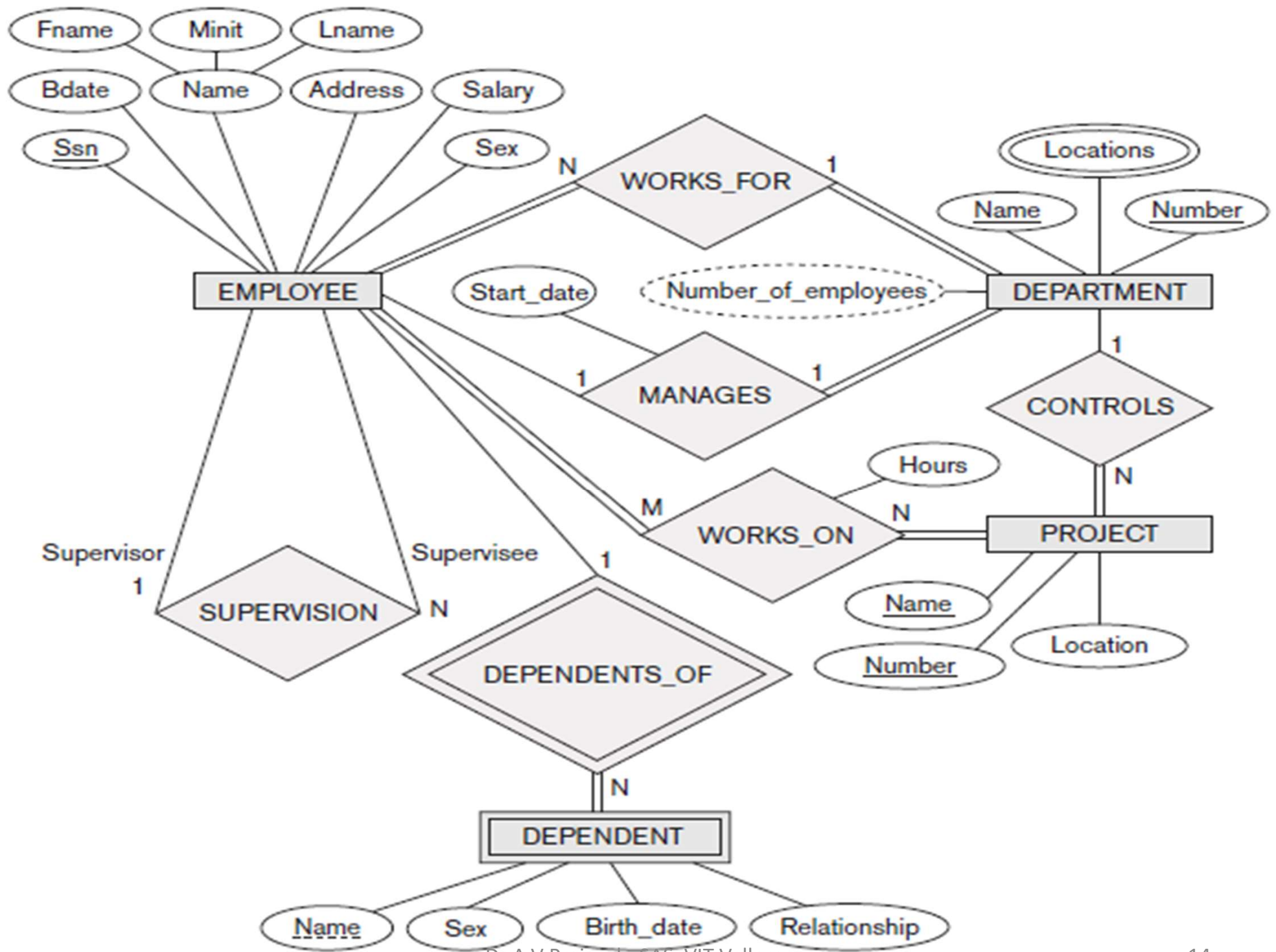
### EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----

### DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

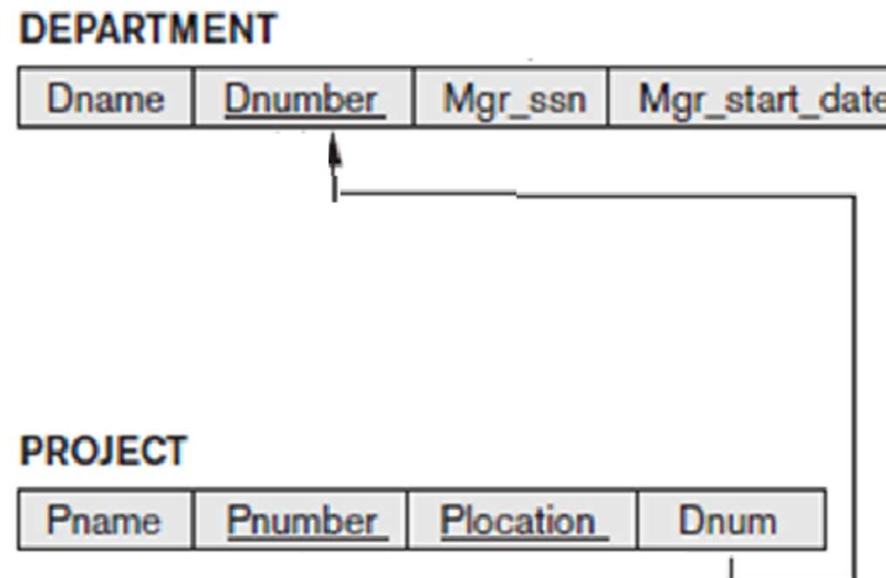




we map the 1:N relationship types WORKS\_FOR, CONTROLS, and SUPERVISION.

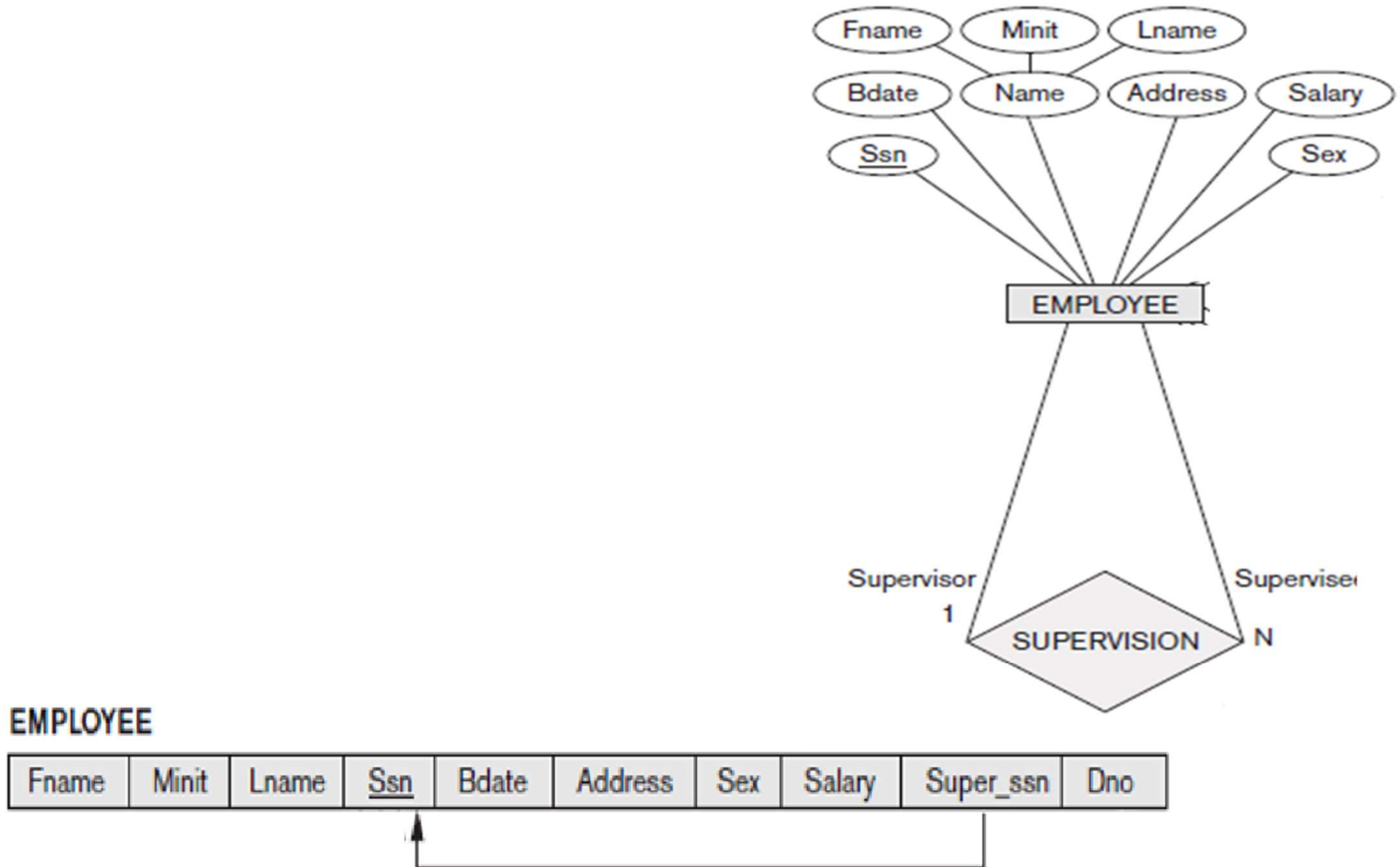
WORKS\_FOR we include the primary key Dnumber of the DEPARTMENT relation as foreign key in the EMPLOYEE relation and call it Dno.

The CONTROLS relationship is mapped to the foreign key attribute Dnum of PROJECT, which references the primary key Dnumber of the DEPARTMENT relation.





For SUPERVISION we include the primary key of the EMPLOYEE relation as foreign key in the EMPLOYEE relation itself—because the relationship is recursive—and call it Super\_ssn.





## EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

## DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

## DEPT\_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

## PROJECT

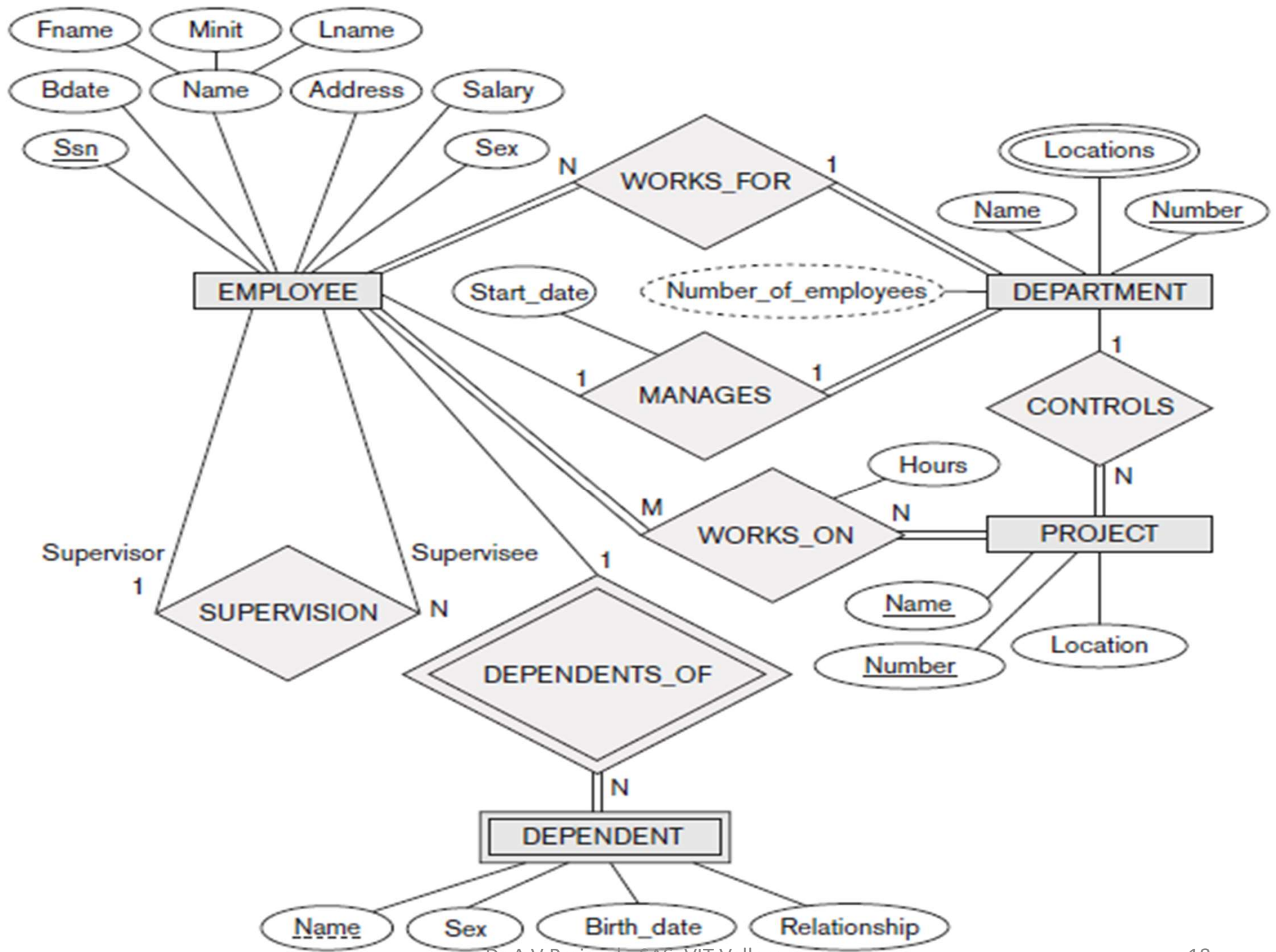
Pname	<u>Pnumber</u>	<u>Plocation</u>	Dnum
-------	----------------	------------------	------

## WORKS\_ON

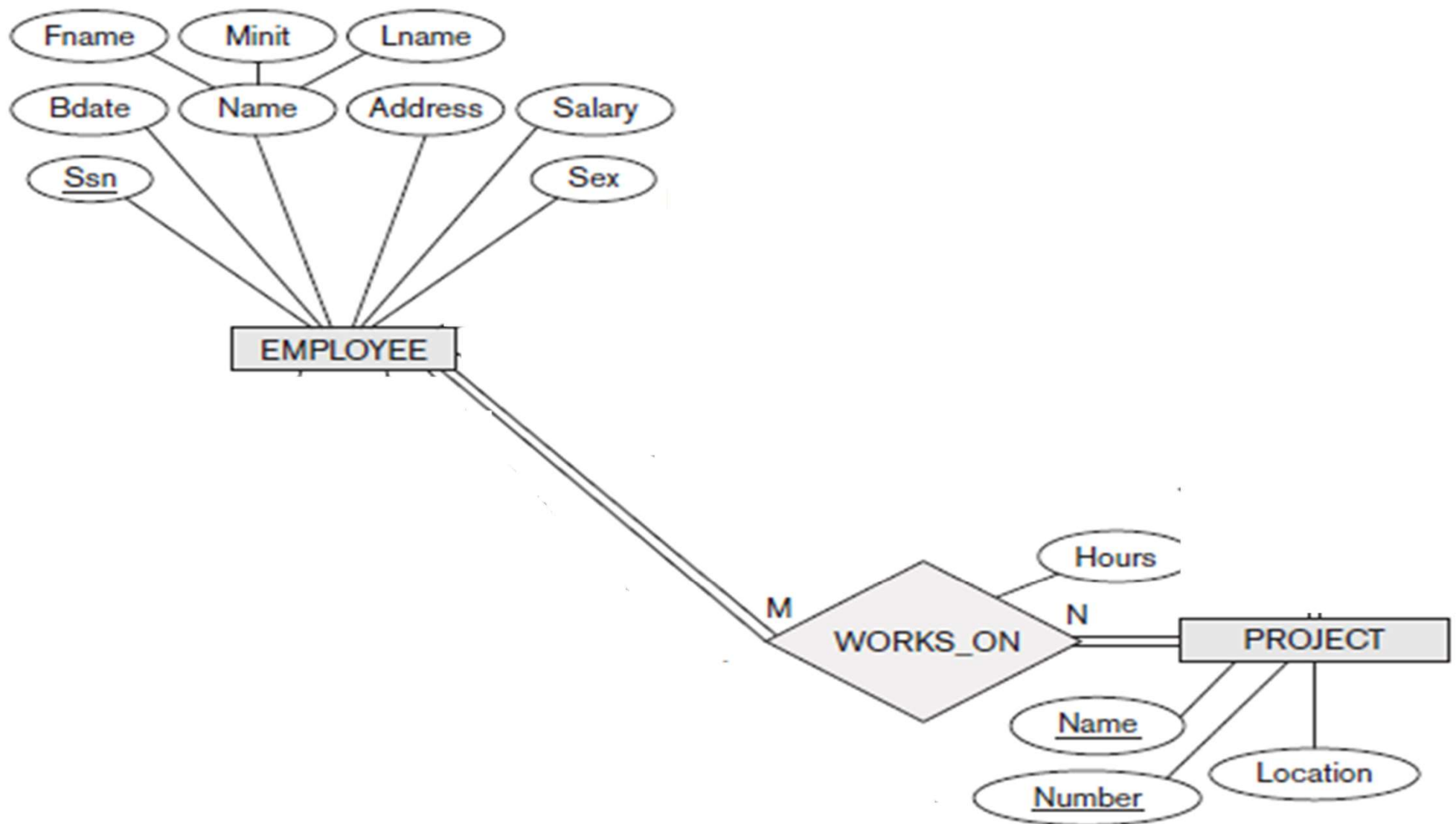
<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

## DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------



# Binary M:N relationships



A new table is created to represent the relationship

Contains two foreign keys - one from each of the participants in the relationship

The primary key of the new table is the combination of the two foreign keys

### EMPLOYEE

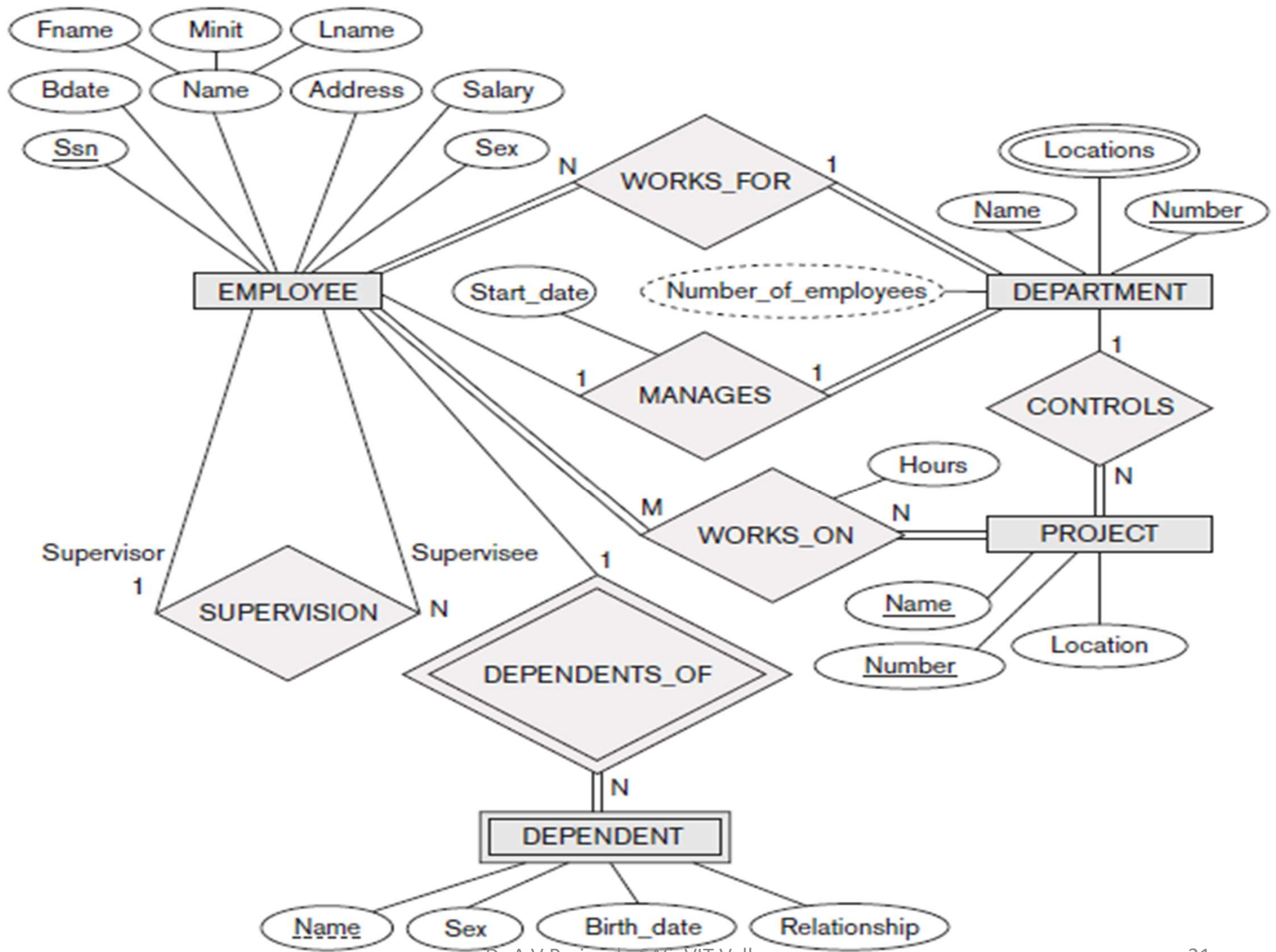
Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

### PROJECT

Pname	<u>Pnumber</u>	<u>Plocation</u>	Dnum
-------	----------------	------------------	------

### WORKS\_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------



## EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

## DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

## DEPT\_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

## PROJECT

Pname	<u>Pnumber</u>	<u>Plocation</u>	Dnum
-------	----------------	------------------	------

## WORKS\_ON

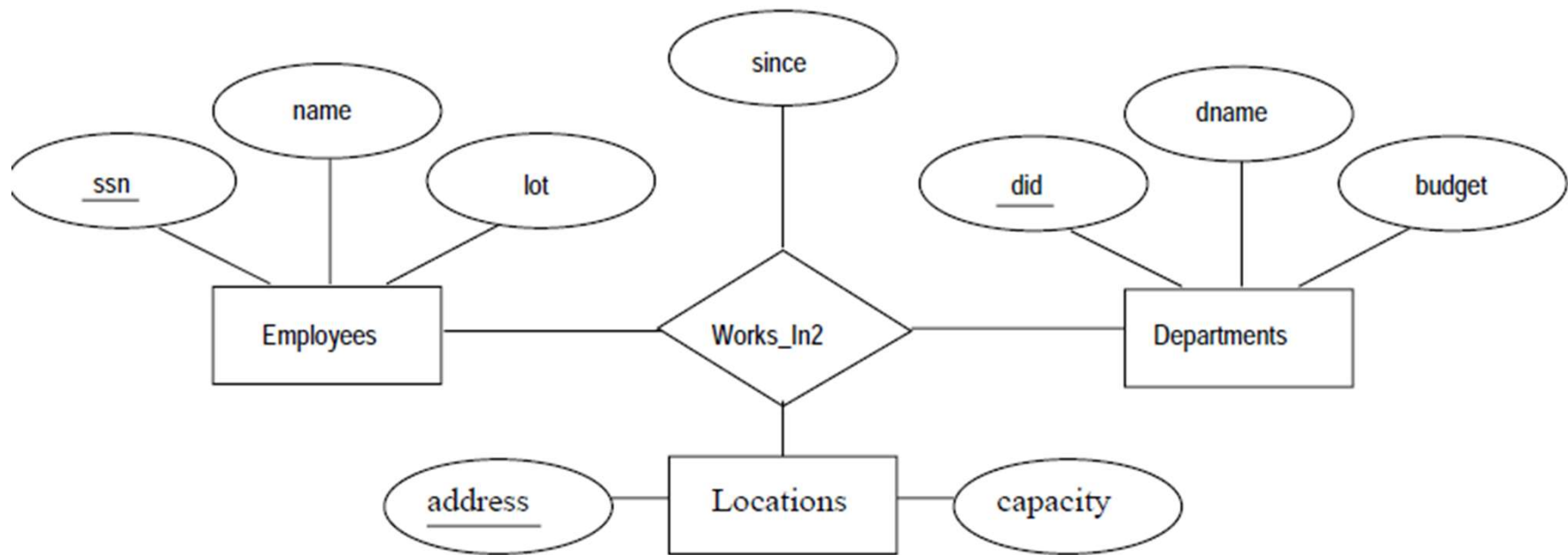
<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

## DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------

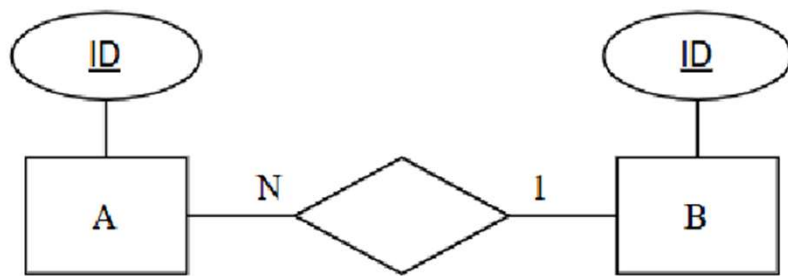


# N-ary Relationships




```
CREATE TABLE Works_In2 ( ssn      CHAR(11),
                          did      INTEGER,
                          address  CHAR(20),
                          since    DATE,
                          PRIMARY KEY (ssn, did, address),
                          FOREIGN KEY (ssn) REFERENCES Employees,
                          FOREIGN KEY (address) REFERENCES Locations,
                          FOREIGN KEY (did) REFERENCES Departments )
```

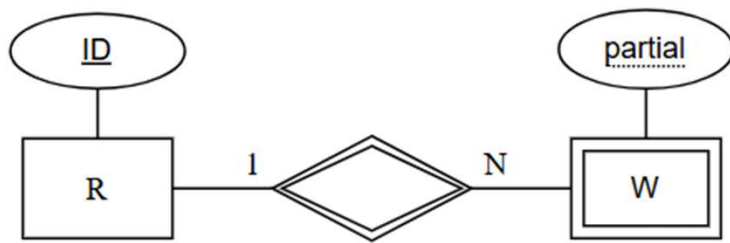
Q-2: Consider the ERD below. We create tables **a** and **b**, each of which have a primary key column named “id”. (Assume there are additional columns from attributes not shown.) What is the simplest way to convert the relationship between **A** and **B**? ✓



- A. Create a column named “a\_id” in table **b**, and make it a foreign key referencing table **a**.
- B. Create a column named “b\_id” in table **a**, and make it a foreign key referencing table **b**.
- C. Create a cross-reference table, **a\_b**, containing columns **a\_id** and **b\_id** as foreign keys referencing **a** and **b** respectively.
- D. Merge tables **a** and **b** into a new table.

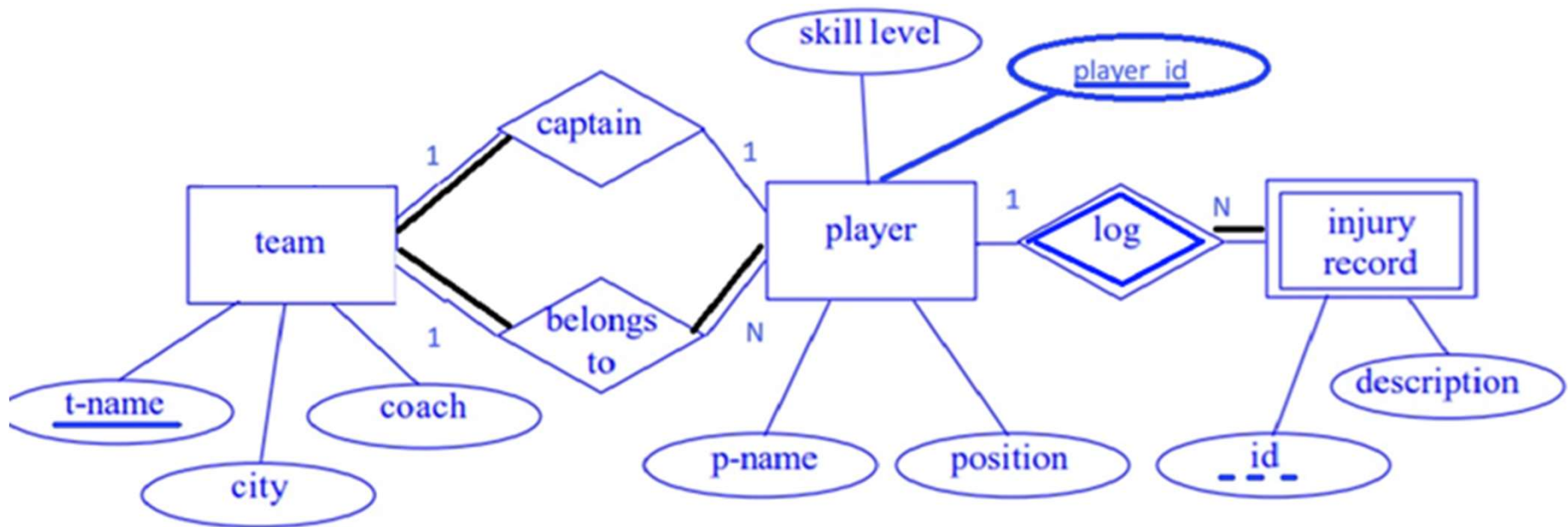


Q-3: Consider the ERD below. We create table **r** with primary key column **id**. What should table **w** look like? 



- ☐ A. The table should have a column **partial** as primary key. Additionally, create a cross-reference table **r\_w**.
- ☐ B. The table should have columns **partial** and **r\_id**. The primary key is **partial**. Add a foreign key constraint on **r\_id** referencing **r**.
- ☐ C. The table should have columns **partial** and **r\_id**. The primary key is a composite of **r\_id** and **partial**. Add a foreign key constraint on **r\_id** referencing **r**.
- ☐ D. The table should have columns **partial** and **r\_id**. The primary key is **r\_id**. Add a foreign key constraint on **r\_id** referencing **r**.

## Convert to relational schema



### Team

Tname (PK)  
Coach  
City  
captainPlayerid (FK)

### Player

Playerid (PK)  
Skill level  
Pname  
Position  
belongstoTname (FK)

### Injury record

Id(PK)  
LogPlayerid (PK)(FK)  
description