

Addis Ababa Institute of Technology

Department of Electrical and computer Engineering

Group 2's project – phase 2

Title: Hospital Management System

Section: Computer Stream/ 4th year

Group Names

1. Plato Gebremdhin
2. Samuel Teshome
3. Tigist Solomon
4. Leul Mengistu
5. Nahom Seleshi

Id No

Atr/1910/10
Atr/5845/10
Atr/9012/11
Atr/9127/10
Atr/8157/10

Submission Date: May/3/2021

Submitted To: INS. Tesfamichael

Mapping of Entities

Employee Super class and it's Subclasses:

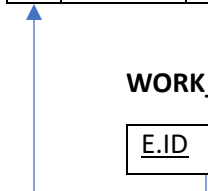
The **EMPLOYEE** entity has several attributes of which the **ID** is the primary key. This entity has 3 composite attributes, when mapping this entity we can consider each of the composite attributes as a single attribute as shown in the table below, there is also a multivalued attribute which is **WorkTime**, for this, another relation called **WORK_TIME** will be constructed that will consist of the attributes of **WorkTime** and a foreign key which is taken from the primary key of the **EMPLOYEE** relation, the primary key of this relation will be the combination of both the foreign key and the **WorkDay** attribute, the other attributes of the employee entity are strong attributes so for each strong attribute a single attribute will be added to the entity relation. The Entity employee has 3 subclasses under it, namely: **DOCTORS**, **MANAGERS** and **OTHER_STAFF**. It is known that each of these entities have no extra attribute of their own, they are disjoint to each other meaning they don't overlap and all subclass completely fill the **EMPLOYEE** superclass, thus the best option to map these entities would be to add a single attribute to the superclass (**EMPLOYEE** entity) called **JobType** and merge all three subclasses to the super class. We will have the following relation:

EMPLOYEE

<u>ID</u>	F.NAME	M.int	L.NAME	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType
-----------	--------	-------	--------	---------	-----	--------	---------	--------	------	--------	------------	---------

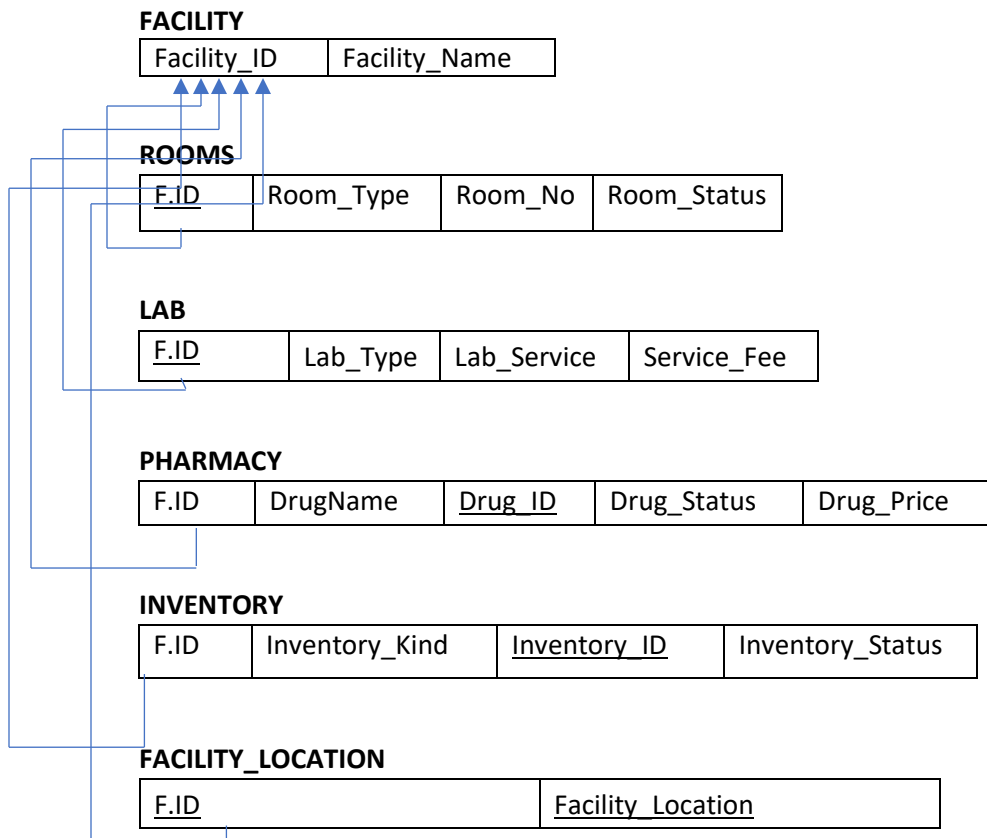
WORK_TIME

<u>E.ID</u>	StartTime	EndTime	<u>WorkDay</u>
-------------	-----------	---------	----------------



Facility Super class and it's Subclasses:

The **FACILITY** entity has several attributes of which the **ID** is the primary key. This entity has 2 simple attributes and 1 multivalued attribute, when mapping this entity, we can create a relation called **Facility Location** for the multivalued attribute which will contain the attribute **Facility location** and **Facility ID** as a foreign key from **Facility** as shown in the table below, the other attributes of the facility entity are simple attributes. The Entity facility has 4 subclasses under it, namely: **Medical care units (ROOMS)**, **Clinical labs and Radiology Center (LABS)**, **PHARMACY**, and **INVENTORY**. These entities have their own attribute, they are disjoint to each other meaning they don't overlap, thus we can map these entities by using additional table for each subclass. We will have the following relation:



SPECIALIZATION:

The SPECIALIZATION entity has two attributes which are S_ID and S_Name with S_ID being the primary key. All the attributes are simple and there is only one functional dependency.

SPECIALIZATION

<u>S_ID</u>	S_Name
-------------	--------

PATIENT ENTITY:

The **patient** entity has several attributes of which the **ID** is the primary key. This entity has 3 simple attributes and 3 composite attributes, when mapping this entity we can consider each of the composite attributes as a single attribute as shown in the table below. so, we will have the following relation:

PATIENT

<u>ID</u>	F.NAME	M.NAME	L.NAME	Phone No	Sex	B.Date	B.Month	B.Year	City	County	Street Name
-----------	--------	--------	--------	----------	-----	--------	---------	--------	------	--------	-------------

STATUS ENTITY:

Status entity is a relation that is related to the employee and manager entities with two relationships **HAS** and **MANAGES** respectively. Status entity has StatusID, StatusLevel, ClearanceLevel and Salary attributes it has also a composite attribute called SalatyStatus which is composed of Paid attribute and unpaid Number of months, each act as there own attribute in the relation. StatusID will be the primary key to the relation.

STATUS

<u>StatusID</u>	StatusLevel	ClearanceLevel	Salary	Unpaid Number of months	Paid
-----------------	-------------	----------------	--------	-------------------------	------

Mapping of Relationships

WORKS_ON:

This relation Ship relates the **OTHER_STAFF** entity and **FACILITY** entity, since this a complete relation ship with cardinality N:1, the cross refence relation can be used to map this relation, thus this new relation will consist of a foreign key referred from the primary Key of **EMPLOYEE** and another foreign key referred from the primary key of **FACILITY**. The primary key of this relation will be the combination of both foreign keys. we will have the following relation:

EMPLOYEE

<u>ID</u>	F.Name	M.int	L.Name	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType
-----------	--------	-------	--------	---------	-----	--------	---------	--------	------	--------	------------	---------

FACILITY

<u>ID</u>	...
-----------	-----

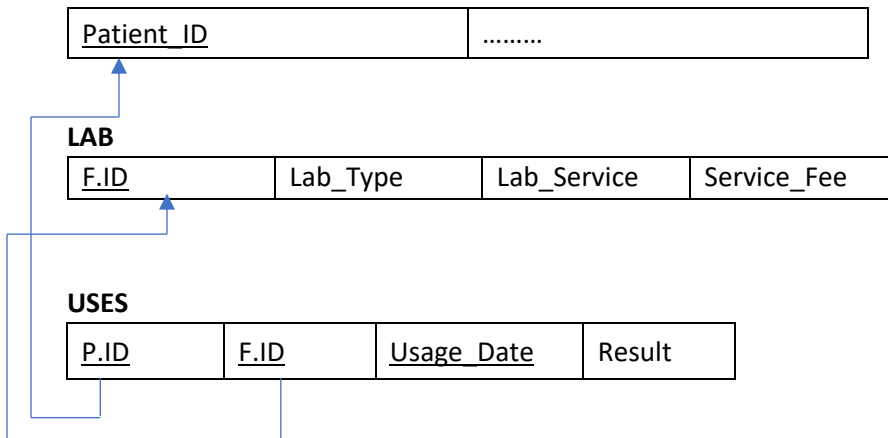
WORKS_ON

<u>E.ID</u>	<u>F.ID</u>
-------------	-------------

USES:

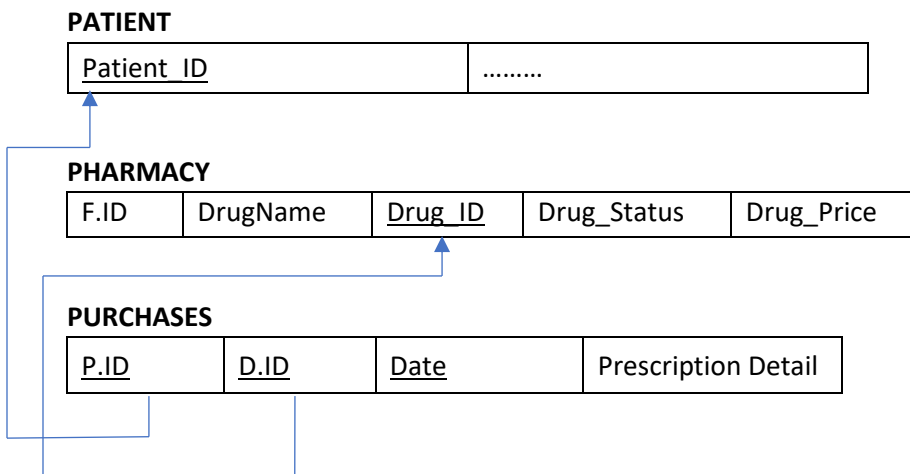
This relation Ship relates the PATIENT entity and LABS Subclass entity, this is not a complete relationship with cardinality N:M, this relation will have its own attributes, thus this new relation will consist of a foreign key referred from the primary Key of PATIENT and another foreign key referred from the primary key of LABS and it will also contain 2 simple attributes of its own. The primary key of this relation will be the combination of both foreign keys and the attribute UsageDate. we will have the following relation:

PAITENT



PURCHASES:

This relation Ship relates the PATIENT entity and PHARMACY Subclass entity, this is not a complete relationship with cardinality N:M, this relation will have its own attributes, thus this new relation will consist of a foreign key referred from the primary Key of PATIENT and another foreign key referred from the primary key of PHARMACY and it will also contain 2 simple attributes of its own. The primary key of this relation will be the combination of both foreign keys and Date. we will have the following relation:



ASSIGNED IN:

This relation Ship relates the PATIENT entity and ROOMS Subclass entity, this is not a complete relationship with cardinality N:M, this relation will have its own attributes, thus this new relation will consist of a foreign key referred from the primary Key of PATIENT and another foreign key referred from the primary key of ROOMS and it will also contain 2 simple attributes of its own. The primary key of this relation will be the combination of both foreign keys and EntryDate. we will have the following relation:

PATIENT

<u>Patient_ID</u>
-------------------	-------

ROOMS

<u>F.ID</u>	Room_Type	Room_No	Room_Status
-------------	-----------	---------	-------------

ASSIGNED_IN

<u>P.ID</u>	<u>Room_NO</u>	<u>Entry_Date</u>	<u>Discharge_Date</u>
-------------	----------------	-------------------	-----------------------

APPOINTMENT:

This relationship relates the **PATIENT** entity and **DOCTORS** entity, since this a partial relationship with cardinality N:1. Thus for appointment we include the primary key Doctor ID of the DOCTORS relation as foreign key in the PATIENT relation and call it D.ID. The attributes of the relation appointment will also be added as an attribute in the patient relation.

We will have the following relation:

PATIENT

<u>ID</u>	F.NAME	M.NAME	L.NAME	Phone No	Sex	B.Date	B.Month	B.Year	City	County	Street Name	<u>D.ID</u>	Start time	date
-----------	--------	--------	--------	----------	-----	--------	---------	--------	------	--------	-------------	-------------	------------	------

EMPLOYEE

<u>ID</u>	F.NAME	M.int	L.NAME	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType
-----------	--------	-------	--------	---------	-----	--------	---------	--------	------	--------	------------	---------

DIAGNOSIS RESULT:

This relationship relates the **PATIENT** entity and **DOCTORS** entity, since this a full relationship with cardinality 1:1. Thus for diagnosis result we include the primary key Doctor ID of the DOCTORS relation as foreign key in the PATIENT relation and call it D.ID. The attributes of the relation diagnosis report will also be added as an attribute in the patient relation.

We will have the following relation:

PATIENT

<u>ID</u>	F.NAME	M.NAME	L.NAME	Phone No	Sex	B.Date	B.Month	B.Year	City	County	Street Name	<u>D.ID</u>	Start time	date	prescription	case
-----------	--------	--------	--------	----------	-----	--------	---------	--------	------	--------	-------------	-------------	------------	------	--------------	------

EMPLOYEE

<u>ID</u>	F.NAME	M.int	L.NAME	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType
-----------	--------	-------	--------	---------	-----	--------	---------	--------	------	--------	------------	---------

SPECIALIZES_IN:

This relationship relates the **EMPLOYEE** entity and **SPECIALIZATION** entity. Using this, another relation called SPECIALIZES_IN is constructed that will consist the attributes of the relation EMPLOYEE (which is E_ID and serves as a foreign key here) and the attribute SPECIALIZATION (which is S_ID and serves as a foreign key here).

EMPLOYEE

<u>ID</u>	F.NAME	M.int	L.NAME	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType
-----------	--------	-------	--------	---------	-----	--------	---------	--------	------	--------	------------	---------

SPECIALIZES_IN

<u>E-ID</u>	<u>S_ID</u>
-------------	-------------

SPECIALIZATION

<u>S_ID</u>	S_Name
-------------	--------

HAS:

This relationship relates the **EMPLOYEE** entity and the **STATUS** entity, and the cardinality ratio is N:1 with complete relationship, thus we can add an extra attribute to the **EMPLOYEE** entity which is a foreign key that is taken from the **STATUS** entity

EMPLOYEE

<u>ID</u>	F.NAME	M.int	L.NAME	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType	S.ID
-----------	--------	-------	--------	---------	-----	--------	---------	--------	------	--------	------------	---------	------

STATUS

<u>StatusID</u>	StatusLevel	ClearanceLevel	Salary	Unpaid Number of months	Paid
-----------------	-------------	----------------	--------	-------------------------	------

MANAGES:

Manages is a relationship that relates EMPLOYEE entity to the STATUS entity and there is a complete relation with manager and status entity with N:M cardinality. To map this relationship that is between the manager and the status entities the cross reference can be used by creating a new relation with the managerID and statusID which are the primary keys in their respective relation tables and the combination of the two keys will be the key to this new relation.

EMPLOYEE

<u>ID</u>	F.NAME	L.NAME	PhoneNo	Sex	B. Date	B. Month	B. Year	City	County	Street Name	JobType	S.ID
-----------	--------	--------	---------	-----	---------	----------	---------	------	--------	-------------	---------	------

MANAGES

<u>ManagerID</u>	<u>StatusID</u>
------------------	-----------------

STATUS

<u>StatusID</u>	StatusLevel	ClearanceLevel	Salary	Unpaid number of months	Paid
-----------------	-------------	----------------	--------	-------------------------	------

THIS IS THE OVERALL MAPPED DIAGRAM:

EMPLOYEE

<u>ID</u>	F.NAME	M.int	L.NAME	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType	S.ID
-----------	--------	-------	--------	---------	-----	--------	---------	--------	------	--------	------------	---------	------

PATIENT

<u>ID</u>	F.NAME	M.NAME	L.NAME	Phone No	Sex	B.Date	B.Month	B.Year	City	County	Street Name	<u>D.ID</u>	Start time	date	prescription	case
-----------	--------	--------	--------	----------	-----	--------	---------	--------	------	--------	-------------	-------------	------------	------	--------------	------

SPECIALIZATION

<u>S_ID</u>	S_Name
-------------	--------

SPECIALIZES_IN

<u>E-ID</u>	<u>S_ID</u>
-------------	-------------

WORK_TIME

<u>E.ID</u>	StartTime	EndTime	<u>WorkDay</u>
-------------	-----------	---------	----------------

WORKS_ON

<u>E.ID</u>	<u>F.ID</u>
-------------	-------------

FACILITY

<u>Facility_ID</u>	Facility_Name
--------------------	---------------

ROOMS

<u>F.ID</u>	Room_Type	Room_No	Room_Status
-------------	-----------	---------	-------------

LAB

<u>F.ID</u>	Lab_Type	Lab_Service	Service_Fee
-------------	----------	-------------	-------------

PHARMACY

<u>F.ID</u>	DrugName	<u>Drug_ID</u>	Drug_Status	Drug_Price
-------------	----------	----------------	-------------	------------

INVENTORY

<u>F.ID</u>	Inventory_Kind	<u>Inventory_ID</u>	Inventory_Status
-------------	----------------	---------------------	------------------

FACILITY_LOCATION

<u>F.ID</u>	<u>Facility_Location</u>
-------------	--------------------------

ASSIGNED_IN

<u>P.ID</u>	<u>Room_NO</u>	<u>Entry_Date</u>	Discharge_Date
-------------	----------------	-------------------	----------------

PURCHASES

<u>P.ID</u>	<u>D.ID</u>	<u>Date</u>	Prescription Detail
-------------	-------------	-------------	---------------------

USES

<u>P.ID</u>	<u>F.ID</u>	<u>Usage_Date</u>	Result
-------------	-------------	-------------------	--------

INVENTORY

<u>F.ID</u>	Inventory_Kind	<u>Inventory_ID</u>	Inventory_Status
-------------	----------------	---------------------	------------------

<u>StatusID</u>	StatusLevel	ClearanceLevel	Salary	Unpaid number of months	Paid
-----------------	-------------	----------------	--------	-------------------------	------

<u>ManagerID</u>	<u>StatusID</u>
------------------	-----------------

Functional dependencies

EMPLOYEE:

In the **EMPLOYEE** relation the primary key (**ID**) determines every other entity so this will be our only functional dependency since there are no other dependencies, thus the closure of ID will be:

$\{ID\}^+ = \{ID, F.Name, M.Int, L.Name, PhoneNo, Sex, B.Date, B.Month, B.Year, City, Country, StreetName, JobType, S.ID\} = \text{EMPLOYEE}$

EMPLOYEE

<u>ID</u>	F.Name	M.int	L.Name	PhoneNo	Sex	B.Date	B.Month	B.Year	City	County	StreetName	JobType	S.ID

FACILITY:

In the **FACILITY** relation the primary key (**Facility_ID**) determines every other entity so this will be our only functional dependency since there are no other dependencies, thus the closure of Facility_ID will be:

$\{Facility_ID\}^+ = \{Facility_ID, Facility_Name\} = \text{FACILITY}$

FACILITY

<u>Facility_ID</u>	<u>Facility_Name</u>

PATIENT:

In the **patient** relation the primary key (**ID**) determines every other entity so this will be our only functional dependency since there are no other dependencies, thus the closure of ID will be:

$\{ID\}^+ = \{ID, F.Name, M.Int, L.Name, PhoneNo, Sex, B.Date, B.Month, B.Year, City, Country, StreetName, D.ID, StartTime, Date, Prescription, Case\} = \text{PATIENT}$

PATIENT

<u>ID</u>	F.NAME	M.NAME	L.NAME	Phone No	Sex	B.Date	B.Month	B.Year	City	County	Street Name	<u>D.ID</u>	Start time	date	prescription	case

FACILITY_LOCATION:

There aren't any functional dependencies that can be extracted from this relation

ROOMS:

{Room_No}⁺ = {Room_NO, Room_Type, Room_Status, F.ID} = ROOMS

<u>F.ID</u>	Room_Type	<u>Room_No</u>	Room_Status
-------------	-----------	----------------	-------------

LABS:

{F.ID}⁺ = { F.ID, Lab_Type, Lab_Service, Service_Fee } = LABS

<u>F.ID</u>	Lab_Type	Lab_Service	Service_Fee
-------------	----------	-------------	-------------

PHARMACY:

{Drug_ID}⁺ = {F.ID, Drug_ID, Drug_Name, Drug_Status, Drug_Price } = PHARMACY

<u>F.ID</u>	<u>Drug_ID</u>	Drug_Name	Drug_Status	Drug_Price
-------------	----------------	-----------	-------------	------------

INVENTORY:

{Inventory_ID}⁺ = { F.ID, Inventory_ID, Inventory_Kind, Inventory_Status} = INVENTORY

<u>F.ID</u>	Inventory_Kind	<u>Inventory_ID</u>	Inventory_Status
-------------	----------------	---------------------	------------------

For the relations the Functional Dependencies will be:

USES:

{Patient_ID, Lab_Type, Usage_Date}⁺ = { Patient_ID, Lab_Type, Usage_Date, Result} = USES

<u>Patient_ID</u>	<u>Lab_Type</u>	<u>Usage_Date</u>	Result
-------------------	-----------------	-------------------	--------

PURCHASES:

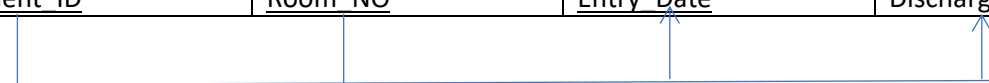
{Patient_ID, Drug_ID, Date}⁺ = { Patient_ID, Drug_ID, Date, Prescription_Detail} = PURCHASES

<u>Patient_ID</u>	<u>Drug_ID</u>	<u>Date</u>	Prescription_Detail
-------------------	----------------	-------------	---------------------

ASSIGNED IN:

$\{\text{Patient_ID}, \text{Room_No}, \text{Entry_Date}\}^+ = \{\text{Patient_ID}, \text{Room_No}, \text{Entry_Date}, \text{Discharge_Date}\} = \text{ASSIGNED IN}$

<u>Patient_ID</u>	<u>Room_NO</u>	<u>Entry_Date</u>	Discharge_Date
-------------------	----------------	-------------------	----------------

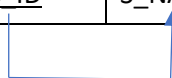


SPECIALIZATION:

In the specialization relation, the primary key (S_ID) determines every other entity so this will be our only functional dependency. Since there are no other dependencies, the closure of S_ID will be:

$\{\text{S_ID}\}^+ = \{\text{S_ID}, \text{S_Name}\} = \text{SPECIALIZATION}$

<u>S_ID</u>	S_NAME
-------------	--------



SPECIALIZES_IN:

There aren't any functional dependencies that can be extracted from this relation.

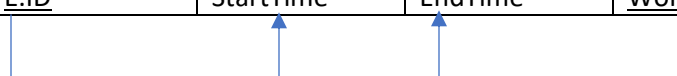
WORK_TIME:

From this relation one functional dependency can be extracted which is the union of **E.ID** and **WorkDay** determines the rest of the attributes, thus the closure of this union is:

$\{\text{E.ID}, \text{WorkDays}\}^+ = \{\text{E.ID}, \text{WorkDays}, \text{StartTime}, \text{EndTime}\} = \text{WORK_TIME}$

WORK_TIME

<u>E.ID</u>	StartTime	EndTime	<u>WorkDay</u>
-------------	-----------	---------	----------------



WORKS_ON:

There aren't any functional dependencies that can be extracted from this relation.

STATUS:

In the **STATUS** relation the primary key (StatusID) determines every other entity so this will be our only functional dependency since there are no other dependencies, thus the closure of StatusID will be:

$\{\text{StatusID}\}^+ = \{\text{StatusID}, \text{StatusLevel}, \text{ClearanceLevel}, \text{Salary}, \text{UnpaidNumberOfMonth}, \text{Paid}\} = \text{STATUS}$

STATUS

<u>StatusID</u>	StatusLevel	ClearanceLevel	Salary	UnpaidNumberOfMonth	Paid
-----------------	-------------	----------------	--------	---------------------	------

A diagram illustrating functional dependencies. A horizontal line starts from the 'StatusID' attribute and branches into five vertical arrows, each pointing to one of the other attributes: 'StatusLevel', 'ClearanceLevel', 'Salary', 'UnpaidNumberOfMonth', and 'Paid'.

MANAGES:

There aren't any functional dependencies that can be extracted from this relation.

Normalization

EMPLOYEE:

This relation is already normalized to BCNF form since there is only one functional dependency where the independent attributes are the super keys.

WORK_TIME:

This relation is already normalized to BCNF form, since there is only one functional dependency and the independent attributes are super keys.

WORKS_ON:

This relation is already normalized to BCNF form since there are no functional dependencies.

FACILITY:

This relation is already normalized to BCNF form since there is only one functional dependency where the independent attributes are the super keys.

Facility Location:

This relation is already normalized to BCNF form, since there is only one functional dependency and the independent attributes are super keys.

USES, PURCHASES, & ASSIGNED IN:

All this relations are already normalized to BCNF form since in all functional dependencies the independent attributes are super keys.

For the subclasses,

ROOMS, LABS, PHARMACY, & INVENTORY:

All this Entities are already normalized to BCNF form since in all functional dependencies the independent attributes are super keys.

PATIENT:

This relation is already normalized to BCNF form ,since there is only one functional dependency where the independent attributes are the super keys.

SPECIALIZES_IN

The relation is normalized to BCNF form since there are no functional dependencies.

SPECIALIZATION

This relation is normalized to BCNF form since there is only one functional dependency and the independent attribute is a super key.

STATUS:

This relation is already normalized to BCNF form, since there is only one functional dependency and the independent attributes are super keys.

MANAGES:

This relation is already normalized to BCNF form since there are no functional dependencies.