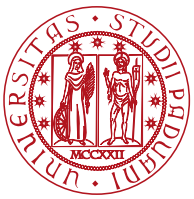

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Database Management Systems, A.Y. 2018/2019
Master Degree in Computer Engineering
Master Degree in Telecommunication Engineering

Homework 3 – Logical Design

Deadline: May 19, 2019

Group Winniest Team	Project Architecture Studio Management	studio ennedue e r c h t e t t i
Last Name	First Name	Student Number
Bardhi	Enkeleda	1194953
Boetto	Marco	1210407
Cattapan	Alessandro	1197597
Gallinaro	Giovanni	1210127
Gastaldello	Nicola	1206748
Terranova	Matteo	1202383

Variations to the Conceptual Design

We have made some changes on the previous ER-schema, in fact we merged some entities, which basically represent the same information. Hence, the entities Step, Activity and Job are all united in the entity Task in the new schema. Furthermore, in the previous schema the standard structure of the project was represented using different Catalogues, which in the new schema are substituted by the entity Template. Since the structure of the Project is representable as a tree, we have used a recursive relationship to define this tree and we added a new boolean attribute called IsRoot to keep track of the root of the tree. As the entity Activity is now represented in Task, we now use TimeSlot in order to keep track of the contribute of each employee in different fractions of time. Then, the ternary relationship Produce is now a binary relationship between Employee and Document and a new relationship called Generate is used to express that a specific document is produced during a certain task. Finally, we introduced a new derived attribute HoursSpent on the entity Project in order to improve the average performances of the database.

Transformation of the Entity-Relationship Schema

Figure 1 shows the Entity-Relationship schema and *Figure 2* shows the transformed Entity-Relationship schema. The following sections describe in detail how to obtain the transformed schema.

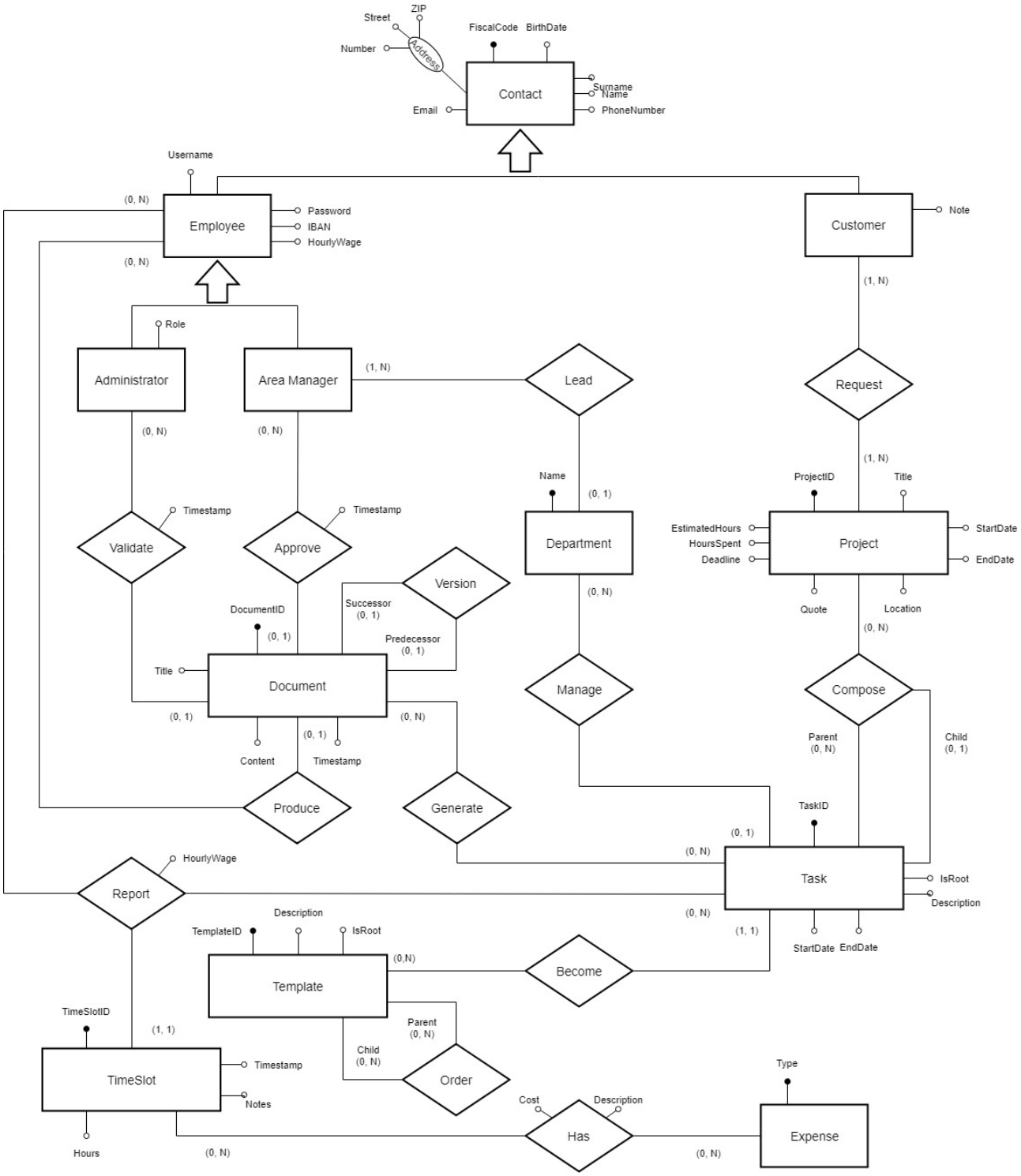

Figure 1 Entity Relationship Schema

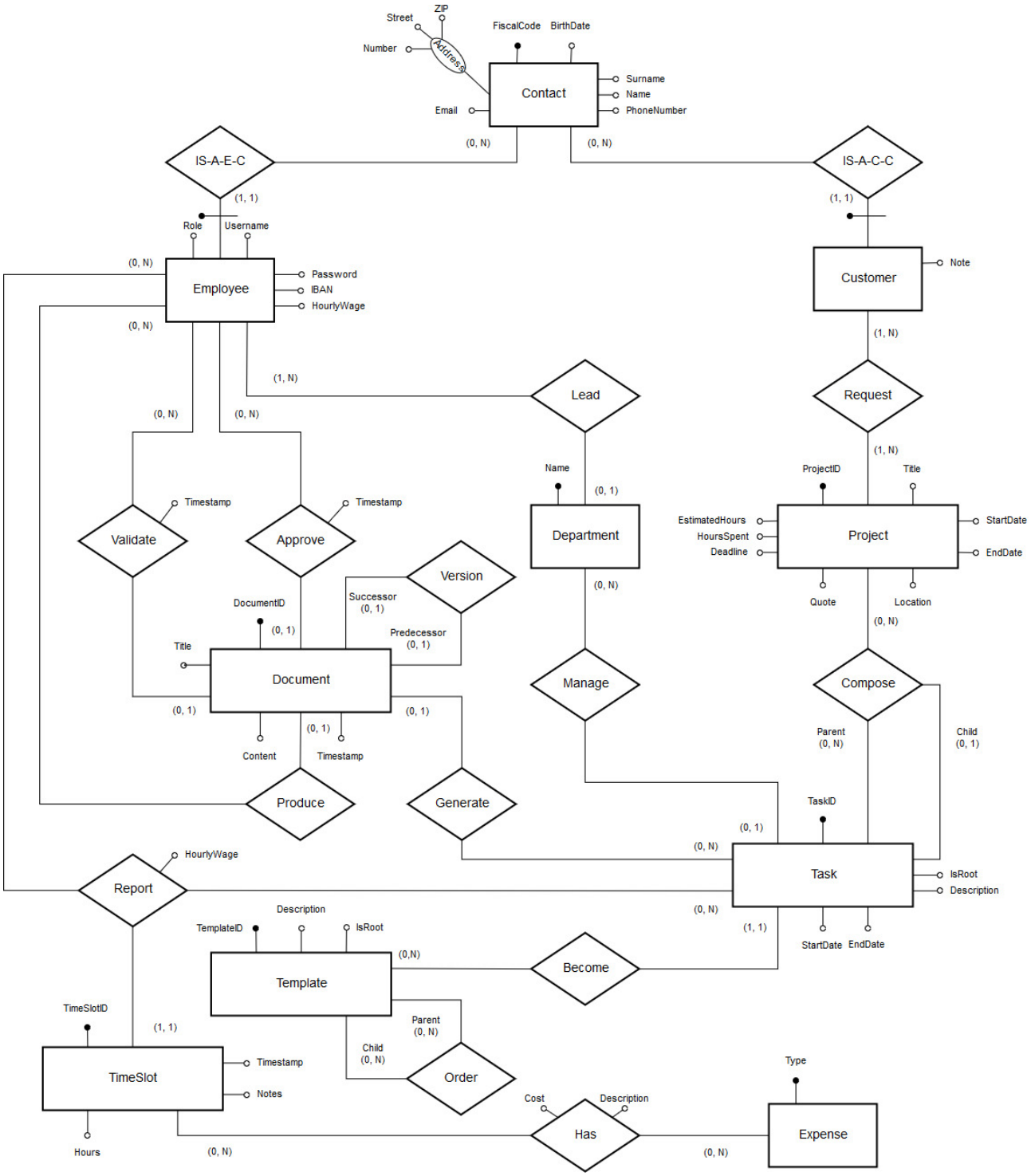

Figure 2 Transformed Entity Relationship Schema

1. Redundancy Analysis

Intensional redundancy: the schema does contain any intensional redundancy.

Extensional redundancy: the schema does not contain any cycle of entities, but it presents in the entity Task two attribute StartDate and EndDate that in some cases can be derived from other instances of the same entity Task. In fact with the entity Task is used to build the workflow of a project, which is a tree, so the value of the attribute StartDate and EndDate of a parent node, can be inferred from the attribute StartDate and EndDate of his child nodes. This kind of redundancy cannot be removed because of the way we have decided to describe the workflow of the projects.
Moreover the entity Project contains the derived attribute HoursSpent. We report below the analysis of the database load to check whether keeping this derived attribute or not.

2. Removal of Multi-Valued Attributes

The schema does not contain multi-valued attributes.

3. Removal of Composite Attributes

The only composite attribute present in the schema is the Address attribute of the Contact entity. Since this attribute has cardinality (1, 1), the component attributes are directly assigned to the Contact entity.

4. Removal of IS-A Relations and Generalizations

The schema contains one not-complete and not-disjoint generalization, represented by the entity Contact, and one not-complete and disjoint generalization represented by the entity Employee.
The case of Employee will be discussed first; since the generalization is not-complete only two out of three main solution are available:

- Solution 1: (*Figure 3*) Two new relationships IS-A-A-E and IS-A-AM-E are added between the entities Administrator and Employee and the entities AreaManager and Employee respectively.
- Solution 2: (*Figure 4*) The entities Administrator and AreaManager are merged into the entity Employee and consequentially receives the attribute Role from Administrator, which will be used to define the belonging to a subclass (regular employee, area manager, administrator or chief).

The best solution is the second one, since the attributes of both the entities are usually used together, in order to fully identify the employee associated to a particular data.
Moreover it only needs the addition of one attribute, that will never be NULL in any case.
The first solution will only introduce useless additional complexity, as it would create two more relationships and, as stated above, in order to give in output some statistics it would be necessary to reconstruct the whole data, navigating through multiple tables.
The entity Contact is a not-complete generalization, so also in this case only two solutions are available:

- Solution 1: (*Figure 5*) Two new relationships IS-A-E-C and IS-A-C-C are added between the entities Employee and Contact and the entities Customer and Contact respectively.
- Solution 2: (*Figure 6*) The entities Employee and Customer are merged within the entity Contact, which receives all the attributes of the removed entities.

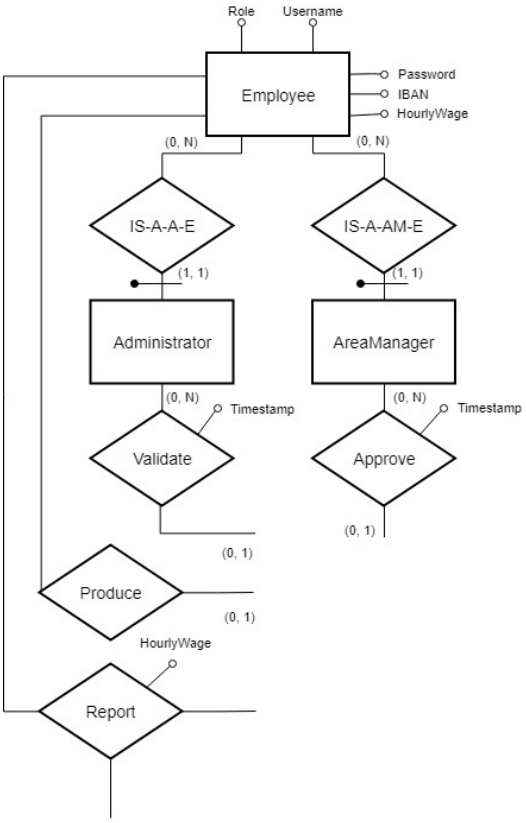

Figure 3 Generalization Removal Option 1

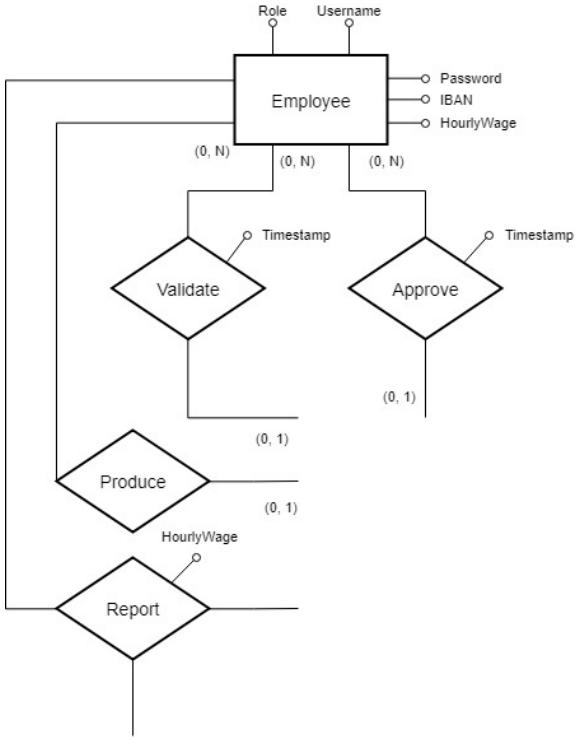

Figure 4 Generalization Removal Option 2

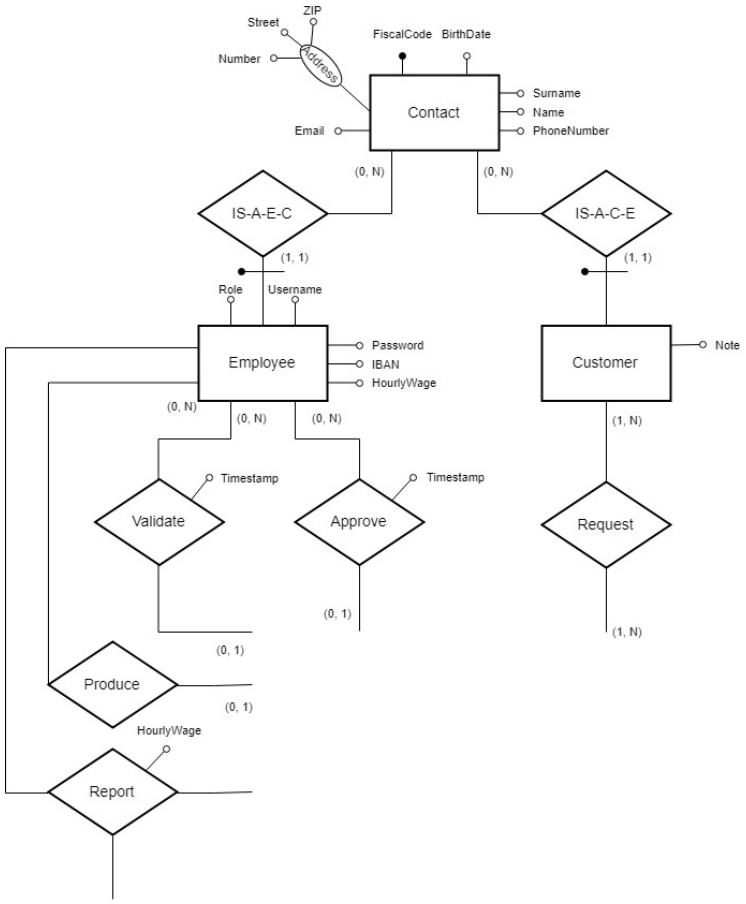

Figure 5 Generalization Removal Option 1

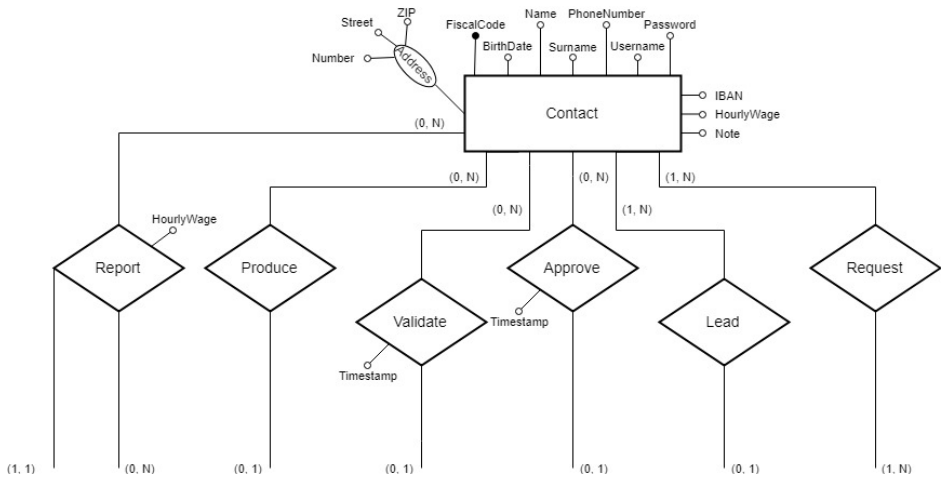

Figure 6 Generalization Removal Option 2

Among these two solutions, the most suitable for this system is the first one, since in that way we avoid systematic NULL values. In fact even though the data of the entity Contact and Employee or Customer are sometimes utilized together, they are also often utilized disjointly (exploiting the username in the case of employee), moreover they have their own specific attributes which in the long term will not be filled in, also adopting such solution we can better represent the contact who are neither customers or employees.
As stated above, the second solution causes the insertion of many systematic NULL values due to the fact that only in some rare cases an instance belongs to both subclasses.

5. Choice of Principal Identifiers

The schema does not contain external identification cycles.

6. Specification of Additional External Constraints

The following external constraints are added as a consequence of the removal of generalizations:

- The entity Employee can be in relation with the entity Document through the relationship Approve only if the Employee's Role attribute has value equal to AreaManager.
- The entity Employee can be in relation with the entity Document through the relationship Validate only if the Employee's Role attribute has value equal to Chief.
- The entity Employee can be in relation with the entity Department through the relationship Lead only if the Employee's Role attribute has value equal to AreaManager.

7. Variations to the Data Dictionary

The following section lists the variation to the entities and relationships tables of the data dictionary.

Entities Table

According to the changes made in the ER-schema the following entities were removed:

- Job
- Job Catalogue
- Step
- Step Catalogue
- Activity
- Activity Catalogue

Additionally, the entities that have been added are:

- Task
- Template
- TimeSlot

The following entities were modified:

Entity	Description	Attributes	Identifier
Contact	A person who interacts with the studio.	• FiscalCode, identifier of the contact, text; • Name, contact name, text; • Surname, contact surname, text; • BirthDate, contact birthdate,	FiscalCode (from contact, it is an external identifier)

		date; • Email, contact email, text; • PhoneNumber, contact phone number, text; • ZIP, the Zone Improvement Plan,text; • Number, the address number of the contact, text; • Street, the street where a Contact lives, text;	
Employee	Person who works in the Studio.	• Username, username of the employee, text; • Password, password used for the login, text; • IBAN, employee's IBAN, text; • HourlyWage, employee's wage per hour, float; • Role, distinction of the role, text;	FiscalCode (from contact, it is an external identifier)
Project	Set of one or more Tasks with the aim of satisfying the customers' request.	• ProjectID, identifier of the project, UUID. • Title, title of the project, text; • Quote, quote of the project, blob; • Location, location related to the project, text; • StartDate, the creation date of the project, date; • EndDate, the delivery date of the project, date; • Deadline, the deadline of the project, date; • EstimatedHours, the total estimated amount of hours needed to complete the project, float; • HoursSpent, the amount of hours spent on a project up to the current time, float;	ProjectID
Task	A single component that compose the actual workflow of the project.	• TaskID, the identifier of a specific task, UUID. • IsRoot, defines if a task is a root of the project's tree, boolean. • Description, a short description of the task, blob. • StartDate, the creation date of the Task, date. • EndDate, the delivery date of the Task, date.	TaskID
Template	Representation of the standard structure of every project.	• TemplateID,the identifier of a specific Template, UUID. • IsRoot, defines if a Template is root of the tree, boolean.	TemplateID

		• Description, a short description of the template, blob.	
TimeSlot	The contribute of each employee in different fractions of time.	• TimeSlotID, the identifier of a specific TimeSlot, serial. • Hours, amount of time spent by the employee in a specific TimeSlot, float. • TimeStamp, when a TimeSlot has been reported, date. • Notes, brief note about the TimeSlot to be reported, text.	TimeSlotID

Relationships Table

The following relationships were removed from the Entity-Relationship schema:

- Produce
- Receive
- Describe
- Instantiate
- Form
- Describe
- Create
- Follow

The following relationships were added to the Entity-Relationship schema:

- Generate
- Become

The following relationships were modified:

Relationship	Description	Component Entities	Attributes
Approve	Every output Document could be approved by an Employee.	• Employee (0, N); • Document (0, 1).	• Timestamp, datetime.
Become	The Tasks are selected from a pre-defined Template.	• Task (1, 1); • Template (0, N).	
Compose	A Project is composed by a certain number of Tasks which are organized in a tree structure.	• Project (0, N); • Task (Parent) (0,N); • Task (Child) (0,1).	
Generate	A Document can be generated in order to start a new Task and also as output of a certain Task.	• Task (0, N); • Document (0, N).	
Has	An Expense can be assigned to a specific TimeSlot to keep track of	• TimeSlot (0, N); • Expense (0, N).	• Cost, float. • Description, text.

	the reimbursement needed.		
IS-A-C-C	An Customer is a Contact since the Studio needs to store all the informations about the customers.	• Contact (1, 1); • Customer (0, N).	
IS-A-E-C	An Employee is a Contact since the Studio needs to store all the informations about the employees.	• Contact (1, 1); • Employee (0, N).	
Manage	Each Task is managed by a Department.	• Task (0,1); • Department (0, N).	
Order	The Template follows a tree structure in which all the nodes are ordered between each other.	• Order (Parent) (0, N); • Order (Child) (0, N)	
Produce	An Employee can create a new Document.	• Employee (0, N); • Document (0,1).	
Report	An Employee can report the number of hours spent on a certain Task.	• Employee (0, N); • TimeSlot (1, 1); • Task (0, N).	• HourlyWage, employee's hourly wage, float.
Validate	Every output Document could be approved by an Employee.	• Employee (0, N); • Document(0, 1).	• Timestamp, datetime.

Analysis of Database Load [mandatory]

In this section, we report the analysis of the database to justify the presence of redundancies in the entity relationship schema. Consider the following two example operations that involve the redundant attribute Objective Data:

- O1: report the hours spent on a task;
- O2: print data about a project, together with the real number of hours spent on it.

Table 1 reports the description of each operation, its frequency and type. Both O1 and O2 are online operations since an employee need to store the hours spent on a task when he has done them and the real hours spent on a project need to be retrieved on the fly. The operation O1 is made on average twice a day by every employee, who are around fifteen.

Operation	Description	Frequency	Type
O1 Report the hours	store the hours spent on a task by an employee	30/day	Online
O2 Print real hours spent on the project	print all data about a project, together with the real number of hours spent on it	1/day	Online

Table 1 Frequency Table

Table 2 reports the access/volume data related to operation O1 with redundancy.

Concept	Construct	Access	Type	Average Access

Report	Relationship	1	W	1 x 30 x 2 = 60
TimeSlot	Entity	1	W	1 x 30 x 2 = 60
Task	Entity	1	R	1 x 30 x 1 = 30
Compose	Relationship	1	R	1 x 30 x 1 = 30
Project	Entity	1	R	1 x 30 x 1 = 30
Project	Entity	1	W	1 x 30 x 2 = 60
Total Access				270

Table 2 Access/Volume Table for Operation O1 with redundancy

Table 3 reports the access/volume data related to operation O1 without redundancy.

Concept	Construct	Access	Type	Average Access
Report	Relationship	1	W	1 x 30 x 2 = 60
TimeSlot	Entity	1	W	1 x 30 x 2 = 60
Total Access				120

Table 3 Access/Volume Table for Operation O1 without redundancy

The difference between Table 2 and 3 is the presence (or not) of the attribute HoursSpent on the entity Project.
Table 4 represents the access/volume data related to operation O2 with redundancy.

Concept	Construct	Access	Type	Average Access
Project	Entity	1	R	1 x 1 x 1 = 1
Total Access				1

Table 4 Access/Volume Table for Operation O2 with redundancy

Table 5 represents the access/volume data related to operation O2 without redundancy. It is important to notice that to count the real number of hours spent on a project we have to go through all the nodes and in particular to all the leaves of a project's tree. In fact only the leaves of the entity Task take part to the Report relationship. Every project has on average 1244 nodes (we are assuming that there are 4 job, which have 10 step each, which at their time have 30 activity each), of which 1200 leaves, and every leaf takes part to the relationship Report once. This last estimated number is strongly dependent on how many times an employee reports the hours for the same Step and on how much time requires a single step.

Concept	Construct	Access	Type	Average Access
Project	Entity	1	R	1 x 1 x 1 = 1
Compose	Relationship	1244	R	1244 x 1 x 1 = 1244
Task	Entity	1200	R	1200 x 1 x 1 = 1200
Report	Relationship	1200	R	1200 x 1 x 1 = 1200
TimeSlot	Entity	1200	R	1200 x 1 x 1 = 1200
Total Access				4845

Table 5 Access/Volume Table for Operation O1 without redundancy

Finally, Table 6 reports the summary of the total accesses per day, showing that the redundancy decreases the number of access from 4965 to 271, which represent a decrease of 95%.

Operation	With Redundancy	Without Redundancy
O1	270	120
O2	1	4845
Total Access/Day	271	4965

Table 6 Comparison of the Number of Accesses for each Operation

Relational Schema

Figure 7 shows the relational schema.
For the many entities in the schema that participate to one-to-many relationships with optional participation, we have made the following decisions:

- Document and Employee are related through the one-to-many relationship Produce. In particular Document is related with an optional participation (0,1). Over all the documents that have to be stored in the database, only a few of them are not produced by employees, therefore, only a few of them would have NULL values in the attribute which relates them to their producer. As a consequence we decided to map such entity into the attribute Producer of Document.
- Document and Employee are related with other two one-to-many relationship: Approve and Validate. In particular Document has an optional partecipation with constraint (0,1) in both the relationships. Moreover Approve and Validate have a similar structure and the observations and decisions taken for one of them is also valid for the other.

Considering a long term deployment of the database, the amount of documents that are not approved or validated is not negligible, therefore we decided to keep the relationship tables separated, in order to avoid systematic NULL values.

- The entity Document also participates to a recursive one-to-one relationship, i.e. Version, with optional participation (0,1) for the versions of the Document. In this case not every document requires an updated version, which would lead to several NULL values, therefore we decided to maintain a table for Version.
- Document and Task are related through the one-to-many relationship Generate, where Document has an optional participation (0,1). In this case almost all the tasks are supposed to produce an output document, therefore we decided to merge the table Generate into Document.
- Department and Task are related through the one-to-many relationship Manage, where Task has an optional participation (0,1). As every task has to be managed by a department, we decided to merge the relationship Manage into Task.
- Compose is a ternary relationship, therefore even if there is an optional participation, all the tables are maintained distinct.
- Department and Employee are related through the one-to-many relationship Lead, where optional participation (0,1). Because in the long term each department is supposed to be led by an area manager, all the NULL values will be filled in, therefore we decided to merge the relationship Lead into Department.

All the other relationships are transformed straightforwardly.

Normalization of the Relational Schema

The relational schema represented in Figure 6 is normalized up to the Boyce-Codd normal form:

- First Normal Form: the absence of composite attributes and the removal of multivalued attributes guarantee that the schema satisfies the requirements of 1NF.
- Second Normal Form: all the relations of the schema that have a primary key constituted of a single attribute are in 2NF. There are some tables derived from relationships in the ER schema that have a primary key with two or three attributes, however there are no functional dependencies between the key and any other attribute in the relations, therefore these relations are in 2NF. Finally, we can conclude that the relational schema is in 2NF;
- Boyce-Codd and Third Normal Form: there are no functional dependencies between the attributes of each relation, therefore we can state that the schema is in BCNF (there are not any non prime attribute which is transitively dependent on a primary key). Besides, since BCNF is stricter than 3NF, the relation schema is also in 3NF.

Data Dictionary

Relation	Attribute	Description	Domain	Constraints
Approve	DocumentID	Identifier of the document	UUID	Foreign key to Document, Not NULL, primary key with FiscalCode
	FiscalCode	Employee's identifier	Text	Foreign key to Employee, Not NULL, primary key with DocumentID
	TimeStamp	When a document has been approved	Datetime	Not NULL
Compose	TaskID_Parent	Identifier of the parent task	UUID	Foreign key to Task, Not NULL, primary key with TaskID:Child and ProjectID
	TaskID:Child	Identifier of the child task	UUID	Foreign key to Task, Not NULL, primary key with TaskID:Parent and ProjectID
	ProjectID	Identifier of the project	UUID	Foreign key to Project, Not NULL, primary key with TaskID:Parent and TaskID:Child
Contact	FiscalCode	Contact identifier	Text	Primary key
	Name	Contact's name	Text	Not NULL
	Surname	Contact's surname	Text	Not NULL
	BirthDate	Contact's birthday	Date	
	Email	Contact email	Text	
	PhoneNumber	Contact's phone number	Text	Not NULL
	ZIP	Postal Code	Text	Not NULL
	Street	Contact street	Text	Not NULL
	Number	Street's number	Text	Not NULL
Customer	FiscalCode	Customer's identifier	Text	Primary key
	Note	Annotation about the customer	Text	
Department	Name	Department name	Text	Primary key
	FiscalCode	Employee's identifier	Text	Not NULL
Document	DocumentID	Identifier of the document	UUID	Primary key
	Title	Title of the document	Text	Not NULL
	Content	Content of the	Blob	Not NULL

		document		
	TimeStamp	When a document has been received or created	Datetime	Not NULL
Employee	FiscalCode	Employee's identifier	Text	Primary key
	Username	Name used for log-in	Text	Not NULL
	Password	Password used for log-in	Text	Not NULL
	IBAN	Employee's IBAN	Text	Not NULL
	HourlyWage	Employee's wage per hour	Float	Not NULL
	Role	Distinction of the role within the studio	Text	Not NULL
Expense	Type	Type of expense incurred by an employee that has to be reimbursed	Text	Primary key
Generate	TaskID	Identifier of the task	UUID	Foreign key to Task, Not NULL, primary key with DocumentID
	DocumentID	Identifier of the document	UUID	Foreign key to Document, Not NULL, primary key with TaskID
Has	TimeSlotID	Identifier of the timeslot	Serial	Foreign key to TimeSlot, Not NULL, primary key with Type
	Type	Type of expense incurred by an employee that has to be reimbursed	Text	Foreign key to Expense, Not NULL, primary key with TimeSlotID
	Cost	Cost of the timeslot	Float	Not NULL
	Description	Description of the timeslot	Text	
Order	TemplateID:Child	Identifier of the child template	UUID	Foreign key to Template, Not NULL, primary key with TemplateID:Parent
	TemplateID:Parent	Identifier of the parent template	UUID	Foreign key to Template, Not NULL, primary key with TemplateID:Child
Produce	DocumentID	Identifier of the document	UUID	Foreign key to Document, Not NULL, primary key with FiscalCode
	FiscalCode	Employee's identifier	Text	Foreign key to Employee, Not NULL, primary key with DocumentID
Project	ProjectID	Identifier of the project	UUID	Primary key
	Title	Title of the project	Text	Not NULL
	StartDate	Start date of the project	Date	Not NULL
	EndDate	End date of the project	Date	
	Location	Location of the project	Text	Not NULL

	Quote	Quote of the project	Blob	
	Deadline	Deadline of the project	Date	Not NULL
	EstimatedHours	Estimated hours of the project	Float	Not NULL
Request	FiscalCode	Customer's identifier	Text	Foreign key to Customer, Not NULL, primary key with ProjectID
	ProjectID	Identifier of the project	UUID	Foreign key to Project, Not NULL, primary key with FiscalCode
Task	TaskID	Identifier of the task	UUID	Primary key
	IsRoot	Determines if the task is a root of the tree	Boolean	Not NULL
	Description	Description of the task	Text	
	StartDate	Start date of the task	Date	Not NULL
	EndDate	End date of the task	Date	
	TemplateID	Identifier of the template	UUID	Not NULL
	Name	Name of the task	Text	Not NULL
Template	TemplateID	Identifier of the template	UUID	Primary key
	Description	Description of the template	Text	
	IsRoot	Determines if the template is a root of the tree	Boolean	Not NULL
TimeSlot	TimeSlotID	Identifier of the timeslot	Serial	Primary key
	TaskID	Identifier of the task	UUID	Primary key
	TimeStamp	When a timeslot has been reported	Datetime	Not NULL
	Notes	Notes of the timeslot	Text	
	HourlyWage	Wage per hour spent by an employee in the timeslot	Float	Not NULL
	FiscalCode	Employee's identifier	Text	Not NULL
	Hours	Time spent by the employee in the specific timeslot	Float	Not NULL
Validate	DocumentID	Identifier of the document	UUID	Foreign key to Document, Not NULL, primary key with FiscalCode
	FiscalCode	Employee's identifier	Text	Foreign key to Employee, Not NULL, primary key with DocumentID
	TimeStamp	When a document has been validated	Datetime	Not NULL
Version	DocumentID: Predecessor	Identifier of the document predecessor	UUID	Foreign key to Document, Not NULL, primary key with DocumentID:

				Successor
	DocumentID: Succecessor	Identifier of the document successor	UUID	Foreign key to Document, Not NULL, primary key with DocumentID: Predecessor

External Constraints

1. When a final document is uploaded by the Employee, it is assumed that it was already verified by the same Employee. So the Timestamp of a document corresponds to the verification date of the document.
2. The final documents can be approved by the Employee whenever its attribute Role has value AreaManager, while they can be validated only when the attribute Role of Employee has value Chief.
3. The Employee can Lead a Department only if its attribute Role has value AreaManager.
4. Before being validated, the documents must be approved.
5. Every user of the application has different permissions:
    - An Employee can access only the activities where he is involved in;
    - An Employee with Role AreaManager can have access to all the data of the jobs related to the departments he leads. Furthermore, he can create and add a new job to a project.
    - Employee with Role Chief and Administrator can create a new project, access all the projects' related data, establish the hourly wage of each employee and manage contacts' data.
6. Make sure that all the tasks are executed in the correct order by the Employee.

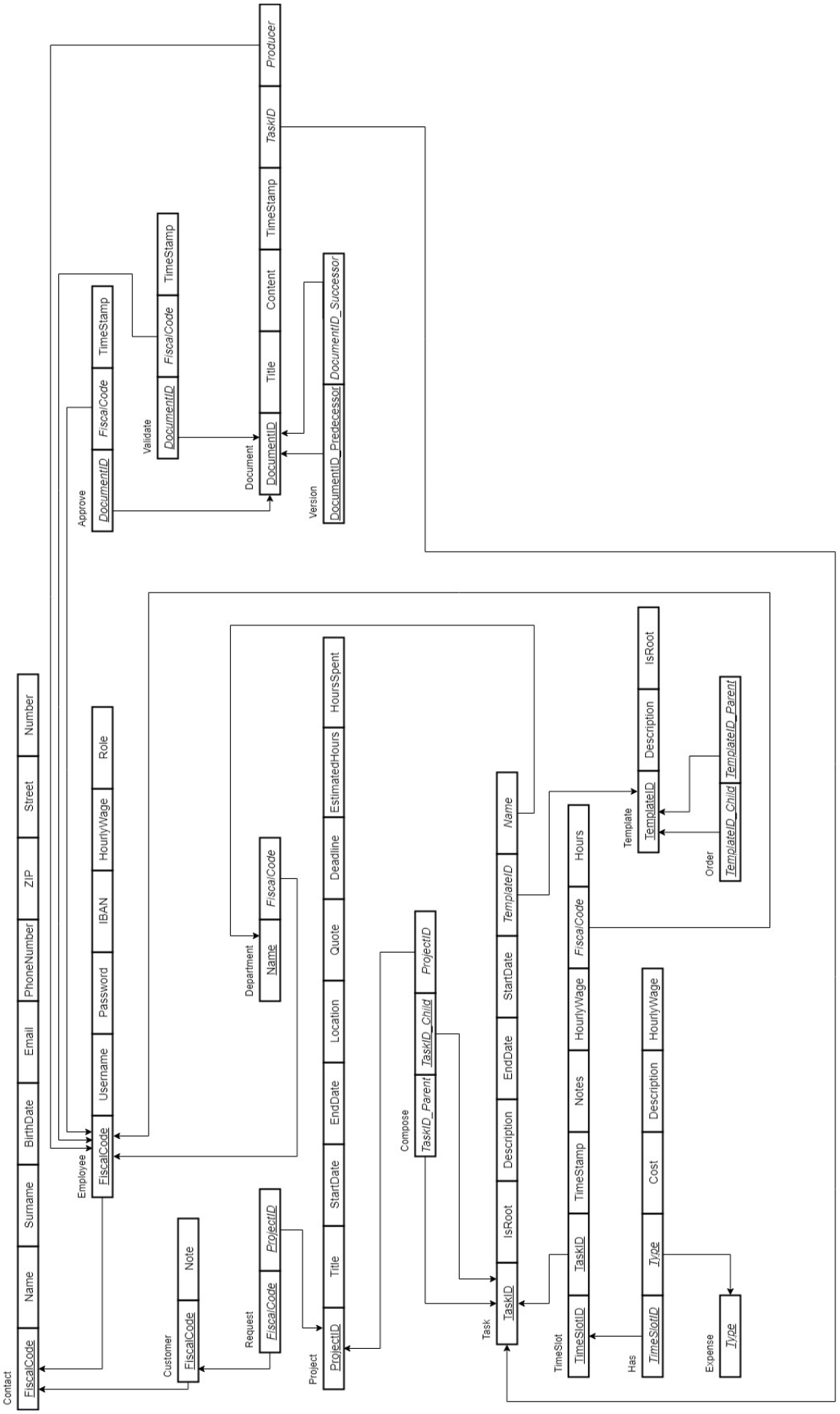

Figure 7 Relational Schema