Czech University of Life Sciences Prague

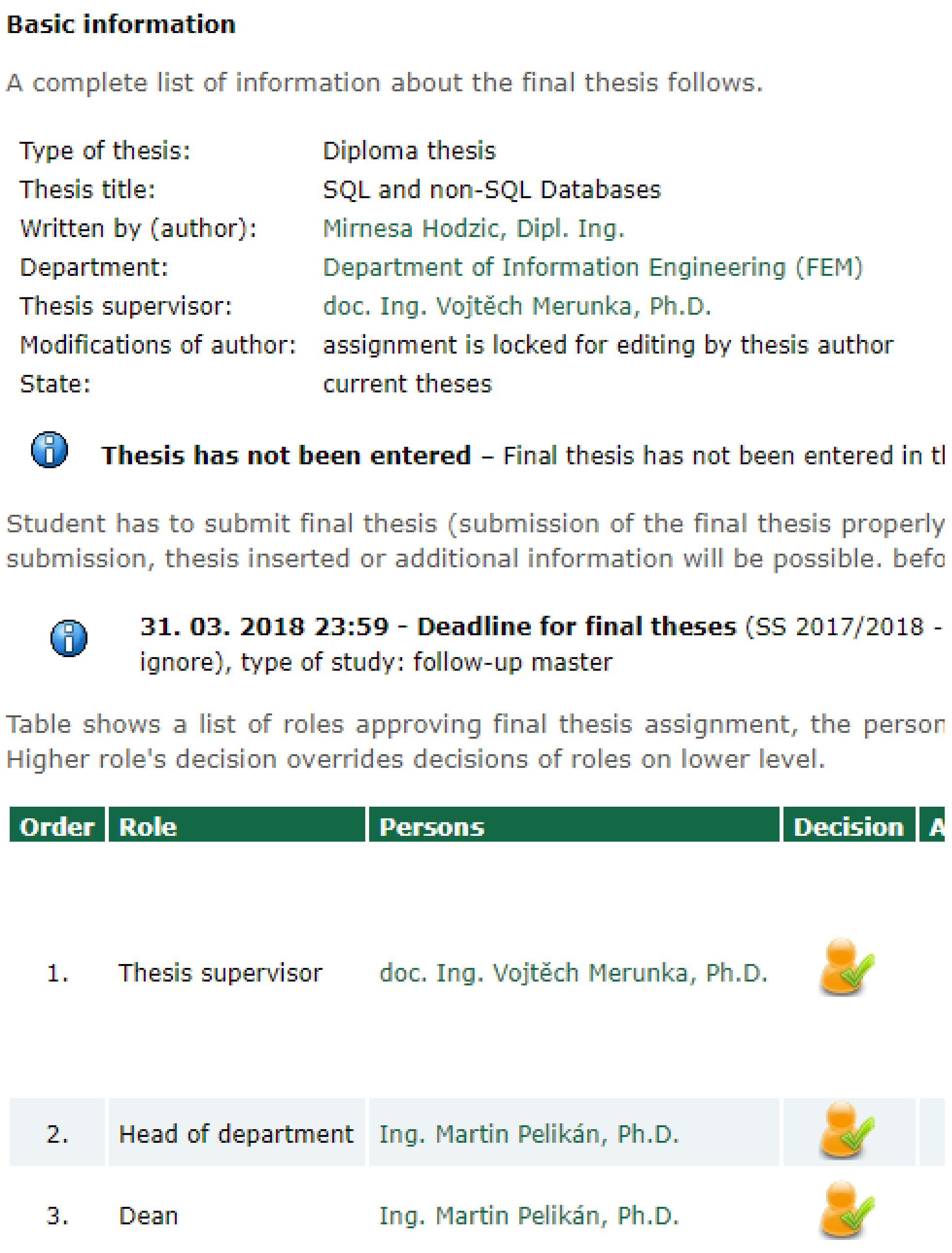
Faculty of Economics and Management

Department of Information Engineering

Diploma Thesis

SQL and NoSQL database

Mirnesa Hodzic



Basic information

A complete list of information about the final thesis follows.

Type of thesis:	Diploma thesis
Thesis title:	SQL and non-SQL Databases
Written by (author):	Mirnesa Hodzic, Dipl. Ing.
Department:	Department of Information Engineering (FEM)
Thesis supervisor:	doc. Ing. Vojtěch Merunka, Ph.D.
Modifications of author:	assignment is locked for editing by thesis author
State:	current theses

Thesis has not been entered – Final thesis has not been entered in tl

Student has to submit final thesis (submission of the final thesis properly
submission, thesis inserted or additional information will be possible. befo

31. 03. 2018 23:59 - Deadline for final theses (SS 2017/2018 -
ignore), type of study: follow-up master

Table shows a list of roles approving final thesis assignment, the person
Higher role's decision overrides decisions of roles on lower level.

Order	Role	Persons	Decision	A
1.	Thesis supervisor	doc. Ing. Vojtěch Merunka, Ph.D.	✓	
2.	Head of department	Ing. Martin Pelikán, Ph.D.	✓	
3.	Dean	Ing. Martin Pelikán, Ph.D.	✓	

! ! !

The back page of thesis assignment insert instead of this text.

In the case your thesis assignment is longer than 2 pages, insert he other pages too.

! ! !

Declaration

I declare that I have worked on my diploma thesis titled "SQL and NoSQL database" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the diploma thesis, I declare that the thesis does not break copyrights of any their person.

In Prague on 31/03/2018 ______________________________

Acknowledgement

I would lid to thank Diploma Thesis supervisor doc. Ing. Vojtěch Merunka, Ph.D., for his advice and support during my work on this thesis.

SQL a NoSQL databaze

Souhrn

SQL a NoSQL databáze jsou nyní propojovány tak, aby bylo možné snadněji přecházet z jednoho prostředí do druhého. Struktura dat tak představuje hlavní faktor, který je třeba brát v úvahu, aby bylo možné toto propojení a snadný přechod z jednoho prostředí do druhého plně využít.

Výhody NoSQL databází jsou již dobře známy. Problém představuje rozhodnutí, jak naložit s již existujícím prostředím. V závislosti na druhu činnosti podporované databázovým prostředím a cílech této činnosti, rozhodnutí má klíčový dopad na množství spotřebovaných finančních prostředků i času. Problém spočívá v tom, že všechny možnosti poskytované relačním prostředím jsou k dispozici i pro NoSQL databáze. Pokud pracujeme s lidmi s různými návyky, hlavním faktorem je konzistence dat.
Pro vylepšení a širší rozšíření NoSQL, struktura dat a přesun do relačního prostředí mohou tento problém vyřešit. Pokud převezmeme prostředky z relačního prostředí – normalizaci databáze, můžeme vytvořit model, který bude vhodný pro každou databázi.

Tato práce podává krátký přehled databází, architektur databází, modelování, jazyků, které lze použít pro práci s databází a možností jak přesunout NoSQL do relačního prostředí. Business Process Model je použit k popisu základních operací při každodenní práci s databází.

Klíčová slova: SQL, NoSQL, business process model, konceptuální diagram, JSON, databázová architektura, datová struktura, relační prostředí, diagram tříd, modelování databáze

SQL and NoSQL database

Summary

The SQL database and NoSQL database are used to make relationship between them as possibilities easier movement from one environment to another. In that way data structure represent main factor what can be consider to implement that possibilities.
Advantage of using NoSQL database are already known, so, now is problem to make decision what to do with already existing environment. Depends of our current business and goal, decision of choice will make key about waste time and money. Everything what is given by relational environment can be also be done using NoSQL database. What can be problem. Problem is consistency of data as main factor when we are working with people with different habits. To improve NoSQL and make it used everywhere, data structure and move NoSQL into relational environment can help to solve that problem. If we are able catch the same thing what are used in relational environment as normalisation of database, than we are able make model what can be applicable for every database.
The paper gives as short view of database, architecture of database, modelling, languages what can be used for work with database, and possibilities how to move NoSQL into relational environment. Also, business process model is used to describe basic operations what we are using every day to work with database.

Keywords: SQL, NoSQL, business process model, conceptual diagram, JSON, architecture of database, data structure, relational environment, class diagram, database modelling

Table of content

1 Introduction 12

2 Objectives and Methodology 13

2.1 Objectives 13

2.2 Methodology 13

3 Literature Review 14

3.1 Introduction, Definition and Type of Database 14

3.1.1 Database and their Purpose 16

3.1.2 Modelling the Database 17

3.1.3 Database Architecture 18

3.1.4 Languages for Work with Database 19

3.1.5 Life Cycle of the Database 20

3.2 Modelling the Database 21

3.2.1 Entities, Links and Attributes 21

3.2.2 Languages for Work with Database 22

3.2.3 The Relational Database 23

3.2.4 Converting the ER shema into Rational 24

3.2.5 Relational Model, Network and Hierarchical Model 25

3.2.6 Normalization of the Database 25

3.3 Database Architecture 27

3.3.1 N-tier Architecture 27

3.3.2 Data Layer or Database Layer 29

3.3.3 MySQL Architecture 29

3.3.4 Transaction Management 31

3.3.5 NoSQL Architecture 33

3.3.5 DB2 Architecture 36

4 Practical Part 38

4.1 Bussiness Architecture 38

4.2 Bussiness Process Modelling 39

4.3 Conceptual Model 46

5 Results and Discussion 53

5.1 Bussiness Architecture 53

5.2 Bussiness Process Modelling 53

5.2 Conceptual Model 61

6 Conclusion 63

7 References 64

8 Appendix 65

List of pictures

Figure 3.1.1. The Relacional Database ---14
Figure 3.1.2. The Network Database --14
Figure 3.1.3. The Hierarchical model ---15
Figure 3.1.4. Object model database --15
Figure 3.1.3.1. Database levels --18
Figure 3.1.5.1. Database life cycle ---19
Figure 3.2.2.1. Tool for modelling ---22
Figure 3.2.3. Relational database_exampleBOOK ---23
Figure 3.2.6.1. No-normal table of database ---26
Figure 3.2.6.2. 1NF table of database --26
Figure 3.3.1.3. 2-tier Architecture --26
Figure 3.3.2.1. Layers ---28
Figure 3.3.3.1. MySQL Architecture ---29
Figure 3.3.5.2. Example of NoSQL Storage --34
Figure 3.3.5.3. Service/ Middle Tier/ Application Layer -------------------------------------34
Figure 3.3.6.1. DB2 Server --36
Figure 4.1.1. Architecture of volunteer program ---38
Figure 4.2.1. Business process model – first step communications --------------------------40
Figure 4.2.2. Business process model – add into database --------- --------------------------41
Figure 4.2.3. Business process model – Program Activities ----------------------------------42
Figure 4.2.4. Business process model – Program Activities ---------------------------------43
Figure 4.2.5. Business process model – Report --44
Figure 4.2.6. Business process model – Program ---45
Figure 4.3.1. Logical view of NoSQL --46
Figure 4.3.2. Conceptual diagram --47
Figure 5.2.1. Business Process Model – Decision_1 ---54
Figure 5.2.2. Business Process Model – Decision_2 ---54
Figure 5.2.3. Business Process Model – Decision_3 ---55
Figure 5.2.4. Business Process Model – Decision_4 ---55
Figure 5.2.5. Business Process Model – Decision_5 ---56
Figure 5.2.6. Business Process Model – Decision_6 ---56
Figure 5.2.7. Business Process Model – Report_1 --57
Figure 5.2.8. Business Process Model – Report_2 --57
Figure 5.2.9. Business Process Model – Report_3 --57
Figure 5.2.10. Business Process Model – Report_4 --58
Figure 5.2.11. Business Process Model – Report_5 --58
Figure 5.2.12. Business Process Model – Relationship_1 -----------------------------------59
Figure 5.2.13. Business Process Model – Relationship_2 -----------------------------------59
Figure 5.2.14. Business Process Model – Relationship_3 -----------------------------------60
Figure 5.2.15. Business Process Model – Relationship_4 -----------------------------------60

Figure 5.2.16. Business Process Model – Report_5 --60
Figure 5.2.17. Business Process Model – Report_5 --60
Figure 5.2.18. Business Process Model – Report_5 --61

Figure 5.3.1. Code JSON --62
Figure 5.3.2. SQL table ---62

1 Introduction

The Diploma thesis is based on the problem how to non relational database put into relational environment. The model what can be used and be applicable for every database still doesn’t exist, so, for now we are looking tor how to find the best solution what is unique for all storages. Currently, it is possible based on data make both NoSQL and SQL database in similar way where we are able to connect similarities and in that way exististing relational database shift to NoSQL database or opposite side.

“Financial Service Volunteer Corp” what is taken as example of business where stored data can be useful for next work but also, there is no prediction of next possibilities in the field of activities, because every country has own law and regulations so, only some data as template can help to see benefit of that instructions and steps. For that reason, even the business is just focus on finance (there is no shops every day) , is necessary think that some result can be anyway be applicable for future but with some modification. The modification require changes in relationships database, so , to skip waste time and money, organisation should operate with NoSQL to be able store and modify all changes what can be came during activities.

2 Objectives and Methodology

2.1 Objectives

The database are one the most used IT invention. Every action and reaction are going throw database, every web order, registration have to stored somewhere. So, the database is common in every electronic transaction and communication. The goal of the thesis is make comparison between SQL and NoSQL database, pros and cons and why to skip one and choose another one, which reason can push us to use both of them. I used one organisation what we are able to see one the web page: https://www.fsvc.org/adam-szubin-and-eunice-panetta-elected-to-fsvcs-board-of-directors/

It is about "Financial Service Volunteer Corp", because based on their business and activities, I am thinking that they should have the database to store and manipulate with all data what can help them as backup to save money and time. The thesis is more focused on actions, processes and data structure what are daily used.

2.2 Methodology

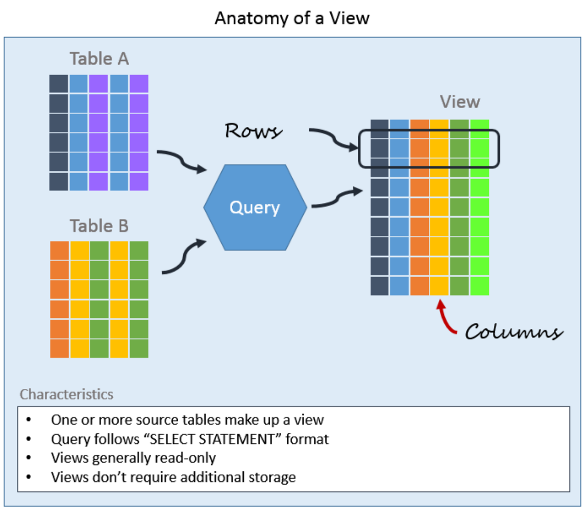
The diploma thesis is implemented by using knowledge of SQL database, JSON as language for NoSQL database and Craft.Case. The SQL and JSON are given in wide description in pages below. CraftCase is business analysis tool what is used to express model of the business. This is way to define and visualize to see how it will be communication between some stages happed and how some decision have impact on whole process and result. Also, during analysis, we are able use some advantages as option "Check and highlight invalid links " and " Highlight elements not implemented in conceptual" what provides checking relationships what are not necessary, errors and so on.
It is important to mentioned that behind CraftCase modelling tool is mathematical background and all fuctionality including simulation is based on mashine state theory and CraftCase model is object oriented. So, that is reason why is used in database modelling.

3 Literature Review

3.1 Introduction, definition and type of database

The database represents a software solution that serves to store data in a way that enables later analysis of data, easier browsing, connection to related data and what are interconnected. Data stored in the database are available to a larger number of users, applications, thus increasing productivity, quality and accuracy in order to further search and analyze.
The database is represented by a set of rules that determine how a single entity can look like a data model. Depending on the set of rules, we have several database models:

- Relational database model where the data and connection between them are shown in the table below



- One or more source tables make up a view
- Query follows "SELECT STATEMENT" format
- Views generally read-only
- Views don't require additional storage

Figure 3.1.1. The Relacional Database

- A network database model where the database is represented by a graph (nodes are types of records, and arcs define the relationship between the data)

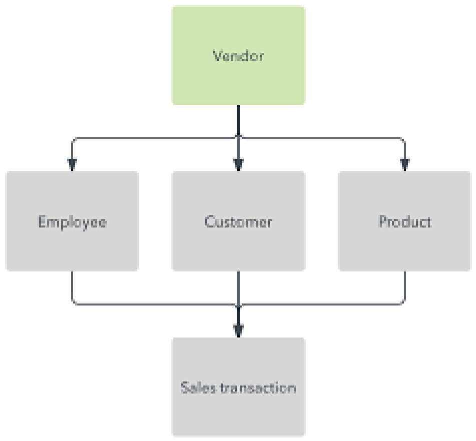


Figure 3.1.2. The Network Database

- A hierarchical model that represents a special case of a network model. The difference in classical network type is reflected in the fact that the model is presented in a hierarchy tree or group of trees.

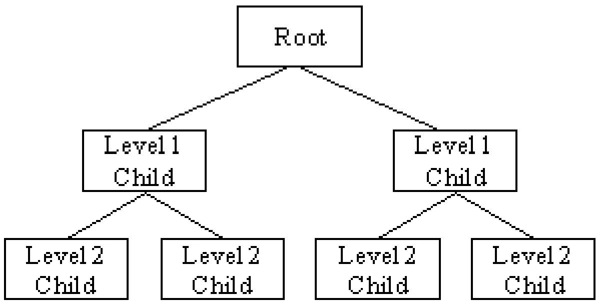


Figure 3.1.3. The Hierarchical model

- Object model representing the database model of data created using object-oriented programming languages. Here, methods of object-oriented languages are used, such as aggregation, inheritance and each object belongs to a class.

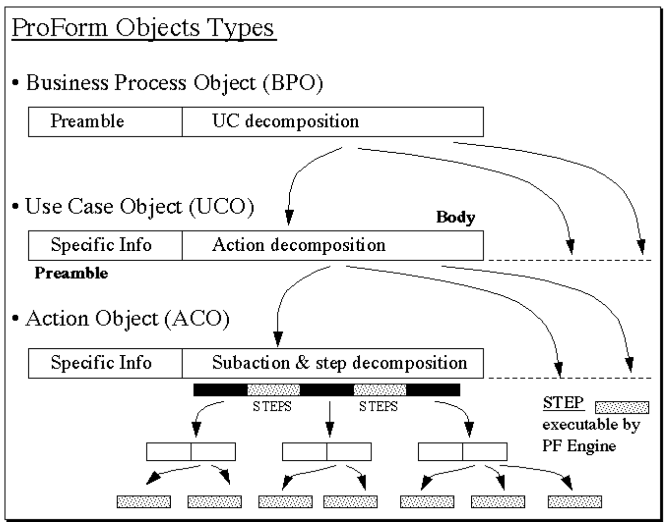


Figure 3.1.4. Object model database

3.1.1 Databases and their purpose

As mentioned above, the database represents a data group that carries value in order to further process (save data, manipulations with data, and analysis the data)
Physical independence of data; allows you to copy data to another file on other disks without requiring any changes in existing applications.

Logic independence of data; allows logical data changes without requiring any changes to existing applications by setting new relationships (record and connection).

Flexibility of access to data; today the user can access the data how he wants, however, before he has accessed the data, he has been defined in advance by the relationships that were given by the database designer.

Availability; allows more users to access data at the same time in different ways, we know from the user's wish. Databases by their nature are an all-routine logical structure. An example of the many-users purpose of the database is the database in transport companies where one database is accessibly to more users, and also applications with geographically remote locations are accessibly to see the availability of vacant seats on the plane, train or bus. Purchase / reservation is automatically shown to other users of the same database, regardless of where they were.

Keep integrity; it is possible to safeguard the correctness and consistency of the data. Signs, mistakes that can arise in case of needless use by the user ie. Incorrect data entry or errors resulting from misprinting the application may affect the integrity and consistency of the complete database. In order to preserve database data, the database designer defines the restriction (constraints). These limitations are preceded by rules which the data must satisfy to be considered valid. One of the examples can be considered; "Learner" or "worker" if there is an "age" attribute, the attribute "age" must contain the value of the value "INT" the 10 to 60. Similarly, the attribute "name" should have the string "string" not INt and so similar . Also, the other way to preserve the integrity of the database is to use in the relations of the "key" that we will talk about later.

Data security; the data is stored in the case that a hardware or system error occurs in the operation of the system software. Also, there must be protection from unauthorized use that is achieved by limiting the rights of users. Pre-defined user rights and their access to the data.

Data control; enables changing and configuring database performance, customer service. The responsible person in charge of updating, modifying and regulating is called the database administrator.

3.1.2 Modelling the database

Modelling of the database represent one of the most important part during decision which database we are going to use. Depends on our nature of the business, our goals, we have to make proper decision where to invest our money and what to do with data what are going to stored in database. Performance, availability, storage, costs are one of important part, but always is forgotten model of the data. Currently, we are manipulate with not to much data, but in future we are going to expend our business and invest in digitalisation of the process of our work. Also, one mainly factor is make our data independent and save. When we speak about independent data it means that data it doesn’t metter on which kind situation should be always be consistent. The variables what describe our objects should be implemented in that way that we don’t feel mess during our actions. The Entities, links and attributes should be carefully consider and optimized to perform our job easier and faster.

3.1.3 Database Architecture

Database architecture is provided in three layers; physical layer, global logic layer and local logical layer.

The physical level of the database is a physical display and the data schedule in the external memory. This database level is visible and accessible only to system developers who have developed DBMS.

The global logical database layer represents the logical structure of the whole database. This level of database is visible to the administrator. Logical structure log is displayed as a schema (text / diagram). For this level, everything that we mentioned as administrative duties, privileges, is defined, data types and connections between data are defined, etc.

The local logical database layer represents the logical structure of the database part of the database used by each application. This database level is visible and available to the end user or application engineer. Writing a logical definition is called a view.

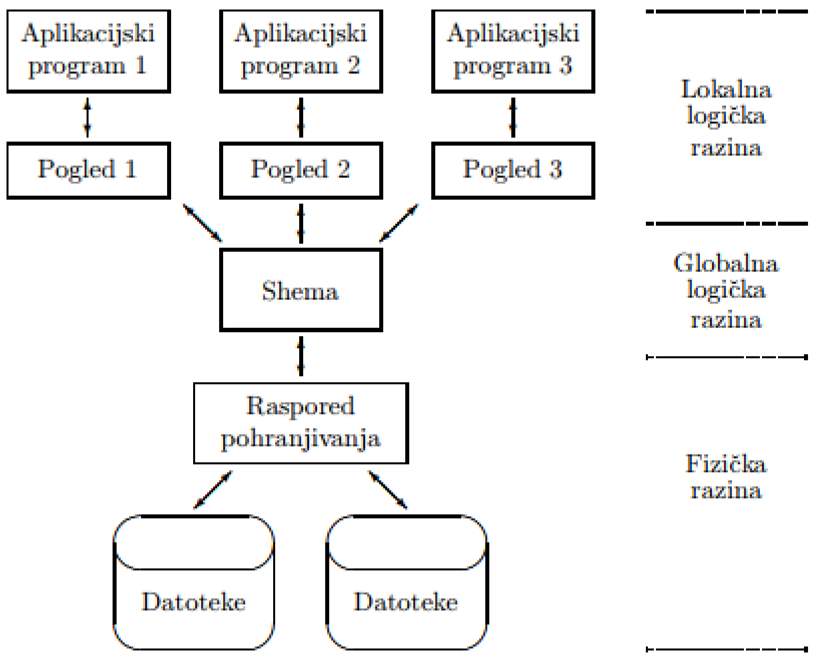


Figure 3.1.3.1. Database levels

Depending on the level of the database, we also have a division / user commitment.

3.1.4 Languages for work with the database

The work of the user with the database is carried out using the user application and special languages. Languages for working with a database are divided into:
Data Description Language (DDL) used to write schemas. This language is used by an administrator or database designer.

- CREATE TABLE [table name] ([column definitions]) [table parameters]

CREATE TABLE [Students] (StudentID int, LastName varchar(255), FistName varchar(255), Address varchar(255));

Data Manipulation Language (DML) is used to establish a connection between the application and the database. This language is used to manipulate data from time to time to enter, change, delete, or read.

- SELECT …… FROM ….. WHERE ……..
- INSERT INTO …..VALUES
- UPDATE …..SET….WHERE ……
- DELETE FROM ……WHERE …..

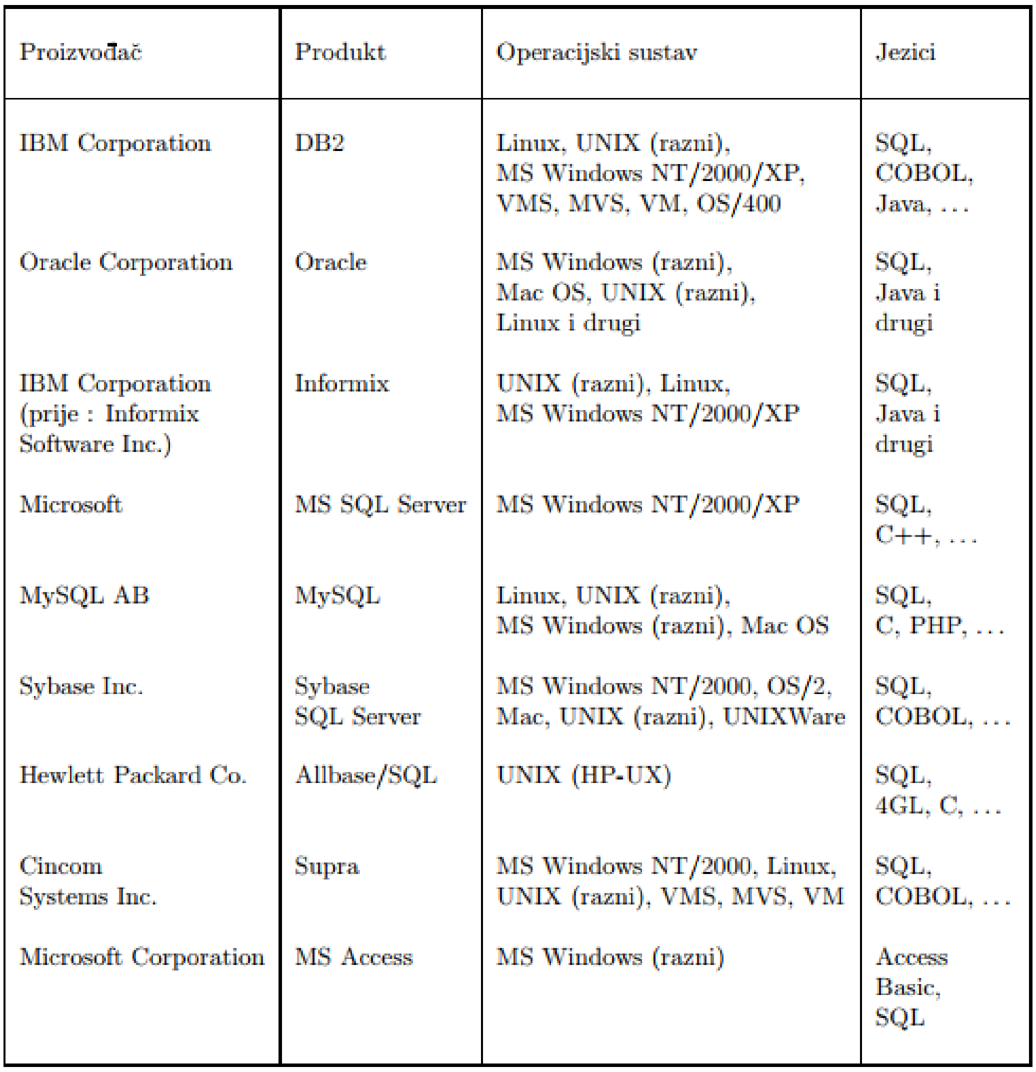
INSERT INTO Students (LastName, FirstName, Address) VALUES (‘Hodzic’, ‘Mirnesa’, ‘Kamycka’);

The Query Language-SQL serves the user for an interactive database search. Commands are procedural help to specify the result that we want to get. In this way, the method used to get the result is not specified.

3.1.5 Life cycle of the database

Today, it is almost impossible to imagine any company without a database. The database may be smaller or larger, but it certainly has a big relevance for each company. The company should make an assessment of whether a database is necessary at all, because in addition to this needs analysis, we have both data modeling and implementation, and it is necessary to check the database ie to test and maintain it.
If we are discussing data analysis, it is necessary to analyze the information / data that needs to be saved, how to make a connection between the data and what operations it is possible and need to do with these data.

The database life cycle can be displayed by tables



Proizvođač	Produkt	Operacijski sustav	Jezici
IBM Corporation	DB2	Linux, UNIX (razni), MS Windows NT/2000/XP, VMS, MVS, VM, OS/400	SQL, COBOL, Java, …
Oracle Corporation	Oracle	MS Windows (razni), Mac OS, UNIX (razni), Linux i drugi	SQL, Java i drugi
IBM Corporation (prije : Informix Software Inc.)	Informix	UNIX (razni), Linux, MS Windows NT/2000/XP	SQL, Java i drugi
Microsoft	MS SQL Server	MS Windows NT/2000/XP	SQL, C++, …
MySQL AB	MySQL	Linux, UNIX (razni), MS Windows (razni), Mac OS	SQL, C, PHP, …
Sybase Inc.	Sybase SQL Server	MS Windows NT/2000, OS/2, Mac, UNIX (razni), UNIXWare	SQL, COBOL, …
Hewlett Packard Co.	Allbase/SQL	UNIX (HP-UX)	SQL, 4GL, C, …
Cincom Systems Inc.	Supra	MS Windows NT/2000, Linux, UNIX (razni), VMS, MVS, VM	SQL, COBOL, …
Microsoft Corporation	MS Access	MS Windows (razni)	Access Basic, SQL

Figure 3.1.5.1. Database life cycle

Data modeling involves links that are established between data to make everything look like one global entity. This whole scheme can later be changed by normalization, i.e. To make the change in order to facilitate data manipulation, increase the performance of the database and give a better result.

The implementation of the database is done by professional engineers ie. Physically, a database is created on the computer. Parameters are set up to allow as much operation as possible over data.
Every database needs to be tested. Professional database testers check database functionality, however, users themselves can test the database for themselves and inform about possible mistakes or shortcomings that can be remedied in which database improvements are made. Professional testers, prior to physical database implementation, check all possible mistakes that may arise in the needs analysis, data modeling and the implementation itself.

Data maintenance involves working with the database after it has been applied in the application, and it deals with repair errors, improvements to the database performance database. This continuously monitors the operation of the database but on the other hand does not interfere with the work of the user.

3.2 Modelling the database

3.2.1 Entities, Links and Attributes

The database should have a realistic picture of links between entities and entities so that it can serve the purpose in the best possible way. Namely, it is necessary to look at the real reasons for having a database, what we want to achieve, what results we expect and who are the factors that are important in making decisions. This means, we have to design a scheme that represents the abstract of the real world with the needs and goals.

Modeling the database depends on:
- Entities
- Connections
- Attributes

An entity is the object or event that represents a factor that interests us. An entity is a factor that carries the meanings that matter to us, we want to store data on that factor-entity, we want to look at such data and see the relationship / dependence of that factor in relation to others.

The relationship is a relationship between the two entities, their dependence.
Attributes represent entity information and are used to describe links between individual entities.

Entities and attributes

An entity can be a human being, object, or event. Entity is everything that precedes interest in our future work. Each entity is described by attributes. Attributes give more information about the entity itself, but only important data that is a link to another entity or are relevant to the entity itself. It is important to note that in situations where an attribute requires an additional description, i.e. then the attribute must be attributed to the entity itself. Also, if the attribute has more value, then in these situations, the attribute itself must be declared an entity. One example is:
The buyer (with its attributes: name, surname, address, id number) is registered as a user of the bank account RACUN.

Connections
Relationships are the relationship between the two entities of several entities. The connection must be specified with a single "ratio" called the functionality of a link:

a) maximum one address --- if for example in the ADRESA entity we put the address of the buyer's home, the address and the delivery etc.)
b) One-to-many (1: N). The functionality is that an entity can be linked to more than one instance of another entity (for example, the BUYER entity can have multiple entities RACUN.
c) Many (M: N). This functionality means that a single copy of an entity can be linked to 0, 1 or more copies of another entity, and also a copy of another entity may be associated with 0, 1 or more copies with the first entity.

3.2.2 Entity Diagram Relationship

Entity diagram relationship known as ER schema or ERD is way to show flowchart of relationships between entities. It is usually used to illustrate relational database, as design also as tool for debugging relational database.

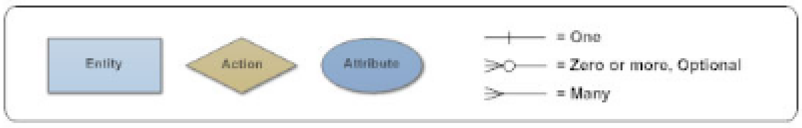


Figure 3.2.2.1. Tool for modelling

Figure 3.2.2.1. shows component what are going to be used for physical illustration database.

3.2.3 The Rational database

In the late 60's of the 20th century, ideas about relational databases appeared, and it was discussed for a long time that after 20 years it would become something that began and applied.
Today, this model of relational databases can be found in almost every company. We will continue to write about this type of data.
The relational model of the database can be represented by a model of a rectangular table that are interrelated with relatives. Each relationship has the name that is listed between the two entities, and by name it is different from the other link. Also, the connection fusion is specified. In the rectangle, the first part, we can see the name of the entity, and after that we have the listed attributes. Each value is added to the value of the data for example the date has a date value, a name - in most cases it has the values of varchar. We can say that each attribute has a defined domain of the attribute - a set of allowed values that define the attribute itself as the data.
If we have a table listing the data on the BOOK entity, we can say that one line of that table is usually one instance of the entity. Red is called n-torque. In one relationship, there must not be two equal n-torques. The number of attributes is the degree of relation, and the number of n-torques is the cardinality of the relation.

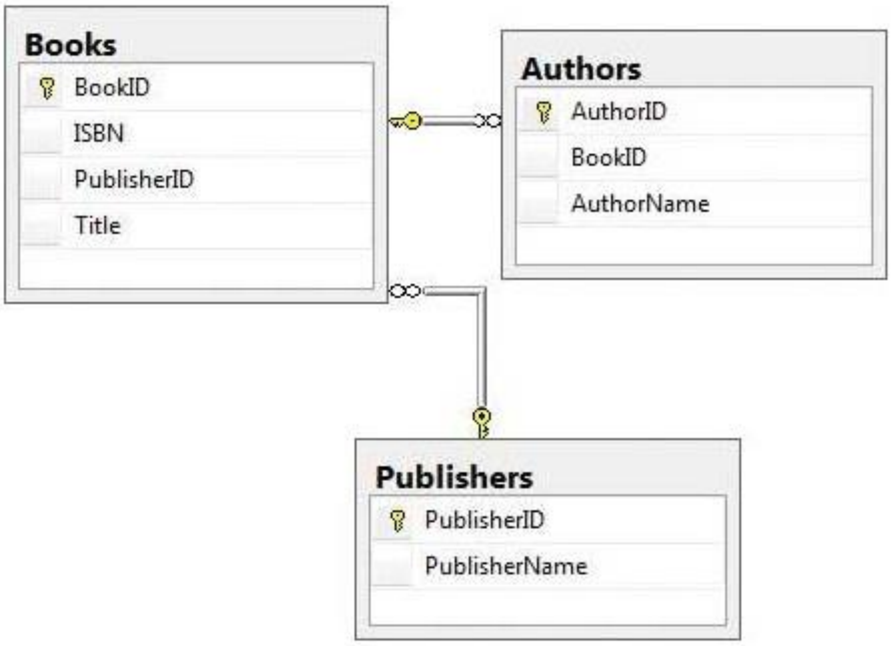


Figure 3.2.3. Relational database_exampleBOOK

We can say that there are no specific rules for sorting attributes. Only what is importat what database fulfill requrements in form of normalisation.

3.2.4 Converting the ER schema into rational

Modeling the database is done in seven steps:

1. Mapping the entity's regulatory types
Each entity E in the ER scheme has a R relation that contains all the attributes of that entity. If we have complex attributes, then the R relation can also contain the free components of those attributes of the entity E. The relationship / relation between the individual entities is set by an attribute that is common to both entities and this attribute is called the key. This key is called foreign key. Also, we have an attribute that is unique and this key is called the primary key.

RADNIK (LIME, SSLOVO, PREZIME, <u>MATBR</u>, DATRODJ, POL, PLATA, ADRESA)
SEKTOR (NAZIV, <u>SBROJ</u>)
PROJEKAT(NAZIV, LOKPR, <u>BROJPR</u>)

2. Maping week data types
Entity E is owned by several weak entities S which also has a relation R. This relation R contains all the free entities of the entity S. If we have complex S atribut entities, then the relation R contains all the free components of all complex attributes from S. The prime key of the relation R represents a combination of the primary key of the owner E and the partial key of a weak entity S.

CLAN-PORODICE (MATBRRAD, IME, POL, SRODSTVO, DATROĐ)

3. Link mapping 1: 1
The V connection (1: 1) in the ER semi is identified by the S and T relations. The primary key of the relation T is introduced as the external key of the relation S. One of the solutions can be the formation of a common relation R for all types of S I T entities that includes all the attributes. The given solution can be good if both entities participate in V.

4. Maped Link 1: N
The relation V (1: N) represents the relationship that the entity S is building on more than one side.

5. Link mapping M: N
Connection (M: N)

6. Mapping more-to-more attributes
A type attribute belonging to the entity E or V will have a new relation R containing the A and I primary key K belonging to the recitation of the entity E or the connection V. The primary key of the relation R represents the combination of A and K.

7. Mapping of nth- connections
If we have a number of connections greater than 2.

3.2.5 Relational model, network and hierarchical model

The database model depends on the purpose itself, or what we want to achieve by using the database. We can consider three basic database models: relational, network and hierarchical database model. By the very structure, the network and hierarchical model are very similar, but the relational model differs significantly from these two. The difference between the relational model I of the hierarchical or the network is the way of showing and using the links between the entities.

- A network model is a model where the connection between the entities can be directly displayed. The same goes for hierarchical fashion. Here we have pointers that serve to "materialize" relationships. This "materialization" is seen to be written in a single record of the address of another linked record. So, we have limitations on link functionality. These models are used for simple operations with one record (entry, change, deletion, reading), which increases the speed and efficiency, but the user of such databases can only use the links that are indicated in the schema.

- The relational model contains a link that is not materialized. This means that the connection is dynamically established while working with data, using the value of the attribute in the n-torque of various relations. It is possible to combine data from various relations here. This gives a cheaper range of data work, but the speed of data work is slower, connections must be reestablished each time. By using this model, the user can create and connections that were not foreseen in the phased-in process itself.

3.2.6 Normalization of the database

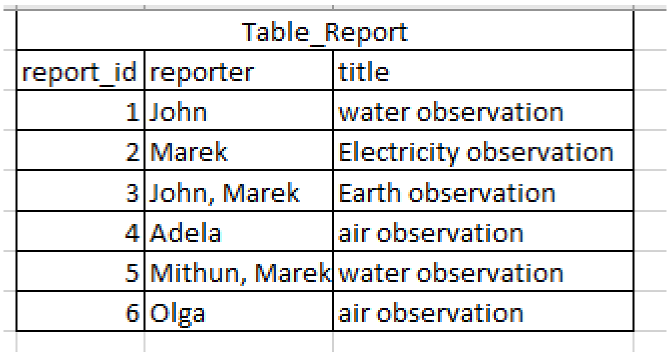
Normalization of a database represents the process of creating a database with a higher level of quality. So, it is necessary to emulate unnecessary data, increase the higher fuctionality, harmonize data and make more sense of data. By this process we organize the data and we get more flexibility of the database.

The theory of normalization is based on the notion of normal forms. These normal forms can be classified into three groups: the first, the second, and the third, the normal form.

And normalization is the application
- mathematical
- formal rules

This creates the correct creation of the model and its logical connection.
Normal Forms:

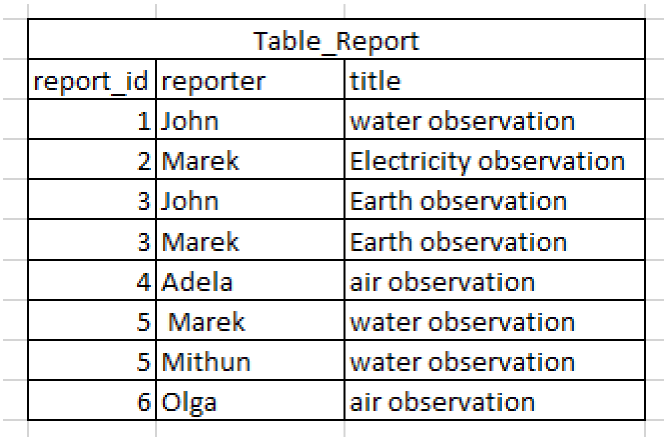
1NF - the first normal form is the process of organizing the data that is organized so that all records must have the same number of fields. This means that the name and last name can not be written in the same field. This procedure achieves easier and more efficient sorting of data (for example, in order to sort data if we have two data in one field: name and surname). Also, this norm introduces a rule that the database can not contain repetitive values.



Table_Report		
report_id	reporter	title
1	John	water observation
2	Marek	Electricity observation
3	John, Marek	Earth observation
4	Adela	air observation
5	Mithun, Marek	water observation
6	Olga	air observation

Figure 3.2.6.1. No-normal table of database

To get 1NF, we should seperate rows with names of two reporter info one for exaple like



Table_Report		
report_id	reporter	title
1	John	water observation
2	Marek	Electricity observation
3	John	Earth observation
3	Marek	Earth observation
4	Adela	air observation
5	Marek	water observation
5	Mithun	water observation
6	Olga	air observation

Figure 3.2.6.2. 1NF table of database

So, here it is made that each row consist only one value.

2NF - another normal form represents the data organization process by which data is organized so that 1NF is satisfied, all fields are single-sided and completely dependent on the primary key. So, each table must have data on only one entity. So, for example, The respect of this form is that we have two entities BUYER AND ORDER, so that there is no mixing, the TABER BUYER contains customer information, and the PROCESS table contains the order information. Otherwise, we would have a mix of data. In this case we divided the data into two tables. This procedure is called decomposition.

3NF - The third normal form represents the organization of data so that the norms 1NF and 2NF are satisfied. Fields should not depend on each other here. This rule does not apply to fields that are connected to the primary key.

3.3 Database Architecture

Database can be presented as 2-tier or 3 tier architecture. The Database architecture depends on few factors :
- how users are connected to the database to solve issue
- reasons why we need that database
- data structure and data manipulations
- performanse
- duration of useage database.

The users can be directly connect to the database and many users in the same time can use the same data. Also, request can be received by intermediary layer . Intermediary layer synthesizes the request and then it sends to database.

3.3.1 N-tier Architecture

2-tier architecture is represented without any user interface, so here we have directly interaction between application program and the database. The user is not involved in interaction with database.
One of example is school application that interact with the database to display the reports of all the students who are signed for different subjects.
In this case, we dont need any inputs from the user to display required data.
We can add plus another example; railway ticket reservation system where we can have situation that two different person from different places want to take tickets for the same destination (start and end points of trip is the same, same day, the same train).so , we have requirement for two different person -same trip same train .

Reservation system will take the request from users, and queues the requests entered by each of them. So, the request will be entered to application layer and request will be sent to database layer. When database finish by processing the request, the result will be send back to application layer for the user.

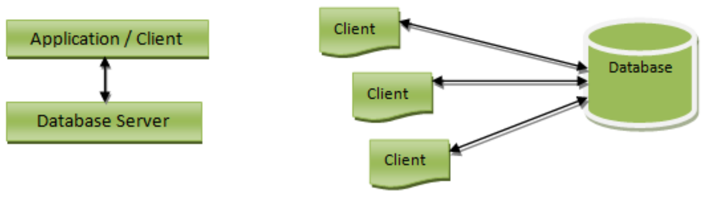


Figure 3.3.1.1. 2-tier Architecture

Advantages of the 2-tier Arhitecture

- Directly communicates between the database and application
- Fast response fast retrieved requested data - (if we have less number of users) Easy to modify – any changes required, directly requests can be sent to database
- Easy to maintain – When there are multiple requests, it will be handled in a queue and there will not be any chaos.

Disadvantages of 2-tier architecture:

- Retrieving data (in case of huge number of users.)

(All the requests will be queued and handed one after another. Here, we will not have respond to multiple users at the same time)

- for some user request this architecture is less cost effective

3-tier database architecture includes plesentation layer and application layer as :

- Presentation layer / User layer presents the layer where users are able to uses the database without any knowledge about underlying database. For example, user layer can be presented as a registration form where user can enter own details . So, user doesn't know or doesn't case about procedures what are happening after pressing submit button (just know that data are saved.

Presentation layer is layer where all the data from the user are taken and   sent to the next layer for processing.

- Application layer represents the underlying program. The application layer include logics like validation , calculations and manipulations of data, thanks by make decision to save entered user's data (if is request valid) or send back the message (if request is invalid) to presentation layer.

3.3.2 Data layer or Database layer

Data layer or Database layer contains all the tables, their mappings and the actual data. This layer communicate with user by using programs in the application layer (inserting data into the tables in the database layer and retrieving data to view in the web browser)

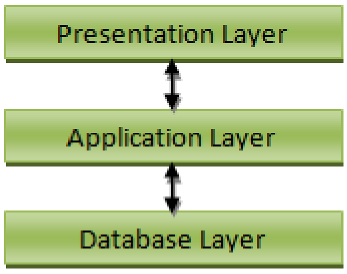


Figure 3.3.2.1. Layers

Advantages of 3-tier architecture:

- maintain and modify are easy because all validations about changes will be already done in application layer .
- Higher level of security because all communication will be done -filtered via application layer-no direct access to data in tables
- time consumptions -multiply request can be manage and responded at the same time

Disadvantages of 3-tier architecture:

- little bit more complex architecture

3.3.3 MySQL architecture

The MySQL application layer consist three components which are used for communication between end user and MySQL RDBMS. Every of this components is specified for different group of users, for example administrative interface component of application layer is for administrators.

Users like data analyst or bussinness analyst or something like that interact with MySQL RDBMS via query interface by using mysql to view result .
The users like clients, they interact with MySQL RDBMS by using application made by C++, java, ruby, PHP etc.

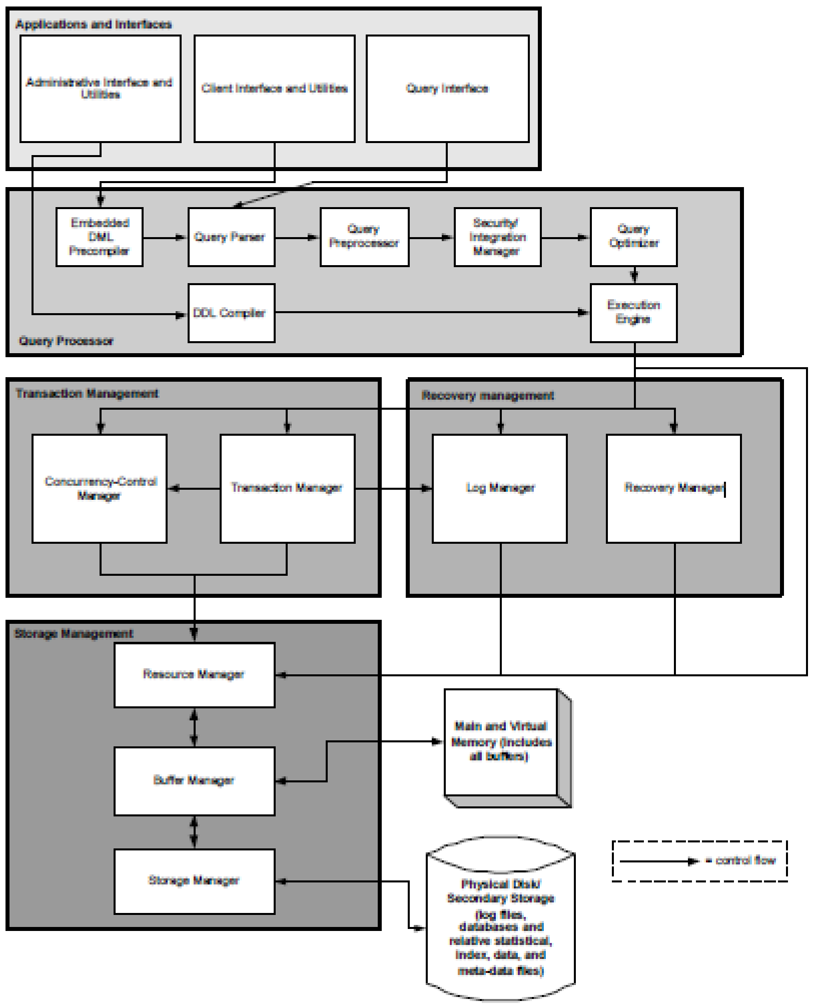

Applications and Interfaces
Administrative Interface and Utilities
Client Interface and Utilities
Query Interface
Embedded DML Precompiler
Query Parser
Query Preprocessor
Security/ Integration Manager
Query Optimizer
DDL Compiler
Execution Engine
Query Processor
Transaction Management
Concurrency-Control Manager
Transaction Manager
Recovery management
Log Manager
Recovery Manager
Storage Management
Resource Manager
Buffer Manager
Storage Manager
Main and Virtual Memory (includes all buffers)
Physical Disk/ Secondary Storage (log files, databases and relative statistical, index, data, and meta-data files)
= control flow

Figure 3.3.3.1.MySQL Architecture

Logical Layer

- Query processor is used to response on users requests to view or manipulate with stored data. Thanks by SQL- data manipulation language what is used to filtered data, optimize and make better view of result , customer will get requested response.

Query processor has few components as Embedded DML precompiler, DDL compiler which are responsible for this job.
One of them is Embedded DML precompiler which is used to precompiler client's request into relevant SQL statements or make translation commands received from client into corresponding SQL statement.
DDL compiler -Data Definition Language compiler is used to process administrator's commands like SQL statements to make database access directly.
Query parser is used to parse MySQL queries given by administrator or end user.
Query preprocessor is used to continue process SQL syntax sent from query parser to check and determine validation of query . There is two possible scenario; one of

them if query is valid than continue process, otherwise if query is not valid than inform client about error of query processing .
Security / integration manager is used after query preprocessor and in cases if MySQL is valid to check access control list for the client. This is very important to prevent unautorizied and malicious user's access.
Query optimizer is used to analyze SQL queries in order to optimize them if cases that previously steps are passed . Thanks by Query optimizer , MySQL
Execution engine will continue process execution queries and access physical database layer. This will be hapend after query's optimization by query optimizer.
Scalability/Evolvability is used to supports the evolvability of the system.

3.3.4 Transaction Management

Transaction Manager

The transaction manager is used to execute transaction of one or more MySQL commands.
Here , in this level are going to be take care about many tasks; resolving any deadlock what can happend is cases that the same data are needed to be processed in the same time in two different transaction, also for issuing the COMMIT command to process transaction and the ROLLBACK SQL command is used to solve crash.

Concurrency-Control Manager

The concurrency-control manager is used to execute transactions separately and independently. Concurrency-Control Manager is responsibly to take care that data are locked for other transaction up to finish and complete data manipulation taken by first transaction.

Recovery Management is given as :

Log Manager
The log manager is used to logging every operation executed in the database. The log manager helps recovery database to its state what was before crashing system if that unhappy situation happend . The log file is saved on disk through the buffer manager and keeps database commands.

Recovery Manager
The recovery manager is used to restore the database to its state system in last time of period what was saved before system crashing. We dont want to loose our data in

the any point of the time, so, recovery should be only from point saving data, also, we want to catch our information anda data what were used in the point of the time when cionnection stopped, so, recovery play very important part of maintance database. There are so many companies what provide solutions which are avble catch our information what were used from point of the time when data are saved up to point of the time when connection stopped or when our database satrt b eout of function.

Storage Management

Storage is physically layer where is not allowed dynamic access. Storage Management includes the Resource Manager, the Storage Manager and Buffer Manager.

Storage Manager

Storage manager represent lower level of Storage Management and interfere requests between the Buffer Manager and secondary storage. The Storage Manager is taking data from physical disk through the underlying disk controller after that taken data will be sent to the Buffer Manager.

Buffer Manager

Buffer Manager is used to make decision and allocate memory resources, it means how many buffers is needed to viewing and manipulating data.
All requests given for allocation buffers are made from the Resource Manager.

Resource Manager

The Resource Manager is used to accept requests from the execution engine, put them into table requests, and request the tables from the Buffer Manager.

NoSQL architecture provides access to data which are on low -level layer by using some of programming languages depends of programmer what makes differences compared on SQL data system where

3.3.5 NoSQL Architecture

NoSQL architecture provides access to data which are on low -level layer by using some of programming languages depends of programmer what makes differences compared on SQL data system where is used data management logic (application languages, SQL, and DB-specific stored procedure languages).

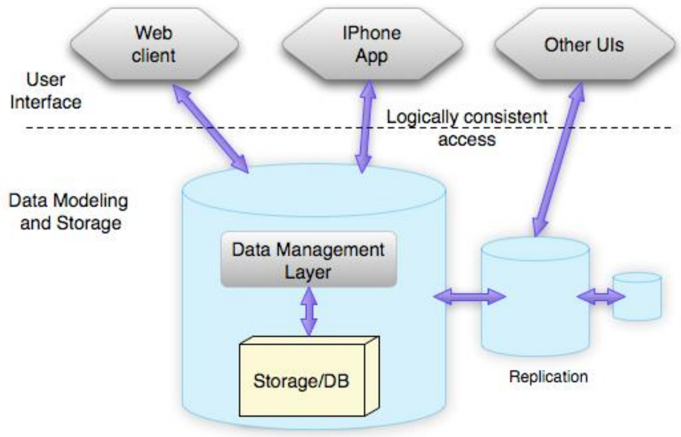


Figure 3.3.5.1. NoSQL Architecture

NoSQL database provides hierarchical nested structures in data entities what make accessing more easy   from parent document into child document. Real relations are necessary if cases where data entities should be accessed individually.

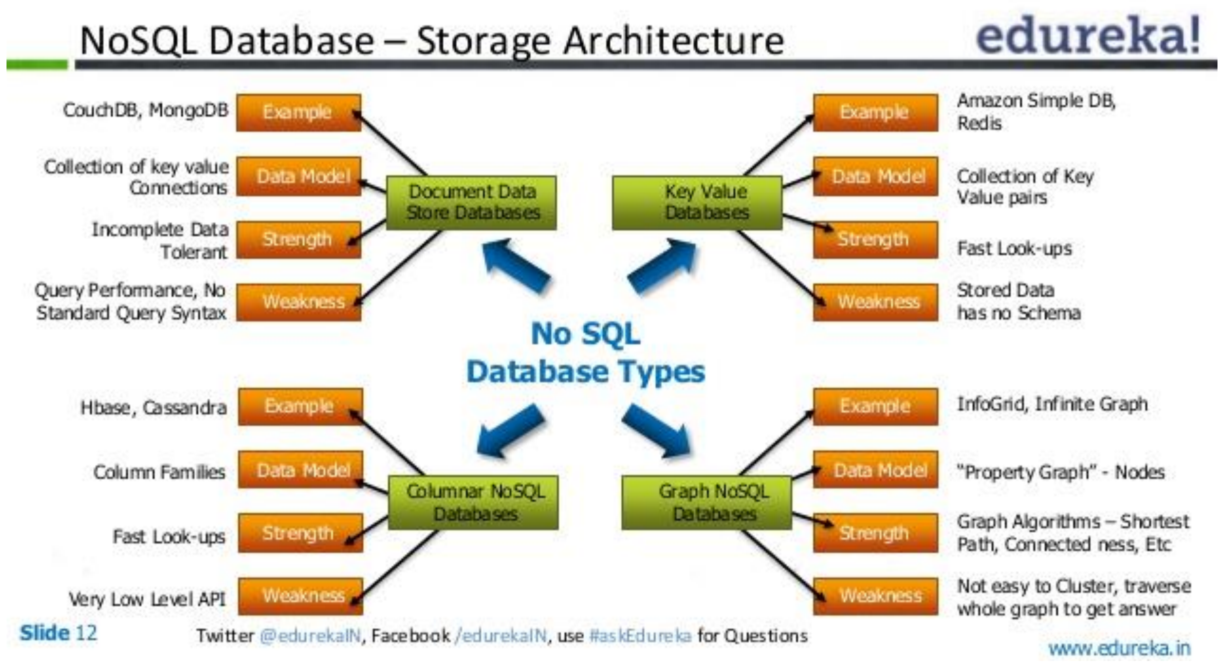


Figure 3.3.5.2. Example of NoSQL storage

Thanks by NoSQL database, unnecessary bi-directional relations may be eliminated and get simply design , but also, in some cases relations databases are better choice and represent important useage.

NoSQL system's roadmap of considerations:

"

- *Data and query model*: Is your data represented as rows, objects, data structures, or documents? Can you ask the database to calculate aggregates over multiple records?
- *Durability*: When you change a value, does it immediately go to stable storage? Does it get stored on multiple machines in case one crashes?
- *Scalability*: Does your data fit on a single server? Do the amount of reads and writes require multiple disks to handle the workload?
- *Partitioning*: For scalability, availability, or durability reasons, does the data need to live on multiple servers? How do you know which record is on which server?
- *Consistency*: If you've partitioned and replicated your records across multiple servers, how do the servers coordinate when a record changes?
- *Transactional semantics*: When you run a series of operations, some databases allow you to wrap them in a transaction, which provides some subset of ACID (Atomicity, Consistency, Isolation, and Durability) guarantees on the transaction and all others currently running. Does your business logic require these guarantees, which often come with performance tradeoffs?
- *Single-server performance*: If you want to safely store data on disk, what on-disk data structures are best-geared toward read-heavy or write-heavy workloads? Is writing to disk your bottleneck?
- *Analytical workloads*: We're going to pay a lot of attention to lookup-heavy workloads of the kind you need to run a responsive user-focused web application. In many cases, you will want to build dataset-sized reports, aggregating statistics across multiple users for example. Does your use-case and toolchain require such functionality? "[1]

NoSQL database can be also part of complex big data architecture as is shown on figure bellow

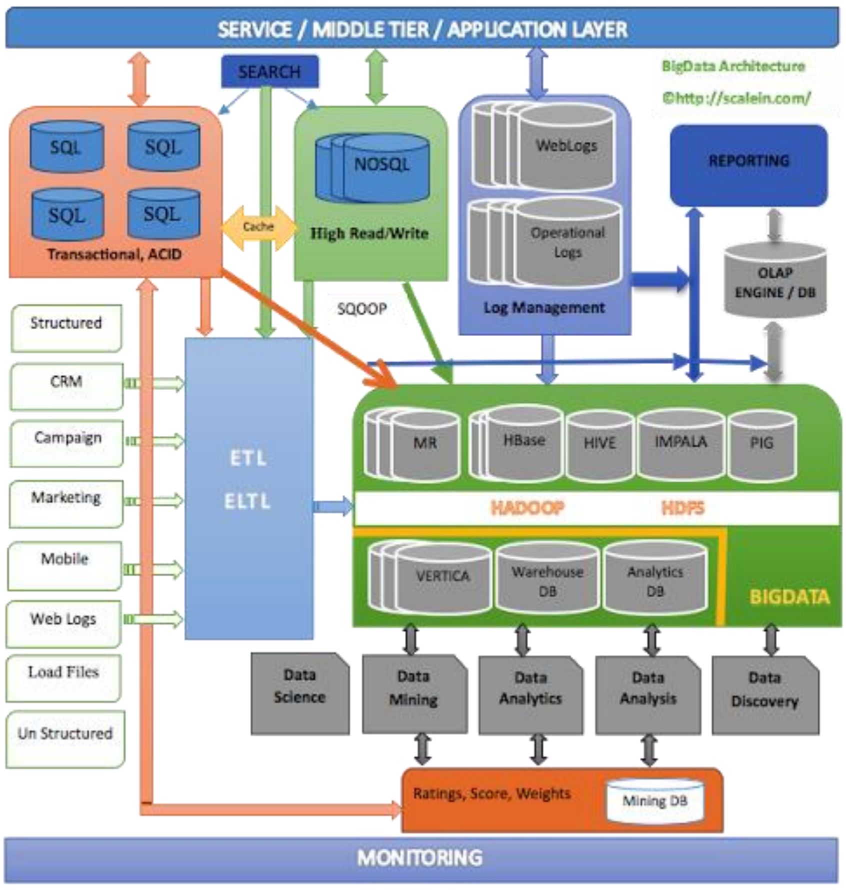


Figure 3.3.5.3. Service/ Middle Tier/ Application Layer

BIG data architecture given on Figure 1.5.5.3. shows below:

- Storage of different data types (structured/relational, semi-structured (marketing/campaign/mobile/web logs), and unstructured (files) and log files
- Loading data in to a variety of data bases
- Mining data for extra insights
- High speed analytics
- Data warehousing for reporting, and reporting over the live system
- Batch analysis (Hadoop and ecosystem)
- Log file management (web, operational)
- Transactional data store
- Web caching
- Search

3.3.6 DB2 Architecture

DB2 database is developed by IBM, 1983 and support rational database management system of storing data. Work with DB2 means using C, C++ and assembly.

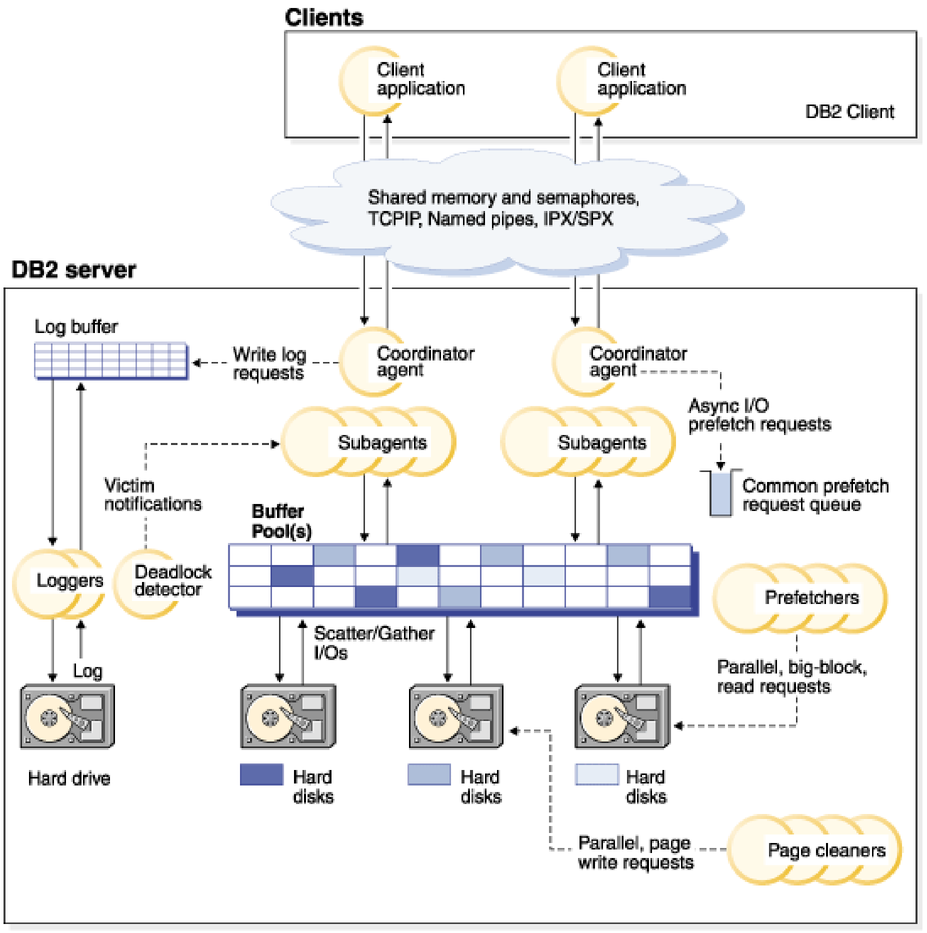


Figure 3.3.6.1. DB2 Server

Engine dispatchable units (EDUs) are given as DB2 agents, prefetchers and page cleaners. DB2 agents are responsible for processing SQL and XQuery. The request given by client will be processed by many subagents. Pooling algorithm is responsible to managing the agent and subagents

" Buffer pools are areas of database server memory where pages of user data, index data, and catalog data are temporarily moved and can be modified. Buffer pools are a key determinant of database performance, because data can be accessed much faster from memory than from disk.

The configuration of buffer pools, as well as prefetcher and page cleaner EDUs, controls how quickly data can be accessed by applications.

- *Prefetchers* retrieve data from disk and move it into a buffer pool before applications need the data. For example, applications that need to scan through large volumes of data would have to wait for data to be moved from disk into a buffer pool if there were no data prefetchers. Agents of the application send asynchronous read-ahead requests to a common prefetch queue. As prefetchers become available, they implement those requests by using big-block or scatter-read input operations to bring the requested pages from disk into the buffer pool. If you have multiple disks for data storage, the data can be striped across those disks. Striping enables the prefetchers to use multiple disks to retrieve data simultaneously.
- *Page cleaners* move data from a buffer pool back to disk. Page cleaners are background EDUs that are independent of the application agents. They look for pages that have been modified, and write those changed pages out to disk. Page cleaners ensure that there is room in the buffer pool for pages that are being retrieved by prefetchers.“ [2]

4 Practical Part

4.1 Business Architecture

This paper is intended to describe and process the work of the organization of financial institutions. As it is very difficult to access the database of banking systems, this paper will deal with banking systems as a Transaction System and Support to Financial Institutions. If we look at the Financial Services Volunteer Corporation, we will see that we have an organization that is non-profit, but still as such deals with finances. The NoSQL database is the best purpose in the domain in which we have a constant change and addition to data, because we do not have to add or change relations between entities.
Let's look at the FSVC page, we can describe a business program that supports financial institutions located in developing countries.

Using tools such as relational database, CraftCase and NoSQL, ie, business diagram, bussines architecture, JSON and SQL queries, we will describe the effect of this collaboration.

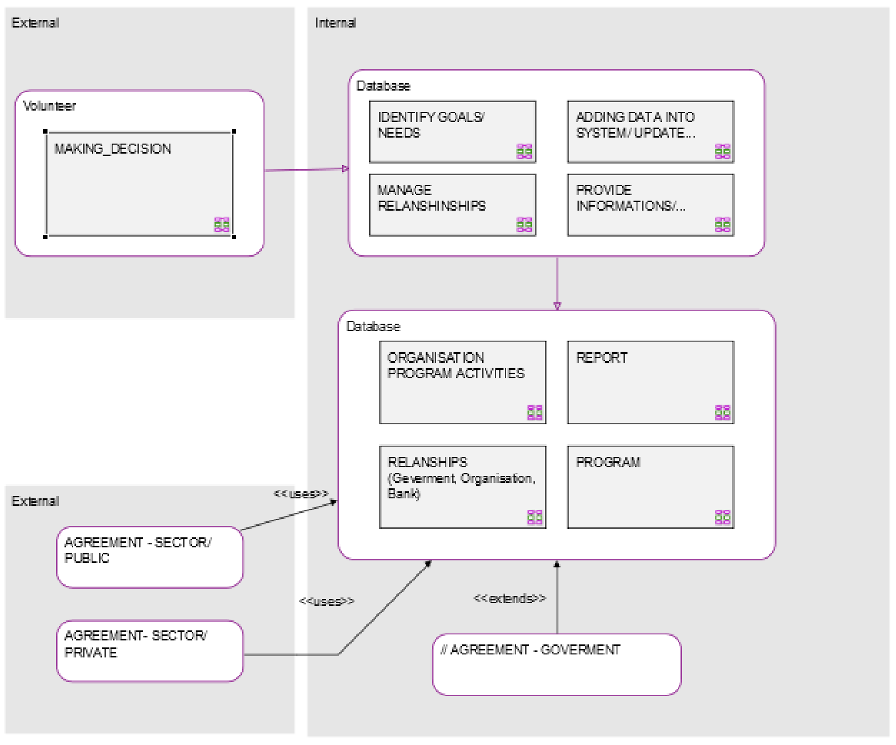


Figure 4.1.1. Arhitecture of volunteer program

4.2 Business Process Modelling

Business process modeling is a business process management and engineering system, that is, a set of activities that describe in detail a business process and a model that can be mapped to cod. This can be thoroughly analyzed, raised to a higher level and mapped into an automated system.
In this way, we do our analysis of our business, we add and take away the necessary and unnecessary for the purpose of optimizing and better results.
In order to create a business process model, in this paper, a Craftcase tool is used that enables, analyzes and simulates each step in order to avoid repetition, duplication of steps, and at best predict the dependence of decisions and how they affect one another.

An organization that provides financial support as a service, and which, on the other hand, represents a non-profit organization, is looking for volunteers to provide their time to financial institutions and institutions of developing countries. Dole fig. Br. Presents the business process of applying new members to this organization. This model represents a step by step and a straightforward sequence that goes beyond the Web site and the Application System for a member of this organization. Each step represented by the model is transformed into the code behind the web page.

The business model can be simulated that we can see all possible connections between the entities and in which way the response and how the reaction of one entity affects the reaction of the other entity. Also, this model can be clearly implemented in the code that each member, each method can be simulated in a number of ways, through a business model through SQL queries and JSON. Any query that requires a further simulation execution can be a query within the code (SQL or JSON).
What is more important is modeling the data --- how to organize the data with which we work, to create a unique form that can be used for all types of relational and non-relational databases.

If we look at figure below, we are able to see all action what are happening during registration one volunteer into system. There is actors, what can be represented as entities and action what can be methods what are going to converted into cod. Action/ methods are depended on some factors as result, so based on the some decision we have fluctuation of the process.

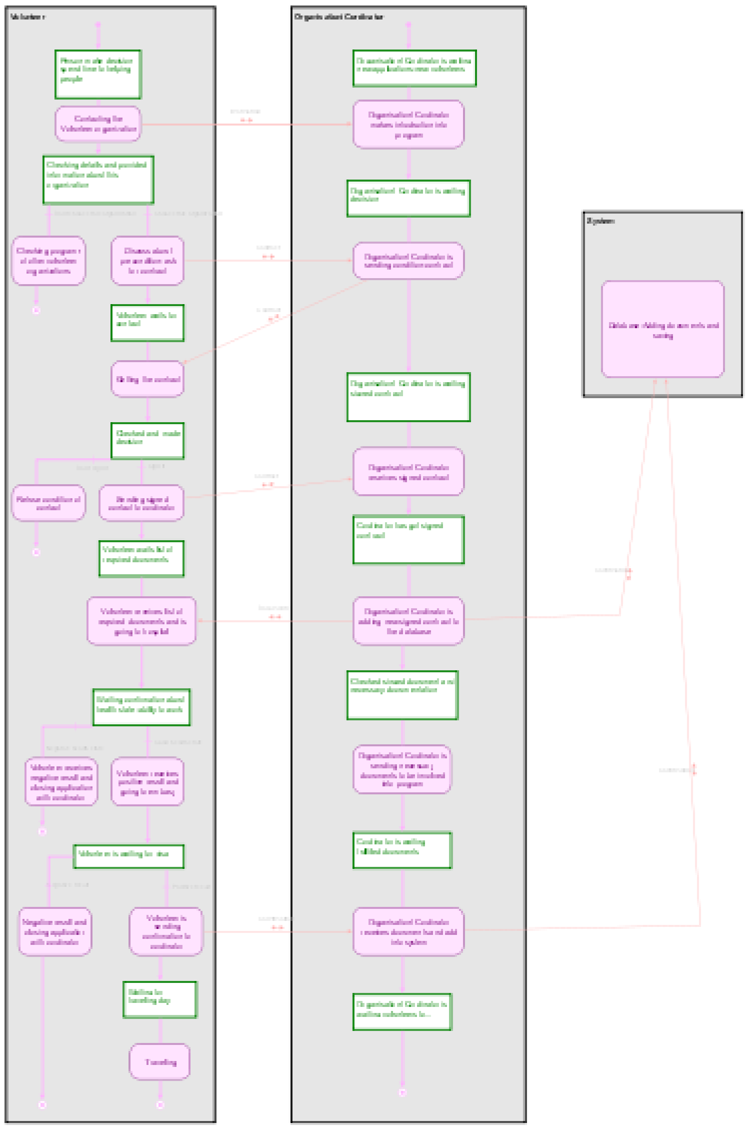


Figure 4.2.1. Business process model – first step comunications

Next, picture no. is the process of adding to the System Base of each new member of the organization, step by step, which should be modeled within the database. In this way we can see all the possible operations that can happen with the database by the administrator as the end user of the database.

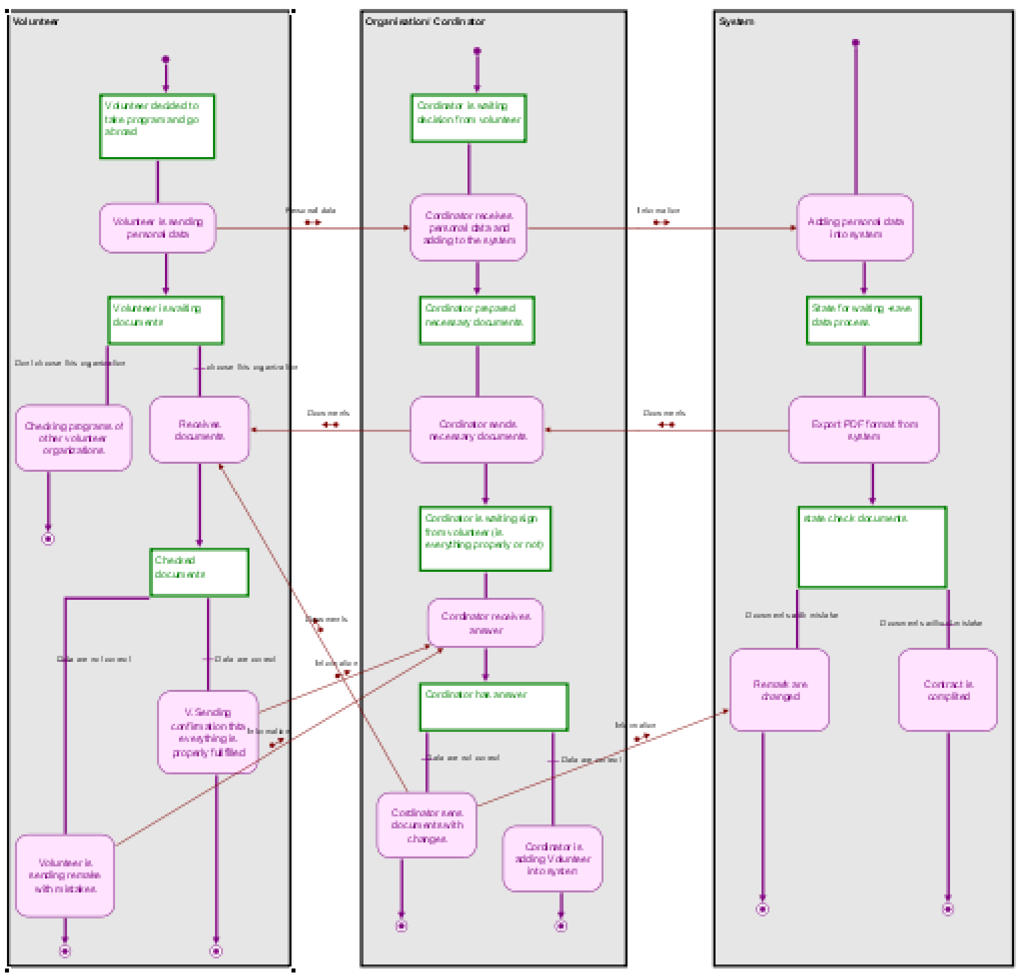

Volunteer
Volunteer decided to take program and go abroad
Volunteer is sending personal data
Volunteer is waiting documents
Don't choose this organization
choose this organization
Checking programs of other volunteer organizations
Receives documents
Checked documents
Data are not correct
Data are correct
V. Sending confirmation that everything is properly fullfilled
Volunteer is sending remarks with mistakes
Personal data
Documents
Information
Organization/ Cordinator
Cordinator is waiting decision from volunteer
Cordinator receives personal data and adding to the system
Cordinator prepared necessary documents
Cordinator sends necessary documents
Cordinator is waiting sign from volunteer (is everything properly or not)
Cordinator receives answer
Cordinator has answer
Data are not correct
Data are correct
Cordinator sens documents with changes
Cordinator is adding Volunteer into system
System
Adding personal data into system
State for waiting save data process
Export PDF format from system
state check documents
Document with mistake
Document without mistake
Remarks are changed
Contract is compiled

Figure 4.2.1. Business process model– add into database

Also, there can be represented others possibilities as modifications, updates and delete as method inside database as modify, update and delete. It is clear that the most used method in our case is add, but also, we should consider that in other cases bussines models for example, we can expect mode method what is like modify or simple like that.

This way describes a business model that represents a relationship between volunteers, organizations and sectors that need to provide adequate support in the realization of business, work and general development and development on the market.

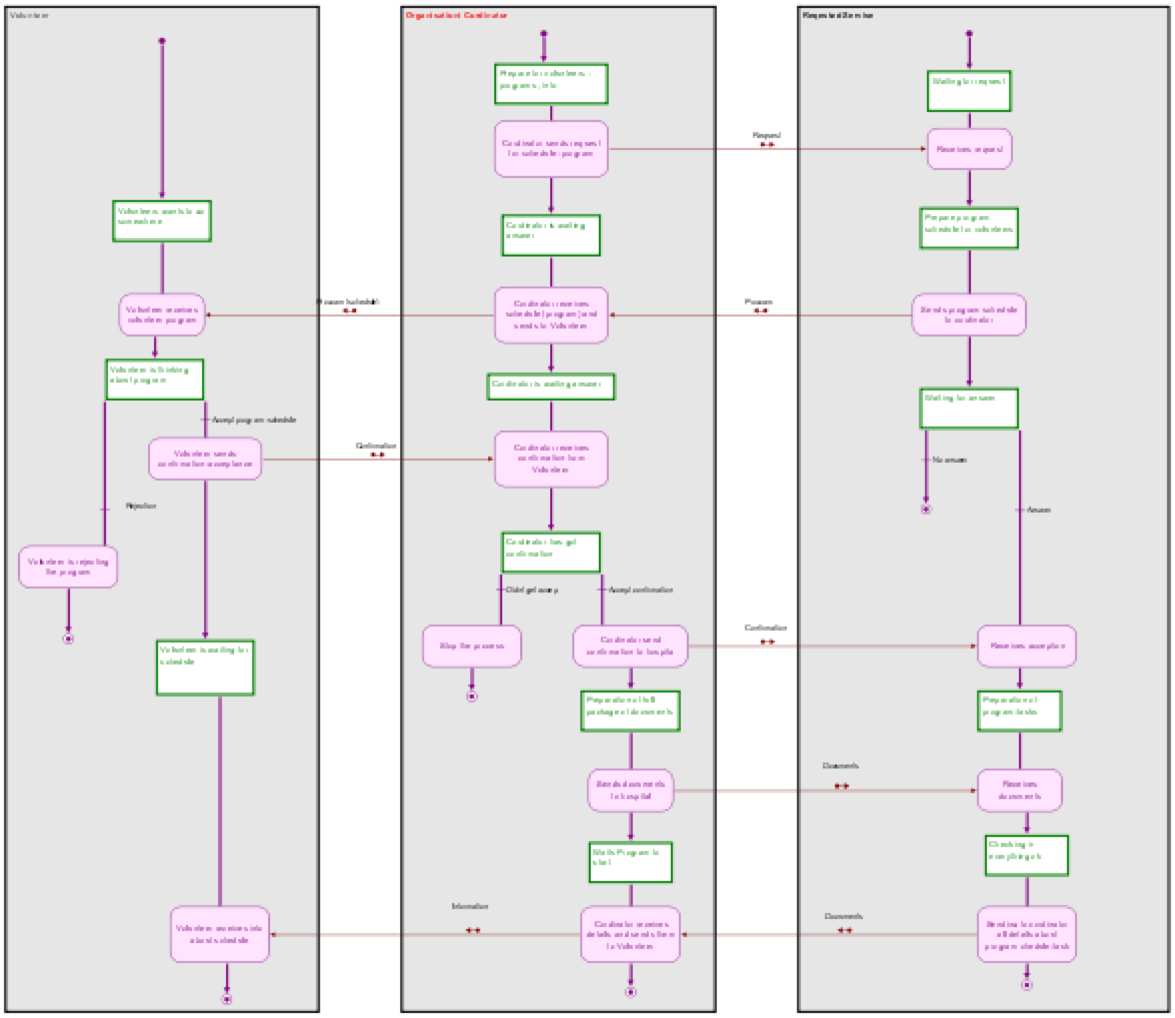
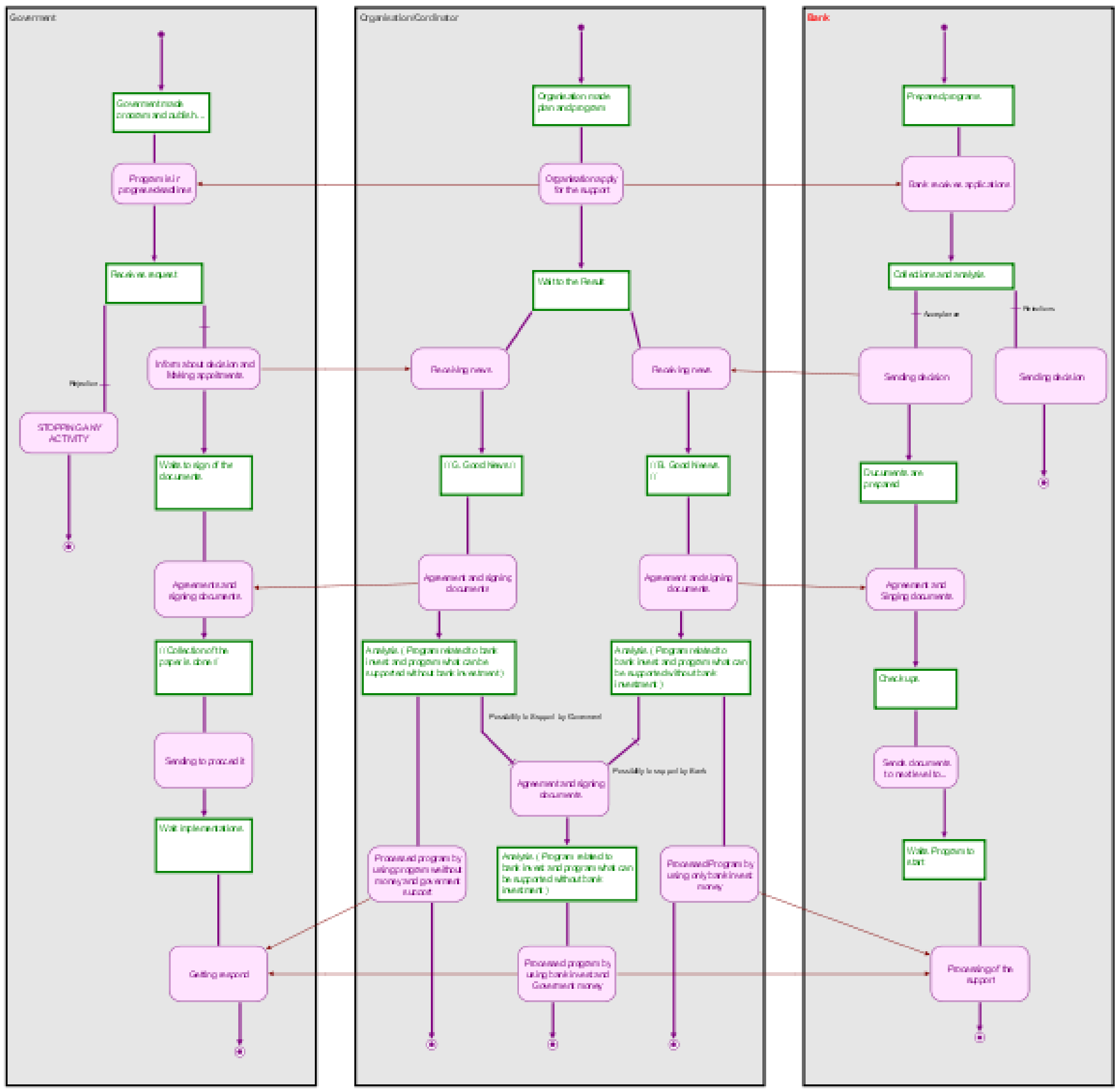


Figure 4.2.3. Business proces model- Program Activities

Here, if we see all actors and actions, we are able so imagine whole process behind this. Program can be updated, modified and also completely changed, what is seen based on reaction, if some decision is accepted or not, we have some mistakes or not and so on. Stop process can have impact into database as delete method.

Also, the same way in this way we can see the operations that are happening between the administrator as the end user and the database.



Government
Government made programs and publish...
Programs is in progress and line
Receives request
Rejection
Inform about decision and making appointments
STOPPING ANY ACTIVITY
Waits to sign of the documents
Agreements and signing documents
!! Collection of the paper is done !!
Sending to proceed it
Wait implementations
Getting support
Organisation/Cordinator
Organisation made plan and program
Organisation apply for the support
Wait to the Result
Receiving news
Receiving news
!! G. Good News !!
!! B. Good News !!
Agreement and signing documents
Agreement and signing documents
Analysis (Program related to bank invest and program what can be supported without bank investment)
Analysis (Program related to bank invest and program what can be supported without bank investment)
Possibility to Support by Government
Possibility to support by Bank
Agreement and signing documents
Processed program by using program without money and government support
Analysis (Program related to bank invest and program what can be supported without bank investment)
Processed Program by using only bank invest money
Processed program by using bank invest and Government money
Bank
Prepared programs
Bank receives applications
Collections and analysis
Acceptance
Rejections
Sending decision
Sending decision
Documents are prepared
Agreement and Singing documents
Check ups
Sends documents to next level to...
Waits Program to start
Processing of the support

Figure 4.2.4. Business process model- RP (Bank, Government, Organisation)

Here, it is seen more actions as accepted, rejected and other what make division based on where our process going we we are able get support from so many institutions. Based on the support, we are able classified which program is going to be executed. That process can be given as update (with "yes" or "no" answer) and also sort method.

The bussines process show us how we can make relationships and how to connect all actions in the database. Also, there is web page what follow all this to make whole process complete. An the end, we have to make report what can be public or not, but anyway it will stored into database for future actions. Here, he have insert method as adding into database. Also, we are able to make that after every received report we are going to stored it into database, so , we will have adding into database, update database, delete, and sort to analysis. Here as we are able to see, we are waiting all report from actors, and after that we are insert reports into database.

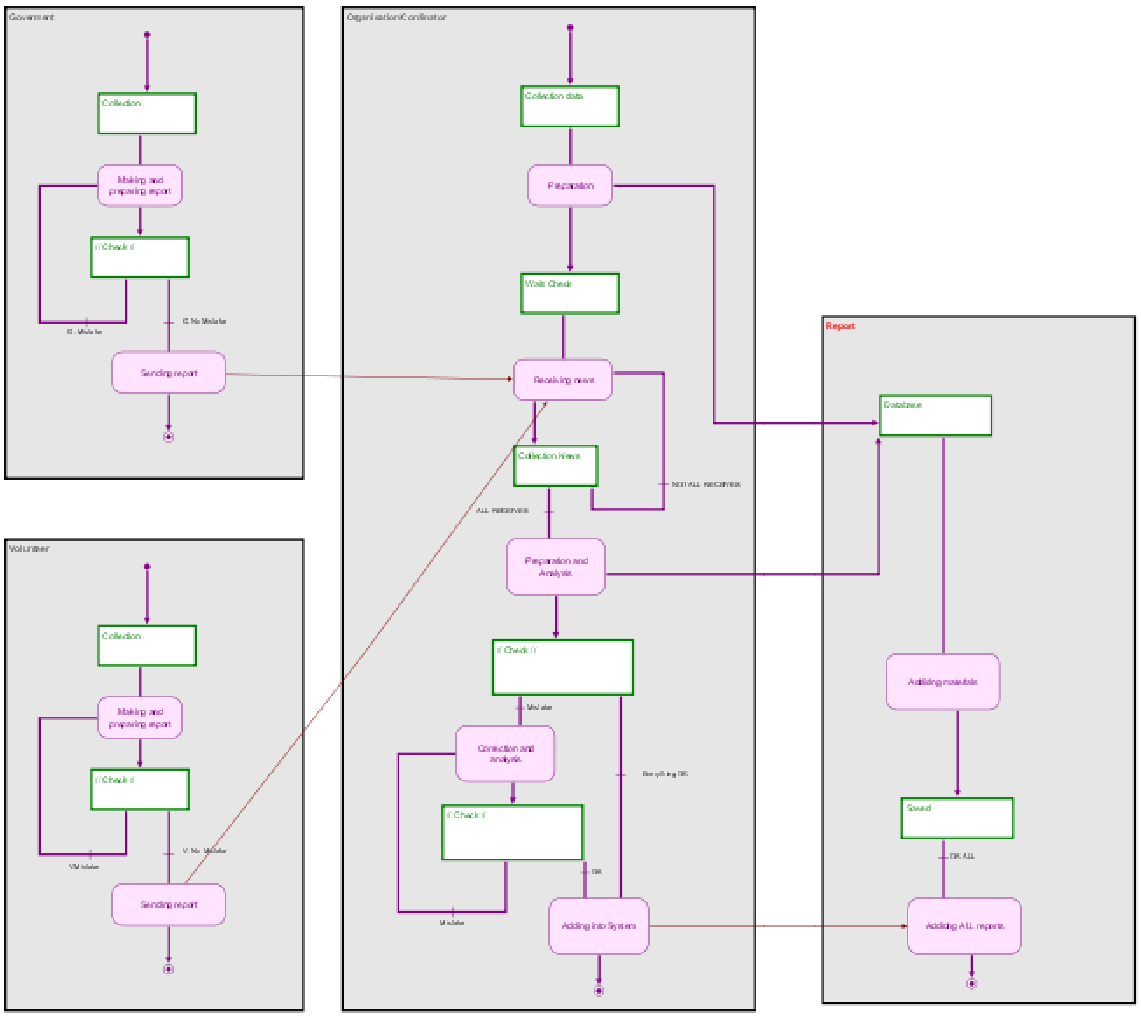


Figure 4.2.5. Business process model- Report

Unfortinally , here we are not able to see importance of our report because, based our report we are ab le continue with our activities, because we are using resources of the companies exactly knowledge of experienced people in field of finance.

Program activities will used resource from treports and also marketing resources to get new experienced banker or finance advisor as volunteer who want to help institution from development countries.

Also, main part play goverment. Because of it, other paties are more interested in the activities of the organisation. Make connection. That is moto of the organisation. Dont forget to help sectors which need help. Not just sectors, also to countries.

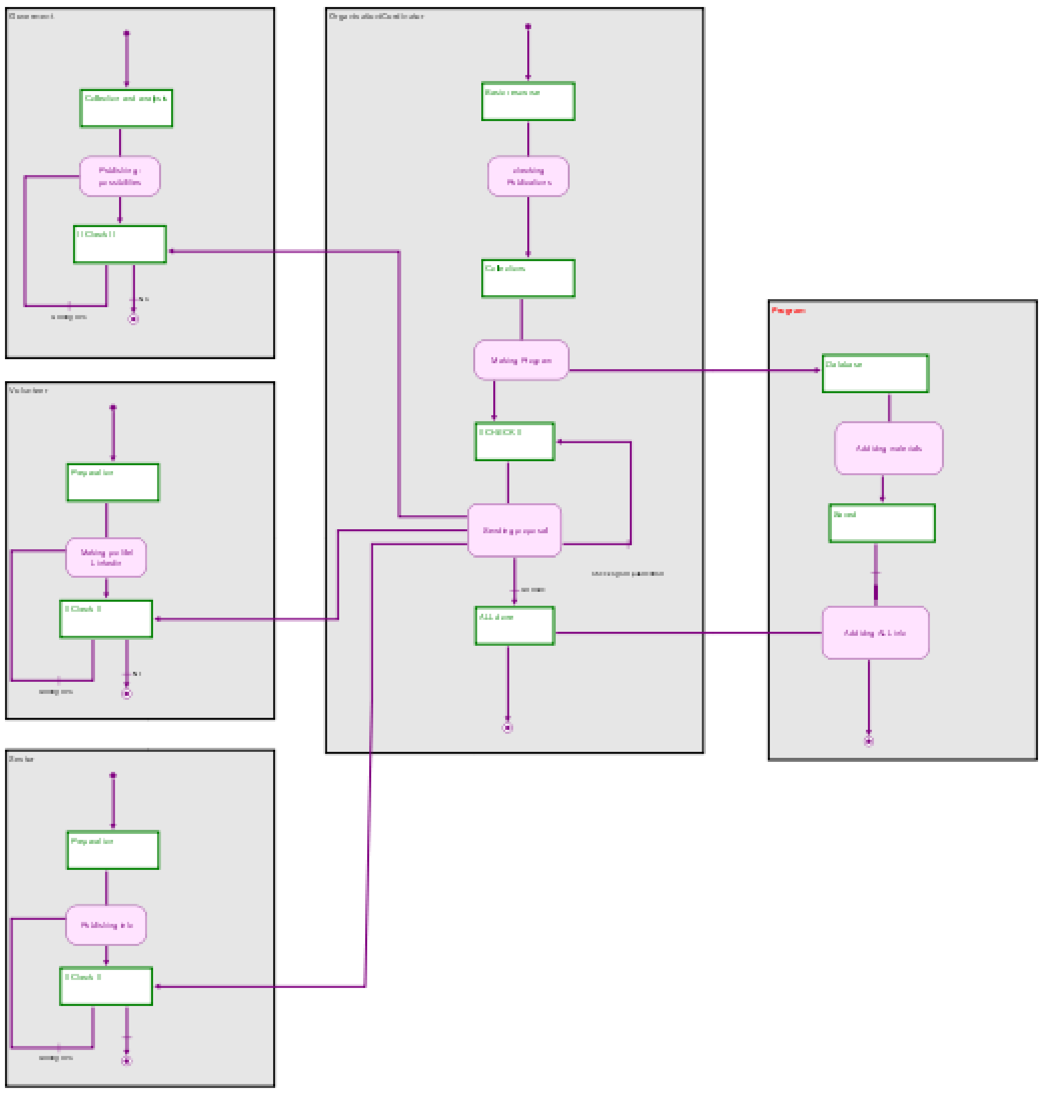
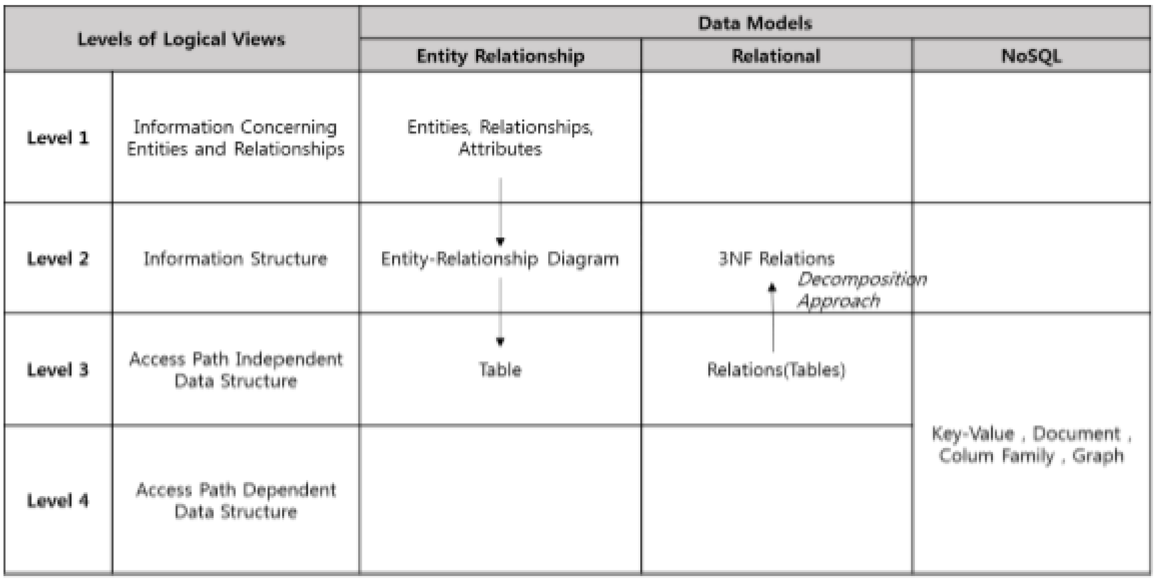


Figure 4.2.6. Business process model- Program

4.3 Conceptual Model

Database model can be represented by using conceptual data model (relational database) and nonconceptual data model (NoSQL). Anyway, try to don’t make big mess, and keep consistency of data, this work will represent NoSQL with applaying conceptual data model. Data model can be represent in way as design method for specific NoSQL, data method what are applicable for every NoSQL.

NoSQL data model



Levels of Logical Views		Data Models		
		Entity Relationship	Relational	NoSQL
Level 1	Information Concerning Entities and Relationships	Entities, Relationships, Attributes		
Level 2	Information Structure	Entity-Relationship Diagram	3NF Relations *Decomposition Approach*	
Level 3	Access Path Independent Data Structure	Table	Relations(Tables)	Key-Value , Document , Colum Family , Graph
Level 4	Access Path Dependent Data Structure			

Figure 4.3.1. Logical view of NoSQL

NoSQL conceptual shema represent high level of the data model / database what can be used to shown by UML ER and so on and also represent real world data.
Logical NoSQL shema represent data structure in logical way.
Physical NoSQL shema represent physical storage structure of the NoSQL.

This conceptual model of the data in NoSQL can be good used to make document MongoBD, in purpose of good representation of the e – commerce system, online banking where we are deal with transaction every millisecond. This is way to push in advance level consistency of used data.

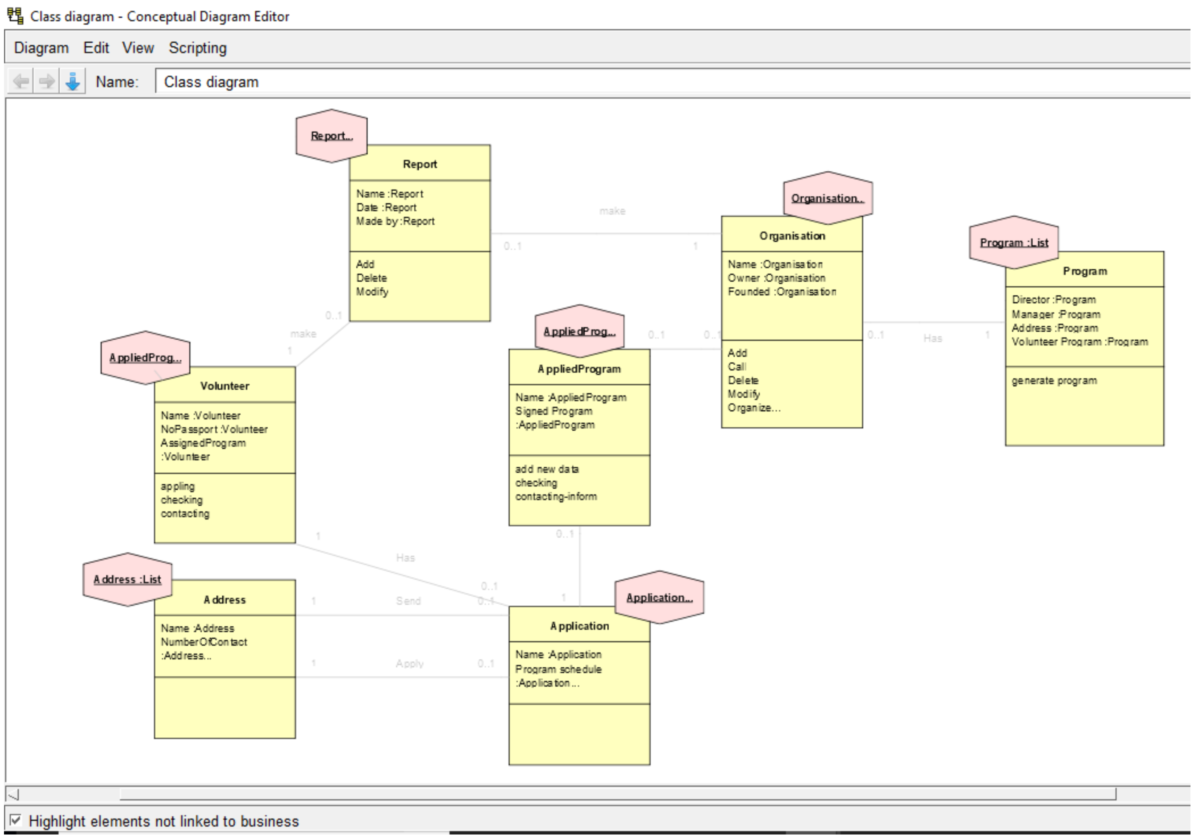
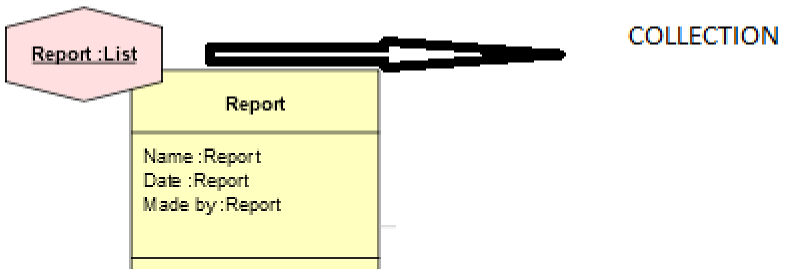


Figure 4.3.2. Conceptual diagram

Here, by Figure no what represent class diagram as UML model , we are able to see NoSQL design model, conceptual data model into logical data model. Here, entity relationships can be used in NoSQL



Modelling database (relational database and NoSQL)

- Entities with one to many relationships

One entitie has more relationships (one volunteer can apply on many programs)
One organization can have more the programs
One volunteer can have more addresses

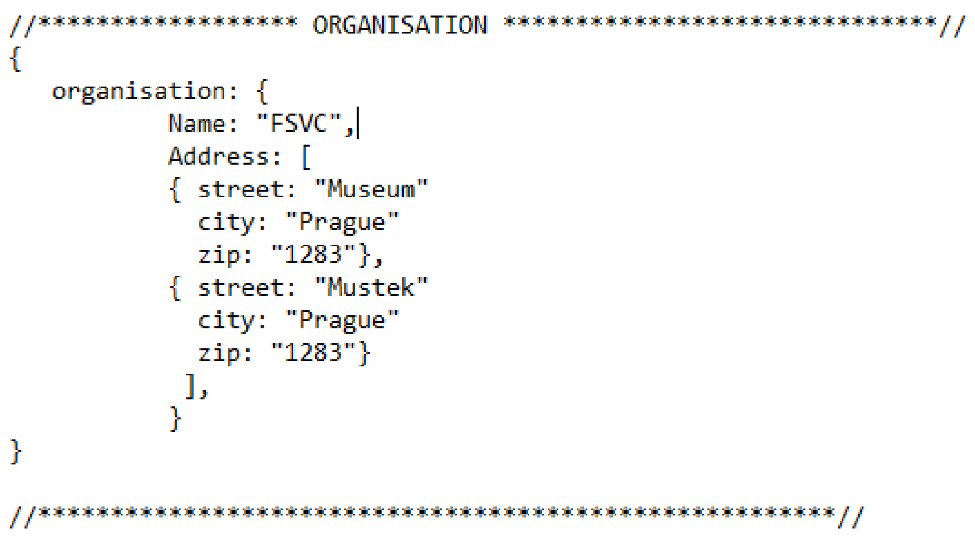
- Many entities can have many relationships

Volunteer can be involved into many Programs, but also Programs can have more members
Volunteer can make many Applications, but also Applications can have more members // example Applications

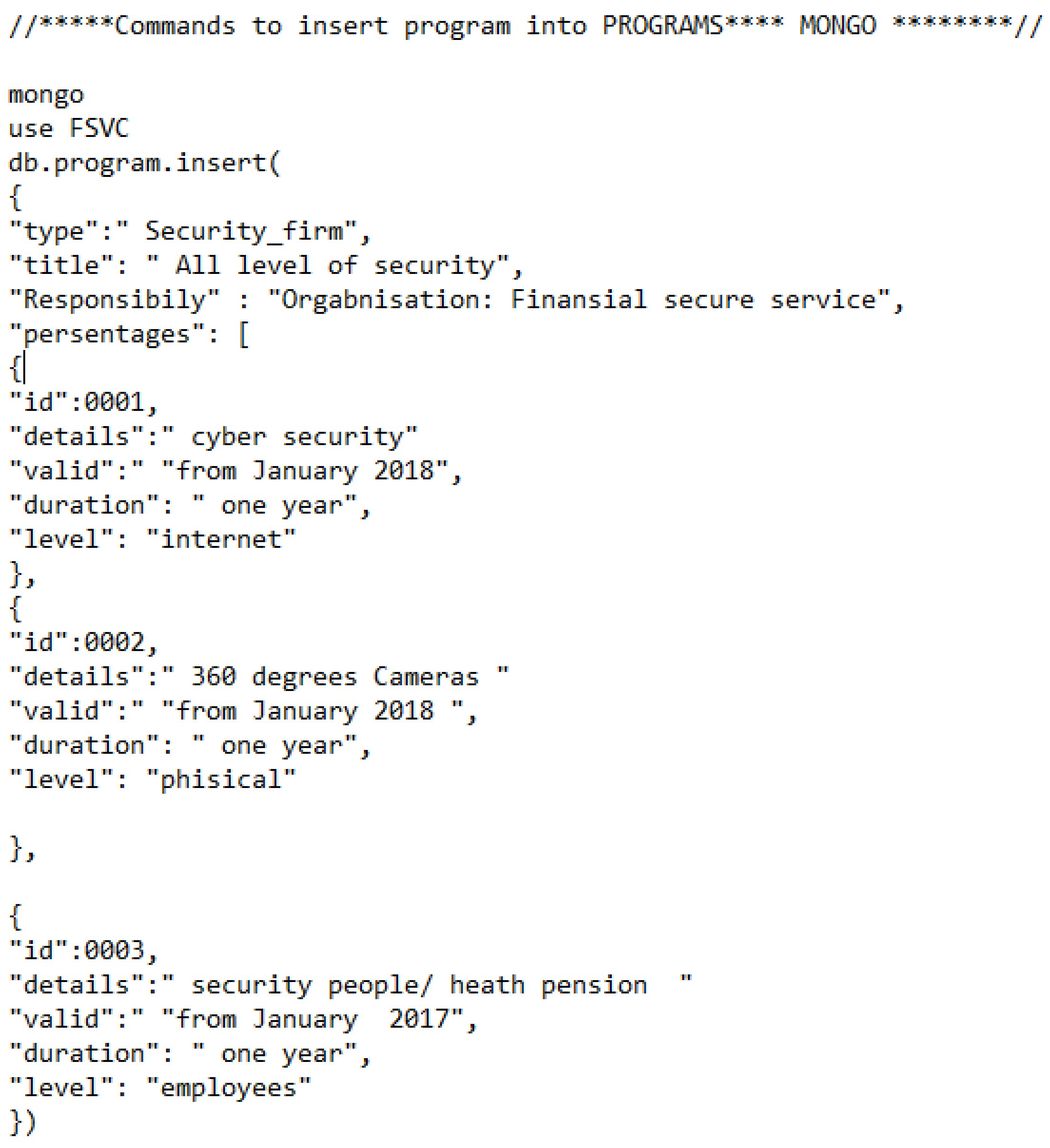
- Hierarchies

// what we can see here; usually during UML representation we are make separate one class what represent “Address” , but even during modeling relational database we, are not going to do that. In the table – entity , we are adding data about of address of the volunteer. Sometimes, we are able make it separate if our data about address are too much long – more details, more than one addresses, so we are adding new table with all description of address data.
Here, many entities are given as array of the addresses.
So, by using JSON are are able add/ insert into database the information about actors/ entities what are going to stored into database. Documents as “volunteer” , “organization”, “program” are made in mongo as bellow. “Address” is embedded into document “volunteer”, usually in SQL database , “address “ is separate entity.

```
//****************** VOLUNTEER ******************************//
{
   volunteer: {
           LastName: "Hodzic",
           FirstName:"Mirnesa",
           Address: [
           { street: "Kamycka"},
           { street: "IPPavlova"}
            ],
           }
}
```



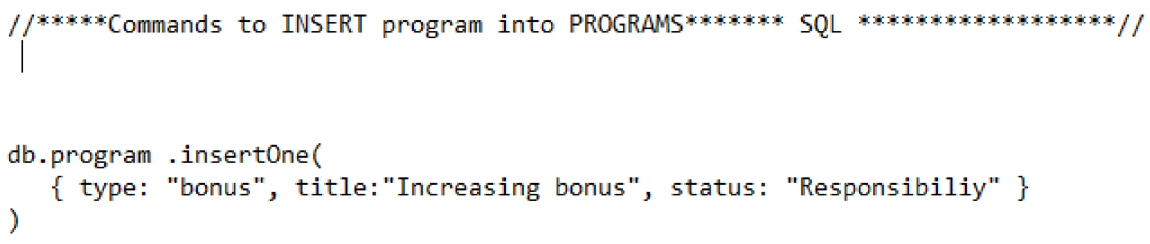
```
//****************** ORGANISATION ******************************//
{
   organisation: {
           Name: "FSVC",
           Address: [
           { street: "Museum"
             city: "Prague"
             zip: "1283"},
           { street: "Mustek"
             city: "Prague"
             zip: "1283"}
            ],
           }
}

//*******************************************************//
```



```
//*****Commands to insert program into PROGRAMS**** MONGO ********//

mongo
use FSVC
db.program.insert(
{
"type":" Security_firm",
"title": " All level of security",
"Responsibily" : "Orgabnisation: Finansial secure service",
"persentages": [
{|
"id":0001,
"details":" cyber security"
"valid":" "from January 2018",
"duration": " one year",
"level": "internet"
},
{
"id":0002,
"details":" 360 degrees Cameras "
"valid":" "from January 2018 ",
"duration": " one year",
"level": "phisical"

},

{
"id":0003,
"details":" security people/ heath pension  "
"valid":" "from January  2017",
"duration": " one year",
"level": "employees"
})
```

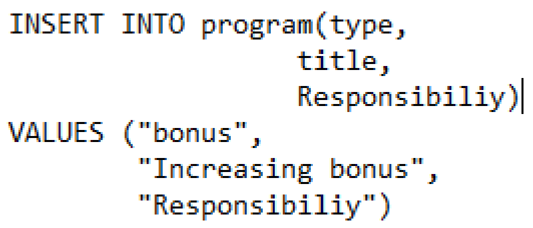
Similar part, embedded data is given above what in SQL is given as additional rows main table.
No, we can just add few words about this, first part
mongo
use FSVC
db.program.insert

means, mongo start and choose the FSVC database and insert like SQL, will insert data into FSVC database, be precisely into List of Programs.



```
//*****Commands to INSERT program into PROGRAMS******* SQL ******************//


db.program .insertOne(
   { type: "bonus", title:"Increasing bonus", status: "Responsibiliy" }
)
```

The same we are doing by using SQL command



```
INSERT INTO program(type,
                    title,
                    Responsibiliy)
VALUES ("bonus",
        "Increasing bonus",
        "Responsibiliy")
```

Result is the same. Given data will be put into database and saved.

Our goal is not just save data, also, our goal is manipulate with the data, take result for future analysis, what represent one of the main purpose of making and modeling data related to database. We are storing data because we will need it to check what is happening with users, result and also where we have succeed ideas.

Command as SELECT what is related to SQL database and FIND what is related JSON, we are able get list of all items what are stored in the database.
For example:
SQL;
SELECT * FROM program
MongoDB;
db.program.find()

To find just few columns of the data in purpose of future analysis

This is valid for SQL database :
SELECT title,
            Responsibility
FROM program

And this is way to perform the same result wit JSON
db.program.find ({},
                          {title: All level of security, Responsibility: Organisation: Finansial secure service })

The result of the code is list of the program based on title and Responsibility.

Also, one of ways, probably also the most used is
SQL database
SELECT title, Responsibility FROM program WHERE type=” bonus”

```
db.program.find( {type: ”bonus” },
                     {title: All level of security, Responsibility: Organisation: Finansial
secure service}
                    )
```

This is way how to get list of the bonuses based on the level of the bonus. Sometimes this is important part of selection.
SQL database
SELECT* FROM program where type=”bonus” ORDER BY level DESC
MONGO database JSON
db.program.find({ type=”bonus”}).sort ({level:-1})

Result is ordered in descaded way, from biggest up to lowest value. Sometimes, we want to see match result with order data, so we need it is this order.

Here, also, is given simple code to get list based on two parameters type or level so, code looks as (for SQL database, NoSQL database)
SELECT * FROM program WHERE type=”bonus” OR level=internet
db.program.find ($or: [{ type=”bonus”},
{level:”internet”}]})

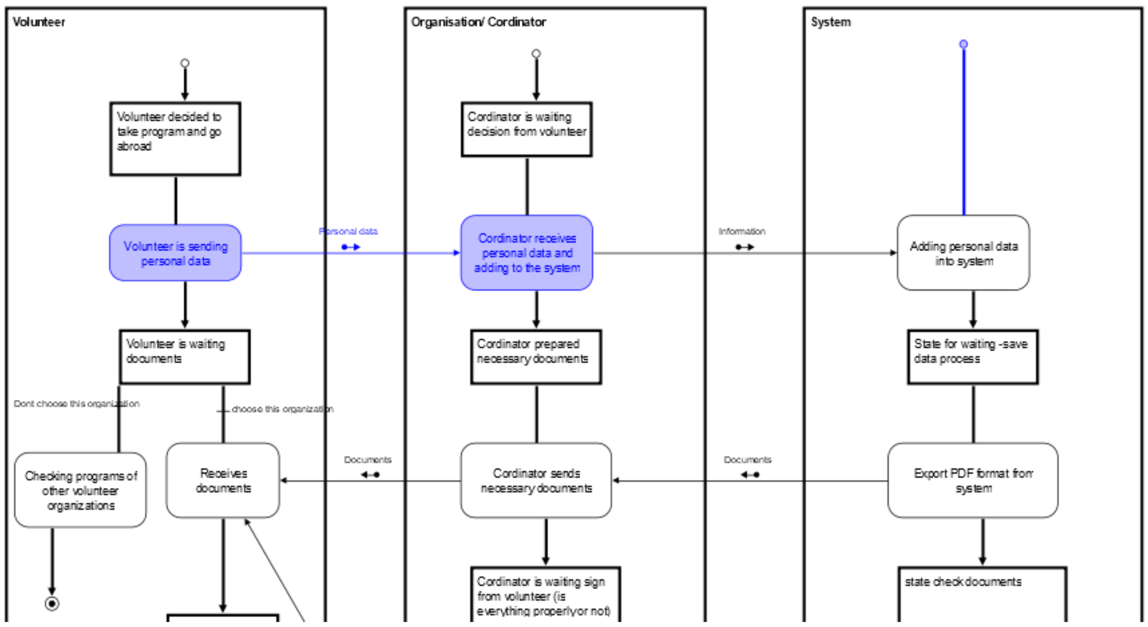
5 Results and Discussion

5.1 Business Architecture

Business architecture help us to understand organisation and connection between business model and business functionality. This is the best way to see what is goal of the business, to make questions and answer on it (who, why , where and how). In our case, it is visible that goal of the organisation is support institutions with regulations/ vision and taktics during work with as initiatives/ project activity. Using strategy, operations and technology we are able to get employee / volunteer who will provide own working hours to help and support others. Goal of the organisation have good plan/ strategy to make program and together with government you are able make connections and implement our tasks. Data are stored, and will be helpful for next project.
Here, what we should mentioned, goal is get good volunteer who will own knowledge give us, so, experts are bringing project based on own experience from financial company.

5.2 Business Process Modelling

Business simulations show as flow of our reactions and making decision. If we see figure below, we are able translate it into code. It is not just about making database, it is also about data structure and information what are needed for our activities. Sending information – decision means daily send emails what occupied our email server. Every new info is going to be added into database, or with every changes our database is going to be updated with action update, modify and delete. Here, we will have whole system what if our volunteer accept our program or not, which kind action we have to take in next step. If we don't have enough volunteer for current project we are getting alarm what it means more activities to get people who are ready spend own time and share experience and knowledge to support and help others. Our actors are volunteer, organisation and system as database.
To be sure, also, communications between organisation and volunteer occupied our server resources because we are going to add every potentially volunteer if is possible, update data with new contact, update with date of contact and date of response and so on. That information even are not important for share knowledge but are important for future analysis and connections.

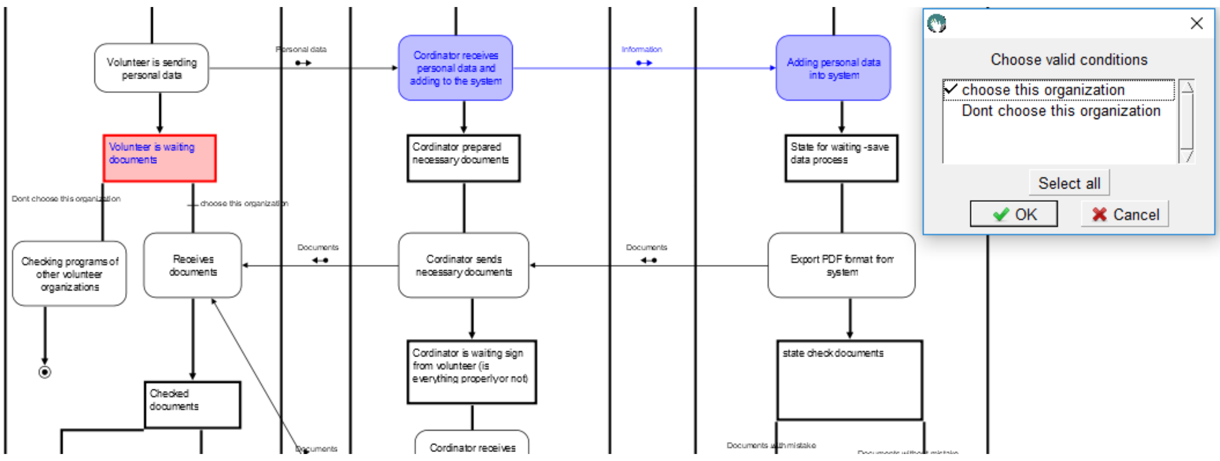


Volunteer
Organisation/ Cordinator
System
Volunteer decided to take program and go abroad
Cordinator is waiting decision from volunteer
Volunteer is sending personal data
Personal data
Cordinator receives personal data and adding to the system
Information
Adding personal data into system
Volunteer is waiting documents
Cordinator prepared necessary documents
State for waiting -save data process
Dont choose this organization
choose this organization
Checking programs of other volunteer organizations
Receives documents
Documents
Cordinator sends necessary documents
Documents
Export PDF format from system
Cordinator is waiting sign from volunteer (is everything properly or not)
state check documents

Figure 5.2.1. Business Process Model – Decision_1

After every actions/ steps next stage is going to be activated and finished, up to moment where we added condition of our actions what make our activities more filtered and more precialy to reach goal.

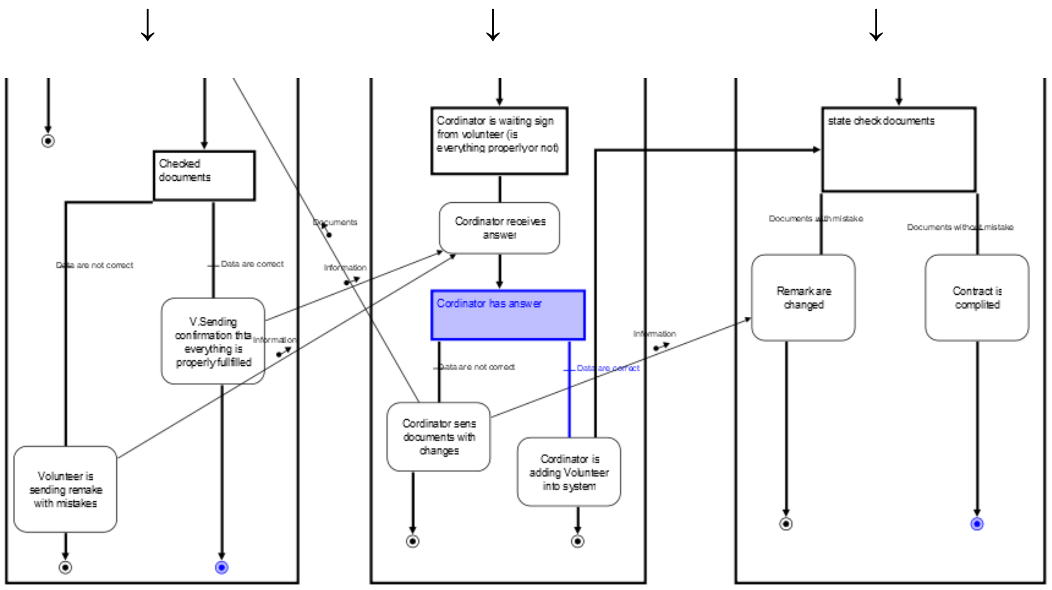
↓ ↓ ↓



Volunteer is sending personal data
Personal data
Cordinator receives personal data and adding to the system
Information
Adding personal data into system
Volunteer is waiting documents
Cordinator prepared necessary documents
State for waiting -save data process
Dont choose this organization
choose this organization
Checking programs of other volunteer organizations
Receives documents
Documents
Cordinator sends necessary documents
Documents
Export PDF format from system
Cordinator is waiting sign from volunteer (is everything properly or not)
state check documents
Checked documents
Documents
Cordinator receives
Documents with mistake
Choose valid conditions
✓ choose this organization
Dont choose this organization
Select all
OK
Cancel

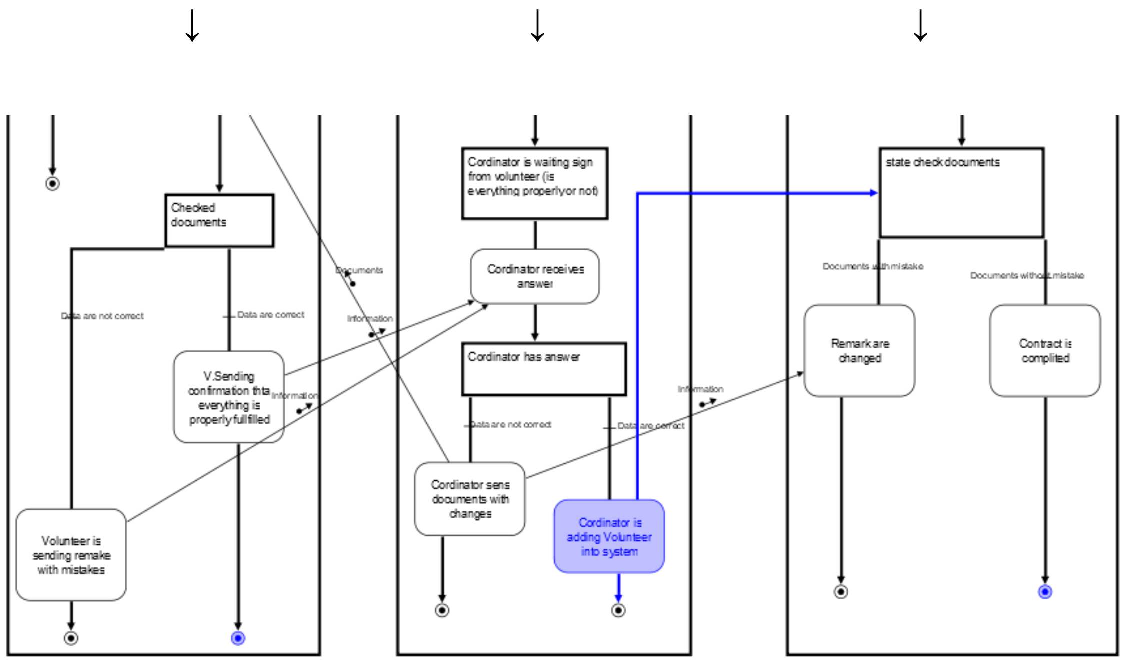
Figure 5.2.2. Business Process Model – Decision_2

In the figure above, next step depends on our decision, and it means all other actors are waiting for decision if their activities depends on that decision. This is good way to make code using JSON and implement NoSQL database.



Checked documents
Data are not correct
Data are correct
V.Sending confirmation thta everything is properly fullfilled
Information
Volunteer is sending remake with mistakes
Documents
Information
Cordinator is waiting sign from volunteer (is everything properlyor not)
Cordinator receives answer
Cordinator has answer
Data are not correct
Data are correct
Cordinator sens documents with changes
Cordinator is adding Volunteer into system
Information
state check documents
Documents with mistake
Documents without mistake
Remark are changed
Contract is complited

Figure 5.2.3. Business Process Model – Decision_3



Checked documents
Data are not correct
Data are correct
V.Sending confirmation thta everything is properly fullfilled
Information
Volunteer is sending remake with mistakes
Documents
Information
Cordinator is waiting sign from volunteer (is everything properlyor not)
Cordinator receives answer
Cordinator has answer
Data are not correct
Data are correct
Cordinator sens documents with changes
Cordinator is adding Volunteer into system
Information
state check documents
Documents with mistake
Documents without mistake
Remark are changed
Contract is complited

Figure 5.2.4. Business Process Model – Decision_4

After process of adding new information we are going to update our database so process of it is given as :

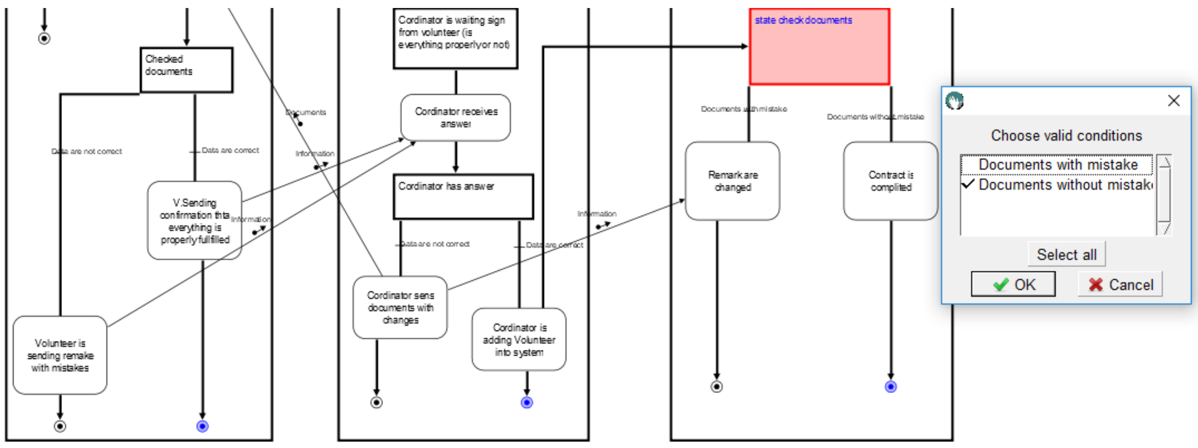

Checked documents
Data are not correct
Data are correct
V.Sending confirmation thta everything is properly fullfilled
Information
Volunteer is sending remake with mistakes
Documents
Cordinator is waiting sign from volunteer (is everything properlyor not)
Cordinator receives answer
Cordinator has answer
Data are not correct
Data are correct
Cordinator sens documents with changes
Cordinator is adding Volunteer into system
state check documents
Documents with mistake
Documents without mistake
Remark are changed
Contract is complited
Choose valid conditions
Documents with mistake
Documents without mistake
Select all
OK
Cancel

Figure 5.2.5. Business Process Model – Decision_5

↓ ↓ ↓

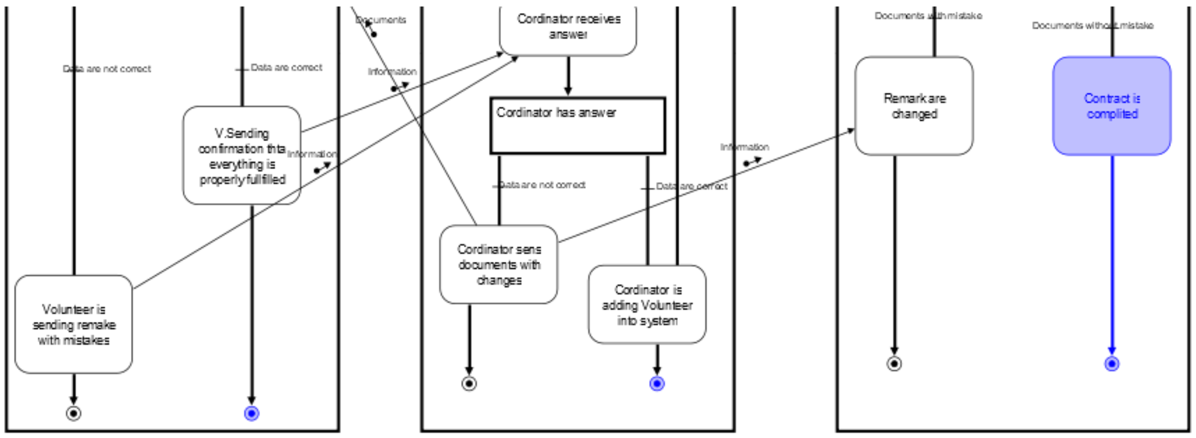

Data are not correct
Data are correct
V.Sending confirmation thta everything is properly fullfilled
Information
Volunteer is sending remake with mistakes
Documents
Information
Cordinator receives answer
Cordinator has answer
Data are not correct
Data are correct
Cordinator sens documents with changes
Cordinator is adding Volunteer into system
Information
Documents with mistake
Documents without mistake
Remark are changed
Contract is complited

Figure 5.2.6. Business Process Model – Decision_6

Process is finished in this field communications – adding updating and modify our information.

Process making reports represent example how our NoSQL can be embedded with so many information keeping just one document. We are able making more projects based on one program in different conties and be supported by many volunteers. Also, that can be done this year, 2 years ago or more. All information can be store in one document by using embedded data as arrays. Process is descriebed by figures below

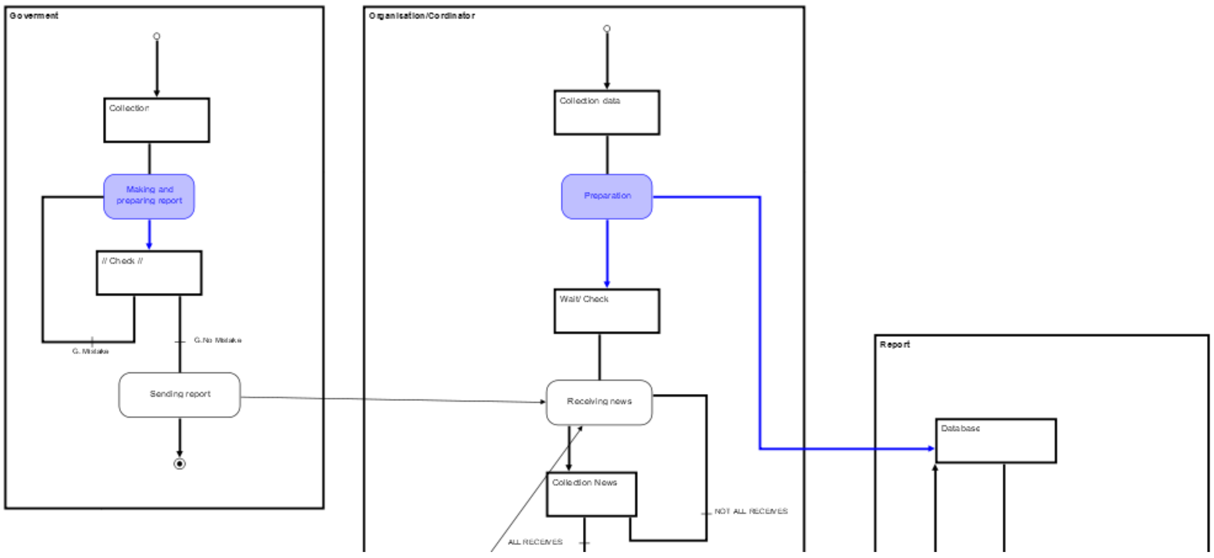


Figure 5.2.7. Business Process Model – Report_1

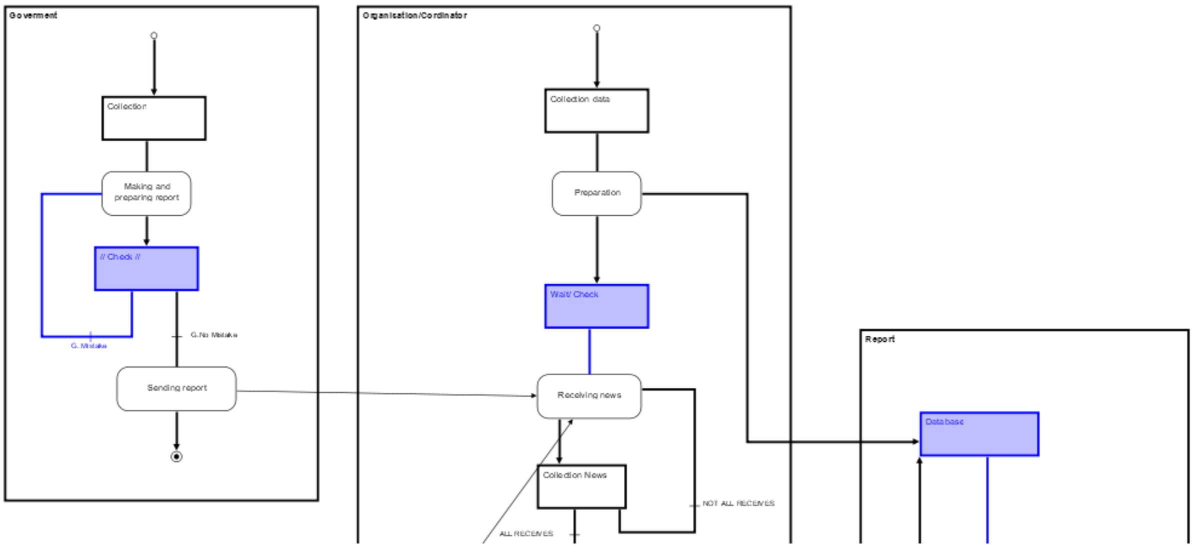


Figure 5.2.8. Business Process Model – Report_2

↓ ↓ ↓

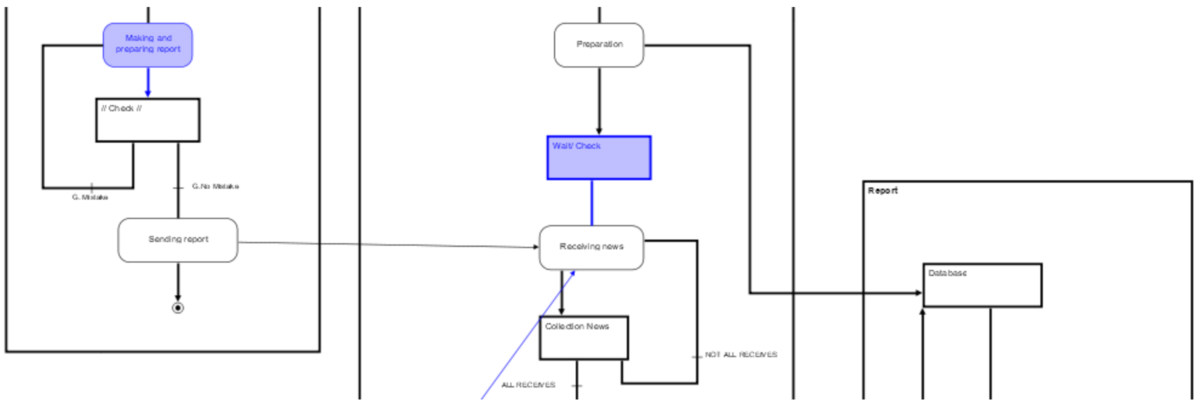


Figure 5.2.9. Business Process Model – Report_3

↓ ↓ ↓

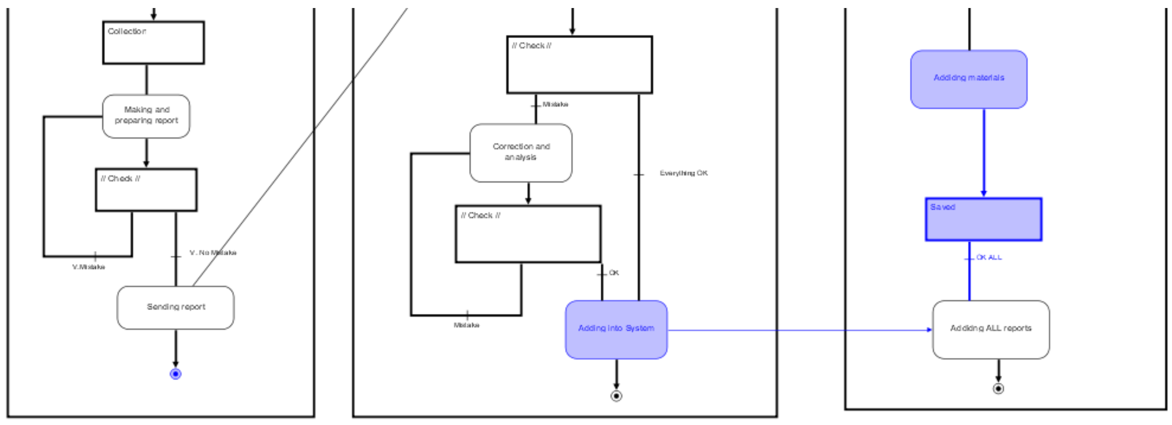


Figure 5.2.10. Business Process Model – Report_4

↓ ↓ ↓

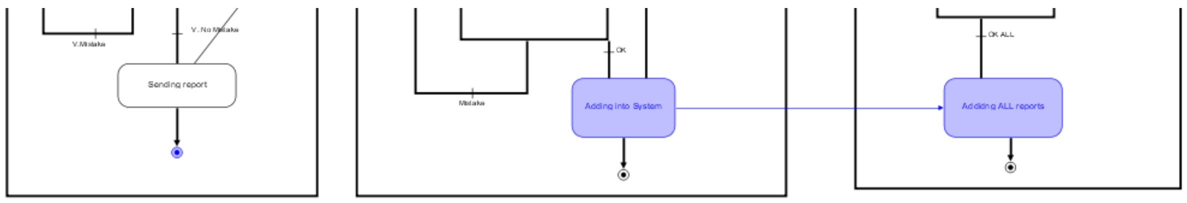
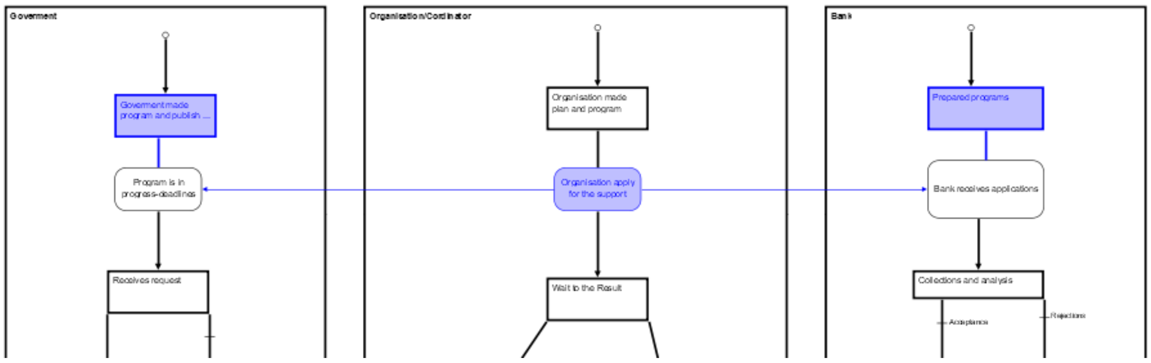


Figure 5.2.11. Business Process Model – Report_5

Ours reports is going to be updated after every new information, so, if one of our actor make mistake, loop is going to be repeat up to condition does get true information, and than continue with next step. Repeating loops are given is condition of the mistake as false.

The organisation exist thanks by relationships with government and institutions, so our chatflow are given as :



Goverment
Organisation/Cordinator
Bank
Goverment made program and publish ...
Organisation made plan and program
Prepared programs
Program is in progress-deadlines
Organisation apply for the support
Bank receives applications
Receives request
Wait to the Result
Collections and analysis
Acceptance
Rejections

Figure 5.2.12. Business Process Model – Relationships_1

After acceptance from both side

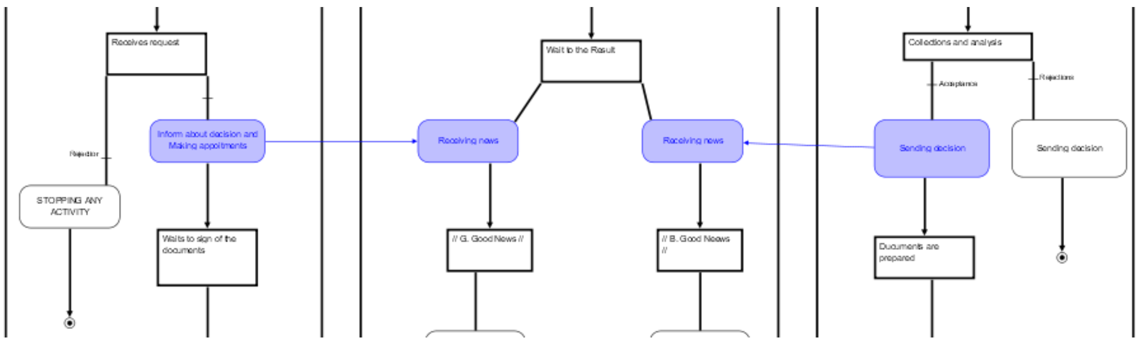
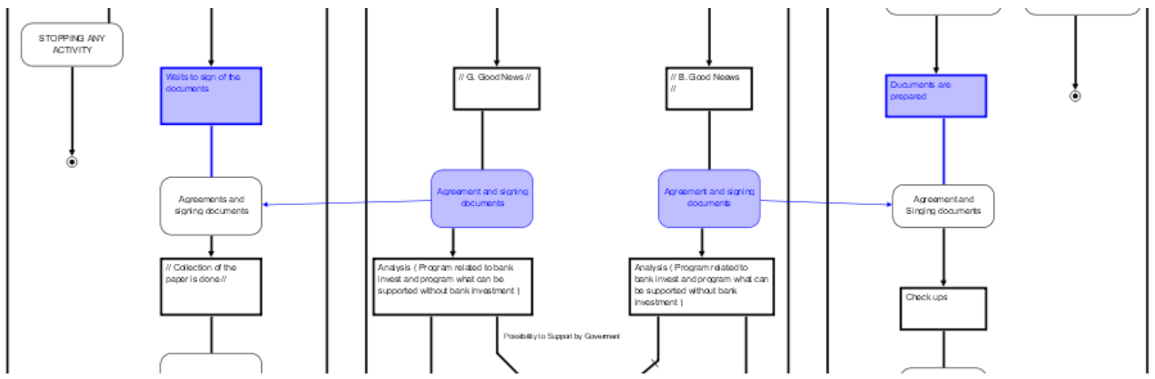


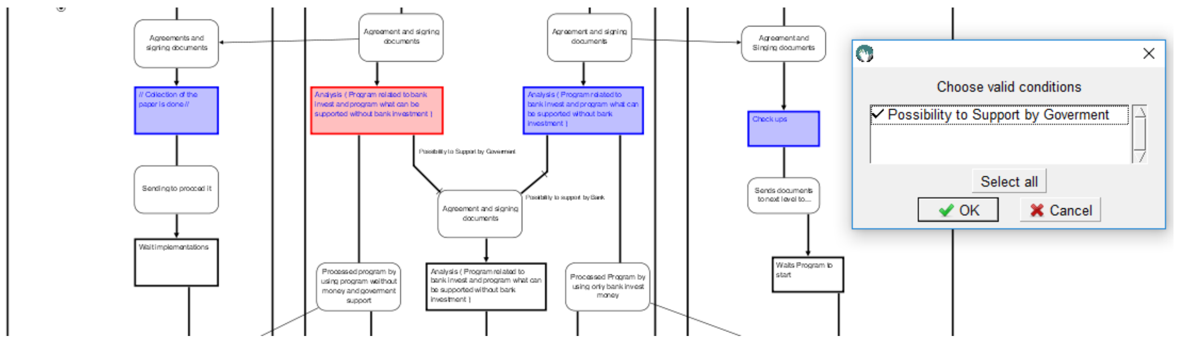
Figure 5.2.13. Business Process Model – Relationships_2

We are in the position to make agreement



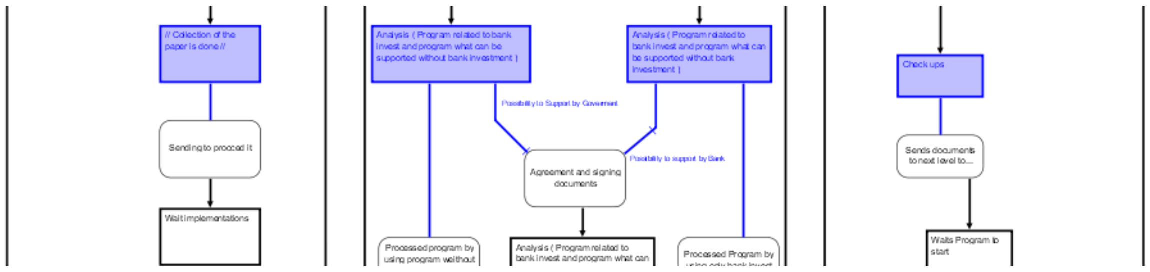
STOPPING ANY ACTIVITY
Waits to sign of the documents
Agreements and signing documents
// Collection of the paper is done //
// G. Good News //
Agreement and signing documents
Analysis (Program related to bank invest and program what can be supported without bank investment)
Possibility to Support by Goverment
// B. Good News //
Agreement and signing documents
Analysis (Program related to bank invest and program what can be supported without bank investment)
Documents are prepared
Agreement and Singing documents
Check ups

Figure 5.2.14. Business Process Model – Relationships_3



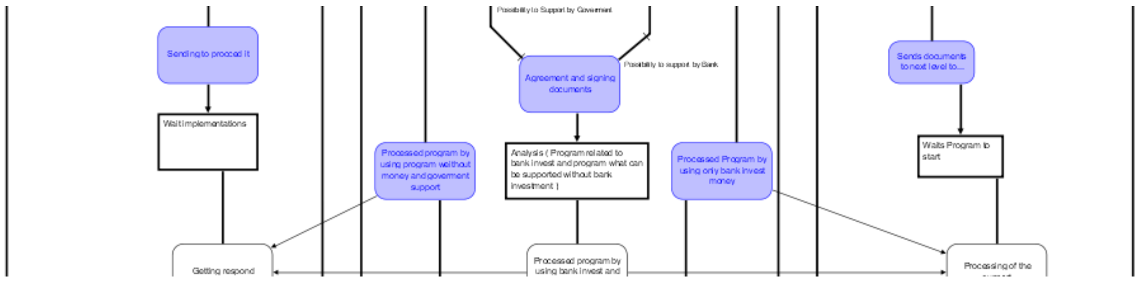
Agreements and signing documents
// Collection of the paper is done //
Sending to proceed it
Wait implementations
Agreement and signing documents
Analysis (Program related to bank invest and program what can be supported without bank investment)
Possibility to Support by Goverment
Agreement and signing documents
Analysis (Program related to bank invest and program what can be supported without bank investment)
Possibility to support by Bank
Agreement and signing documents
Processed program by using program without money and goverment support
Analysis (Program related to bank invest and program what can be supported without bank investment)
Processed Program by using only bank invest money
Agreement and Singing documents
Check ups
Sends documents to next level to...
Waits Program to start
Choose valid conditions
Possibility to Support by Goverment
Select all
OK
Cancel

Figure 5.2.15. Business Process Model – Relationships_4



// Collection of the paper is done //
Sending to proceed it
Wait implementations
Analysis (Program related to bank invest and program what can be supported without bank investment)
Possibility to Support by Goverment
Analysis (Program related to bank invest and program what can be supported without bank investment)
Possibility to support by Bank
Agreement and signing documents
Processed program by using program without
Analysis (Program related to bank invest and program what can
Processed Program by
Check ups
Sends documents to next level to...
Waits Program to start

Figure 5.2.16. Business Process Model – Relationships_5



Possibility to Support by Goverment
Sending to proceed it
Wait implementations
Agreement and signing documents
Possibility to support by Bank
Processed program by using program without money and goverment support
Analysis (Program related to bank invest and program what can be supported without bank investment)
Processed Program by using only bank invest money
Sends documents to next level to...
Waits Program to start
Getting respond
Processed program by using bank invest and
Processing of the

Figure 5.2.17. Business Process Model – Relationships_6

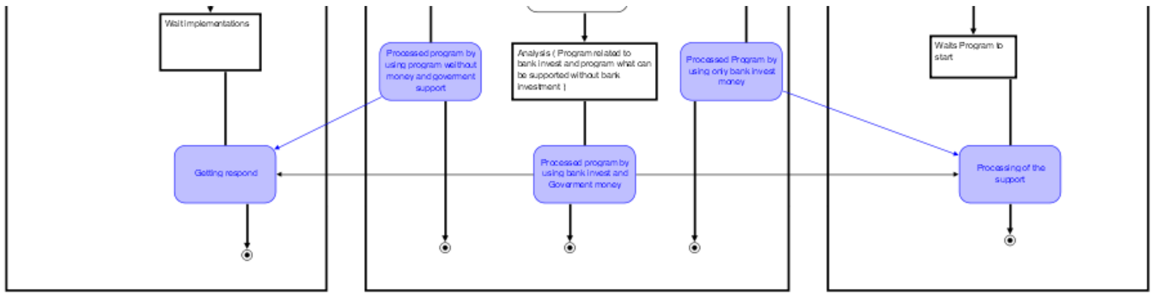
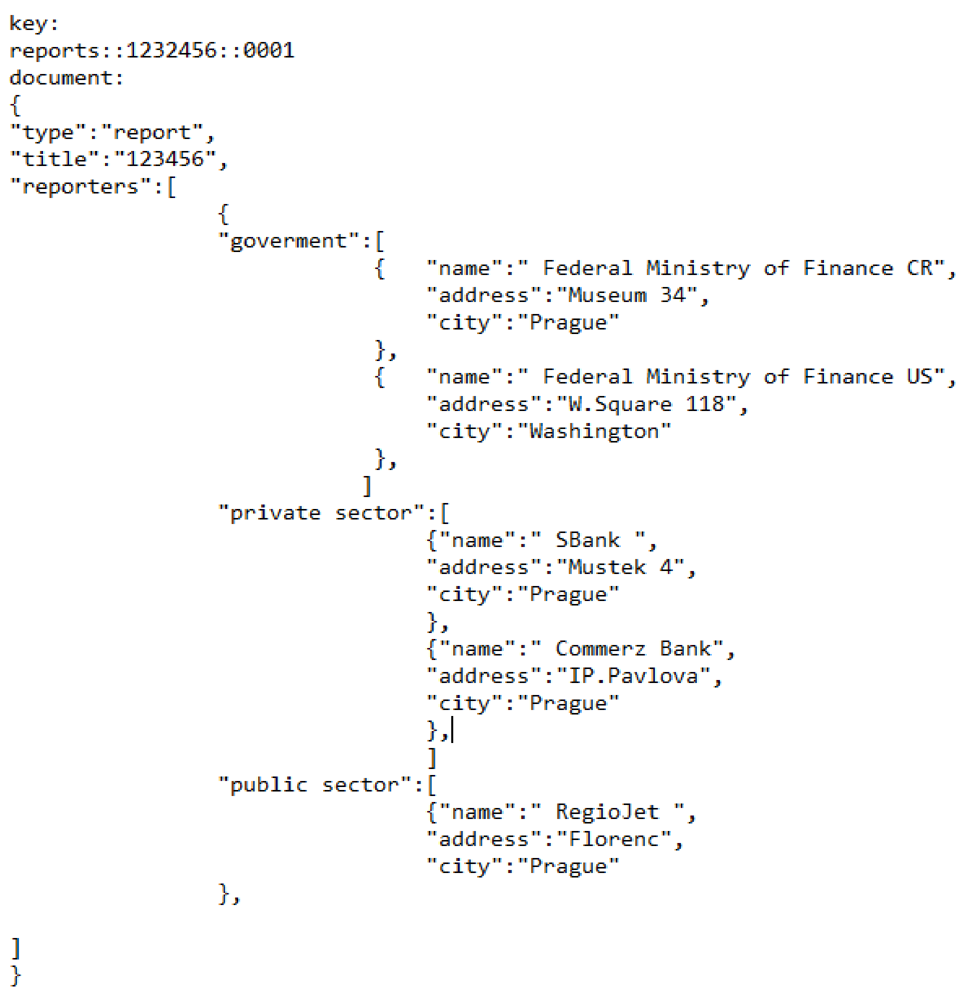

Wait implementations
Processed program by using program without money and goverment support
Analysis (Program related to bank invest and program what can be supported without bark investment)
Processed Program by using only bank invest money
Waits Program to start
Getting respond
Processed program by using bank invest and Goverment money
Processing of the support

Figure 5.2.18. Business Process Model – Relationships_7

The steps above that are related to process given in figures relationships government, organisation and bank show as posibilities of support sectors depends our our income. Last steps should be made in details, how much freedom do we have to support sectors based on field of business. Probably, something what has importance for public goods, it will take first place and more money.

5.3 Conceptual Diagram

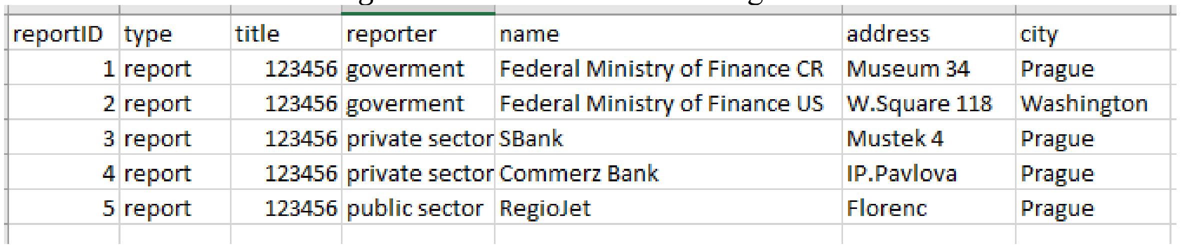
Conceptual diagram is one of ways to show us how to move NoSQL database into relational environment and keep all benefits what NoSQL database has compared to relational database like scalability, availability, speed, performance and so on. Relational database is good for so many business architecture, but in the period of digitalisation where we don't want to use money and where is everything trucked and stored, we need something what can be used with more functionalities as adding new information without changes structure/ architecture of the business. That is NoSQL database very useful, because it is possible very easy embedded new information into already existing documents. Data structure is very important part what will be focus of research as normalisation and optimisation of relationships between data. One of the most important part is bank transaction, how to optimize and make model of the data during transaction, make it independent and keep all information before and after transactions without making less problems in fact of the speed transfer data throw multiply filesystems and servers. NoSQL database allows us to make model od our data how we want, without too much spending time where it is going keeping our goal to provide good service to institutions. Thanks by flexibility of making nested elements (arrays and objects) , we are able to make embedded document depends on the current situation, stored it as group of the document or stored our data as individual documents. One of example, there is document of the report where are nested all reports about one project depends on the volunteer, organisation and government. Just only one documents. In rational database, it will be plus additional tree tables.



```
key:
reports::1232456::0001
document:
{
"type":"report",
"title":"123456",
"reporters":[
                {
                "goverment":[
                            {   "name":" Federal Ministry of Finance CR",
                                "address":"Museum 34",
                                "city":"Prague"
                            },
                            {   "name":" Federal Ministry of Finance US",
                                "address":"W.Square 118",
                                "city":"Washington"
                            },
                           ]
                "private sector":[
                                {"name":" SBank ",
                                "address":"Mustek 4",
                                "city":"Prague"
                                },
                                {"name":" Commerz Bank",
                                "address":"IP.Pavlova",
                                "city":"Prague"
                                },
                                ]
                "public sector":[
                                {"name":" RegioJet ",
                                "address":"Florenc",
                                "city":"Prague"
                },

]
}
```

Figure 5.3.1. Code JSON

Relational database will be given how it's shown in the Figure no.5.3.2.



reportID	type	title	reporter	name	address	city
1	report	123456	goverment	Federal Ministry of Finance CR	Museum 34	Prague
2	report	123456	goverment	Federal Ministry of Finance US	W.Square 118	Washington
3	report	123456	private sector	SBank	Mustek 4	Prague
4	report	123456	private sector	Commerz Bank	IP.Pavlova	Prague
5	report	123456	public sector	RegioJet	Florenc	Prague

Figure 5.3.2. SQL table

So, conclusion is that we are able manage NoSQL in relational – oriented environment but we have to first deeply research data structure what is going to be used in our business. Adding new information can be easy as embedded arrays, but we have to very careful to keep consistency of the data for future analysis.

6 Conclusion

The data structure are important part of analysis and also important part during process of choosing right database for our business. Making connections, correlations and dependency between data, we are able by using techniques of normalisation make our database in advance level, optimized and faster. Also, here is shown that we are able no relational NoSQL database move into relational database based on data model. Right now, there is no one model what can be the best option for every database, it doesn't matter it is relational or not. In this papers, it is described MongoDB what is the most closer to the relational database and mostly used for store data of ecommerc systems. Also, in the papers are included business model to shown which kind activities are going to operate with database.

Work with database it doesn't means just sore data as operation delete, update, insert and so on, also, we are storing data to use them after some period. The bank system doesn't go to deeply analysis as some institution what are main job to give as statistics about our world, people and environment including materials what are necessary for our life. During that analysis we can expect more complicated manipulations with data.

My opinion is that MongoDB as other NoSQL databases will find more will find more usage area than now, so, that is open question specially in transaction what are happening every moment in billions number. Make our needs faster with keep consistency of data and less consummation of the storage. The data structure and model what is applicable for every type of database is still open question. Here, is just given one example how look work with both and which kind operation are usually used during work in FVSC.

7 References

1. The NoSQL Ecosystem, Adam Marcus, aosabook.org[online], http://www.aosabook.org/en/nosql.html

2. Five Steps For Choosing and Implementing a Database - Michelle Regal and Laura Quinn, June 15, 2015. NTEN [online], https://www.nten.org/article/five-steps-for-choosing-and-implementing-a-database/
3. IBM Knowledge Center, IBM [online], https://www.ibm.com/support/knowledgecenter/SSEPGG_10.1.0/com.ibm.db2.luw.admin.perf.doc/doc/c0005418.html

4. Tops for Selection the Right Database for your APP -James Higginbotham, Jul 29, 2016. DZONE [online], https://dzone.com/articles/3-tips-for-selecting-the-right-database-for-your-a
5. Financial Services Volunteer Corps – FSVC, FSVC [online], https://www.fsvc.org/adam-szubin-and-eunice-panetta-elected-to-fsvcs-board-of-directors/
6. MongoDB – MongoDB, Cristian Amor Kvalheim, MongoDB[online], http://mongodb.github.io/node-mongodb-native/schema/chapter1/
7. What is business architecture – STAGROUP, 2018, STA GROUP [online] http://www.stagrp.com/architecture/business-architecture/what-is-business-architecture/
8. SQL Database – W3schools , W3School [online] https://www.w3schools.com/sql/sql_create_table.asp
9. Introduction to LINQ - Granville Barnett, September 15, 2007 .KIRUPA.COM [online] https://www.kirupa.com/net/next_gen_data_access_linq_pg3.htm
10. Entity Relationship Diagram – SmartDraw, SmartDraw.com [online] https://www.smartdraw.com/entity-relationship-diagram/
11. DB2 – Datbase tutorialspoint [online] https://www.tutorialspoint.com/db2/db2_databases.htm
12. Baze podataka, Nebojsa Lukic dipl.ing.el, Avgust 2007 Bijeljina, Tehnicka skola Mihajlo Pupin[online] http://tehnicari.weebly.com/uploads/3/6/3/6/3636418/baze_3.pdf
13. Baze podataka, Robert Manger, Rujan 2003 Zagreb, Sveuciliste u Zagrebu Prirodno matematicki fakultet, [online], http://jadran.izor.hr/~dadic/EKO/baze-podataka.pdf
14. Normalizacija, EdukacijaRS, edukacija.rs [online], http://edukacija.rs/it/baze-podataka/normalizacija-relacija
15. NoSQL Architecture, Kris Zyp, May 11, 2010, Sitepen [online], https://www.sitepen.com/blog/2010/05/11/nosql-architecture/
16. An Object-Oriented Database Model for Business Entities and Process, Michael Missikoff and Robert Pizzicannella, October 1998, ERCIM NEWS,[online], https://www.ercim.eu/publication/Ercim_News/enw35/missikoff.html

8 Appendix

List of Supplements…