Table of Contents

[Introduction: 2](#_Toc81836446)

[Requirement 3](#_Toc81836447)

[1.1 Scenario: 3](#_Toc81836448)

[1.2 User groups and Business needs: 3](#_Toc81836449)

[1.3System requirements: 3](#_Toc81836450)

[2 design 4](#_Toc81836451)

[2.1.1 Conceptual Schema: 4](#_Toc81836452)

[2.1.2 Logical Schema: 5](#_Toc81836453)

[2.1.3Physical Schema: 6](#_Toc81836454)

[2.2 Mapping: 7](#_Toc81836455)

[2.3 Normalization: 7](#_Toc81836456)

[First normal form: 7](#_Toc81836457)

[Second normal form: 9](#_Toc81836458)

[3 Development: 10](#_Toc81836459)

[3.1 create database, tables, insert: 10](#_Toc81836460)

[3.1.1 create database: 10](#_Toc81836461)

[3.1.2 tables: 11](#_Toc81836462)

[3.1.3 insert: 15](#_Toc81836463)

[3.2 create users and give privileges + view and procedures 19](#_Toc81836464)

[3.2.1 view: 19](#_Toc81836465)

[3.2.1procedures: 22](#_Toc81836466)

[3.2.2 create users: 22](#_Toc81836467)

[3.2.2 privileges: 23](#_Toc81836468)

[4 user and technical documentation: 26](#_Toc81836469)

[4.1 User documentation: 26](#_Toc81836470)

[4.2 Technical documentation: 30](#_Toc81836471)

[5 test the system against user and system requirements + output and input validation: 43](#_Toc81836472)

[output and input validation: 43](#_Toc81836473)

[Test the created views and see if they work correctly: 45](#_Toc81836474)

[Test the created procedures and see if they work correctly: 45](#_Toc81836475)

[Test the created Users and see if they work correctly: 46](#_Toc81836476)

[How to run a wamp server 64 program? 47](#_Toc81836477)

[A test to show the data contained in the database: 47](#_Toc81836478)

[Test the system against user and system requirements: 48](#_Toc81836479)

[6 database maintenance: 49](#_Toc81836480)

[7 interface and output designs: 50](#_Toc81836481)

[8 Diagrams and flowcharts: 52](#_Toc81836482)

[9 Evaluate the effectiveness of the design in relation to user and system requirements. 54](#_Toc81836483)

[10 Evaluate the effectiveness of the database solution in relation to user and system requirements, and suggest improvements. 55](#_Toc81836484)

[11 Evaluate the database in terms of improvements needed to ensure the continued effectiveness of the system. 55](#_Toc81836485)



Name: Hasan Alhwietat. Id: 20120098. Specialization: Computer science.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Introduction:

I'm employed as an assistant database developer for a large IT consulting firm. The company wanted to create a database for a school.

Define the database:

It is an organized set of structured information or data, that is usually stored electronically in a computer system. The database is usually controlled by the DBMS.

Schema definition:

The design of the database is called the scheme. The scheme is of three types: (physical schema, Logical schema and conceptual scheme).

There are three types of Normal Forms are:

First Normal Form (**1st NF**).

Second Normal Form (**2nd****NF**).

Third Normal Form (**3rd NF**).

# Requirement

## 1.1 Scenario:

school database

The database keeps the information about the school, like (Id, name, email, building num\_address, city\_address). many teachers work in the school, the teacher teaches many courses. the school has many classrooms. Each classroom has more than one course and the course is given in one room. A student can take many courses. One course is taken by more than one student. the course has many assignments. Each student has many assignments. The database keeps the information about teachers, such as (id, f\_name, l\_name, phones, city\_address, street name\_address, email). The specified student information is kept in the database (ID, name, l\_name, city\_address, street\_address, building address\_number, health status, date of birth, phones). Also, classroom information is kept in the database (ID, room name, room capacity). The database keeps the information about courses, like (ID, name). The database keeps the information about assignments, such as (id, name the assignment, assignment tag)

## 1.2 User groups and Business needs:

1. Teachers: storing their data They can browse information about their students, ask them for homework, and send their grades to Administrative staff or download over the school website, and know the name of the school in which he teaches.
2. Students: can choose the teacher they want and can the courses schedule either over the phone or using the school's website, and know the name of the school in which he studies. also should be able to access the database using a web browser to request grades and submit assignments.
3. root: He can do all the things he wants from seeing the number of students in the school, the number of teachers, the subjects that are given in this school and the first ones who got the highest marks.

## 1.3System requirements:

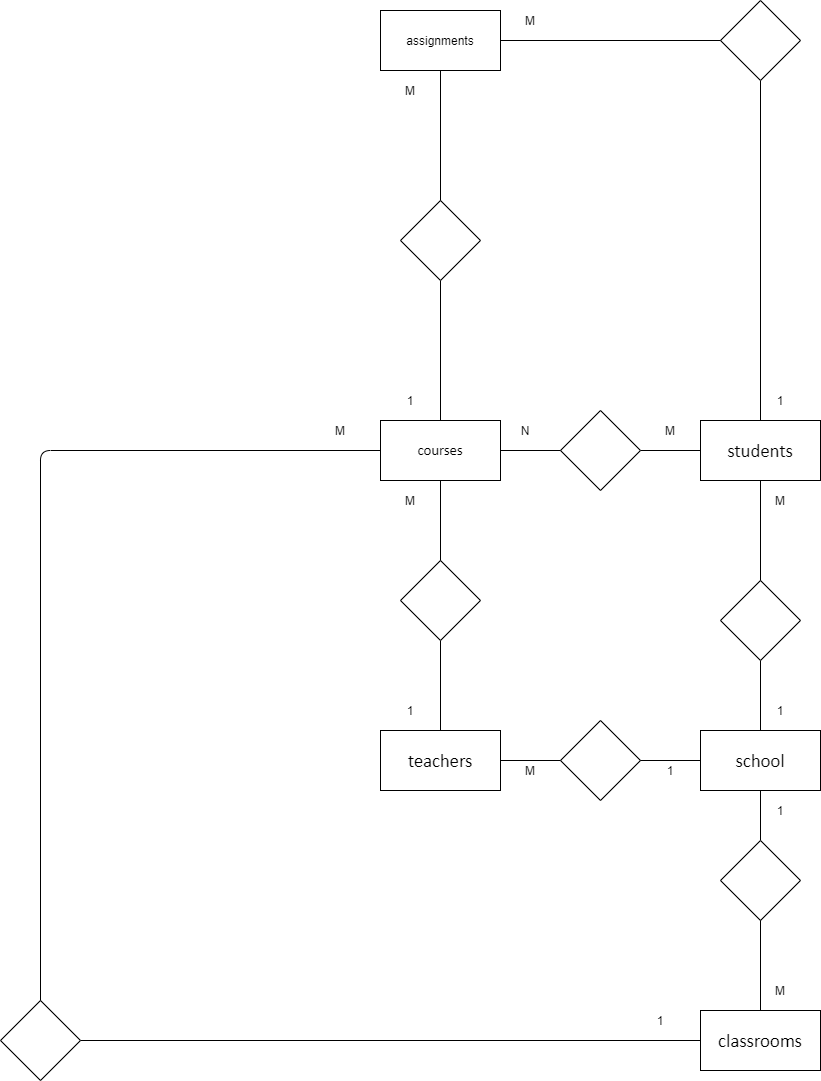
1-Many computers or laptops are required for teachers and administration to store data.

2-The building must have an internet connection to save data related to the school on its official website.

# 2 design

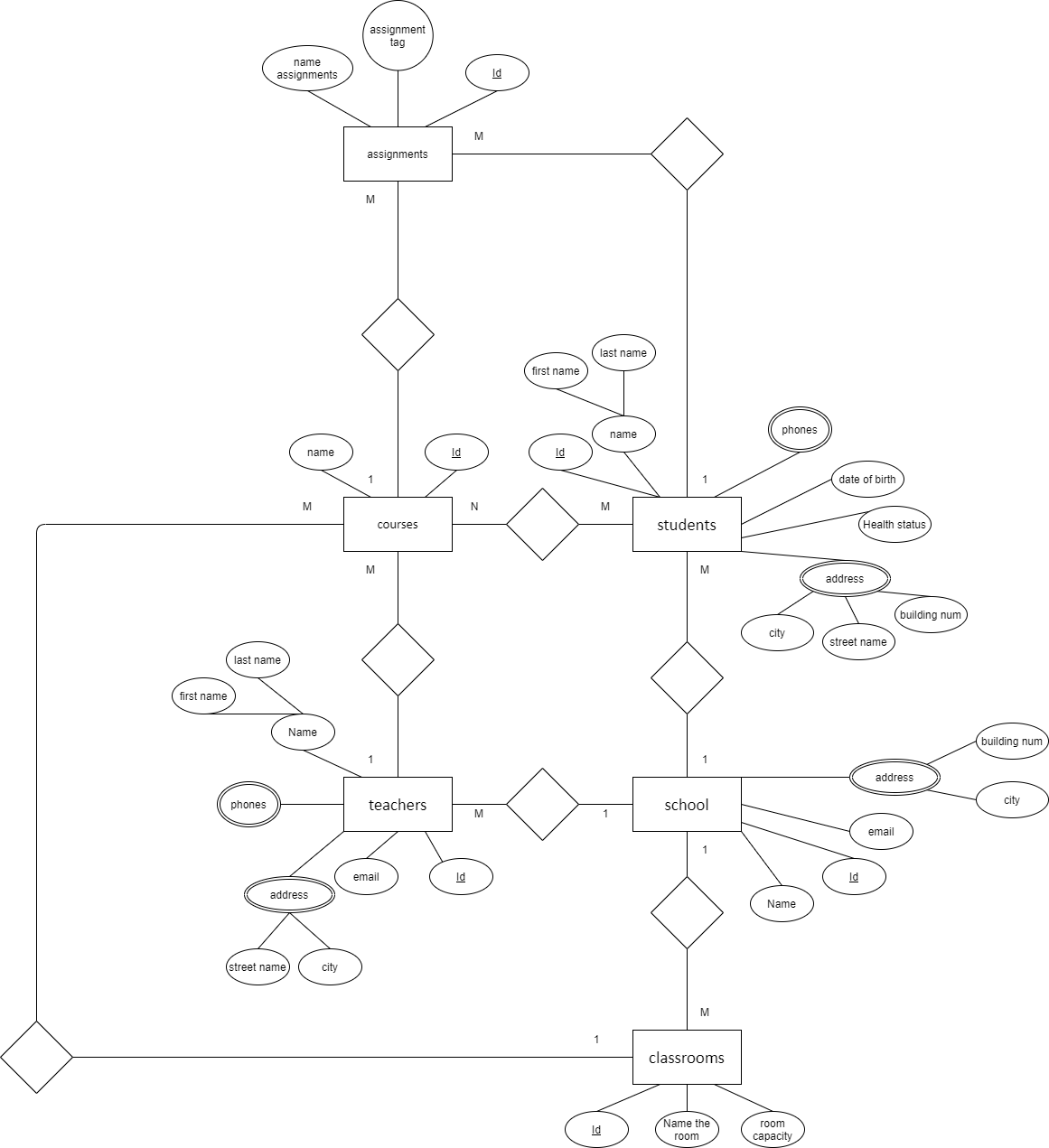
## 2.1.1 Conceptual Schema:

A model that develops the meaning of the basic concepts of a problem area, defines its collective structure and identifies the appropriate vocabulary necessary to communicate about it constantly.



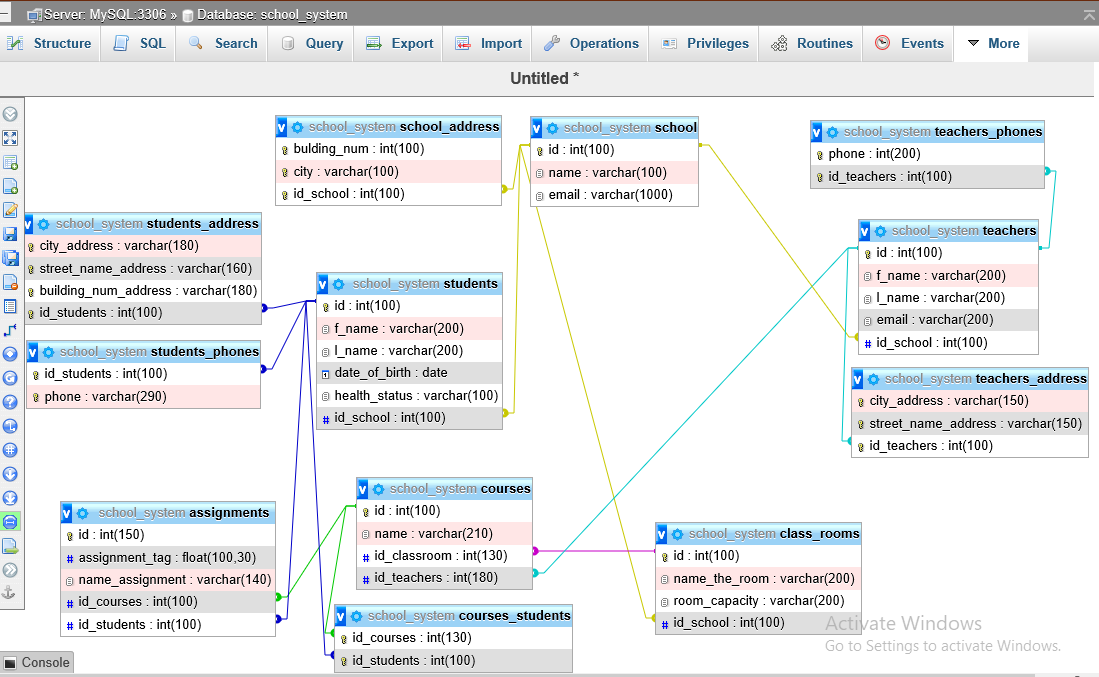
## 2.1.2 Logical Schema:

A logical data model describes data in as much detail as possible, regardless of how it is physical implemented in the database.



## 2.1.3Physical Schema:

It represents how the model will be built into the database. Shows all entity structures, attribute data type, attribute constraints, primary key, foreign key, and relationships between tables.



## 2.2 Mapping:

The process of matching fields from multiple data sets in a schema or central database. Data mapping is required for data migration, ingestion, processing, and data management. Ultimately, the goal of data mapping is to consolidate multiple data sets into one.

School (id, name, email)

School\_address (bulding\_num, city, id\_school)

Classrooms (id, name\_the\_room,room\_capacity,id\_school)

Teachers (id, f\_name, l\_name, email, id\_school)

Teachers\_address (city\_address, street\_name\_address, id\_teachers)

Teachers\_phones (phone, id\_teachers)

Students (id, f\_name, l\_name, date\_of\_birth, health\_status, id\_school)

Students\_address (city\_address, street\_name\_address, building\_num\_address, id\_students)

Students\_phones (id\_students, phone)

Courses (id, name, id\_classroom, id\_teachers)

Courses\_Students (id\_courses, id\_students)

Assignments (id, assignment\_tag, name\_ assignment, id\_courses, id\_students)

## 2.3 Normalization:

It is a way of organizing data in a database that helps you to avoid data duplication, insertion, updating and deletion of data. This involves creating tables and creating relationships between those tables according to rules designed to protect data and to make the database more resilient by eliminating redundancy and inconsistent dependency.

### First normal form:

1. Each cell in the table must contain one value (the table is Atomic).
2. Each record needs to be unique.
3. Values stored in a column should be of the same domain
4. All the columns in a table should have unique names.

#### To apply the **1st** NF:

1-If there is any multi-valued or complex attribute on table, we should Extract them to another table (new table) using one-to-many relationship (Making everything **Atomic**).

2-The primary key of the new relation is a combination of the primary key of the original relation plus an attribute from the newly created relation for unique identification.

#### 1)School:

The school has many bulding\_num and many cities. Because there are multi-valued on the table, we should Extract them to another table. Create a new table School\_address (id\_school, city, bulding\_num) and three column are primary key in school\_address.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Name | Bulding\_num | City | email |
| 1 | Imam Ali School | 10 | Amman | Imam\_Ali@gmail.com |
| 2 | Al-Hasan School | 20 | Jerash | Al-Hasan@gmail.com |
| 1 | Imam Ali School | 15 | Zarqa | Imam\_Ali@gmail.com |
| 2 | Al-Hasan School | 33 | Amman | Al-Hasan@gmail.com |

school: School\_address:

|  |  |  |
| --- | --- | --- |
| Id | Name | email |
| 1 | Imam Ali School | Imam\_Ali@gmail.com |
| 2 | Al-Hasan School | Al-Hasan@gmail.com |

|  |  |  |
| --- | --- | --- |
| Id\_school | Bulding\_num | city |
| 1 | 10 | Amman |
| 2 | 20 | Jerash |
| 1 | 15 | Zarqa |
| 2 | 33 | Amman |

#### 2)classrooms:

1NF conditions don't apply to The classroom that doesn't have multiple values ​​so it shouldn't be split into another table.

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Name\_the\_room | Room\_capacity | Id\_school |
| 1001 | Drawing room | 30 | 1 |
| 1002 | construction room | 20 | 2 |
| 1003 | Book room | 30 | 1 |
| 1004 | laboratory room | 20 | 2 |

#### 3)Assignments:

1NF conditions don't apply to The Assignments that don't have multiple values so they shouldn't be split into another table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Assignment\_tag** | **Name\_ assignment** | **Id\_courses** | **Id\_students** |
| **200** | **90** | **Half Math** | **12345** | **111** |
| **201** | **99** | **Half Physics** | **12344** | **111** |
| **202** | **100** | **Half Math** | **12345** | **113** |
| **203** | **88** | **Half Physics** | **12344** | **113** |

### Second normal form:

1-Relational (table) must be in 1st NF.

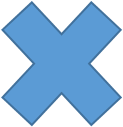
2-All non-key attributes are fully functional dependent on the primary key.  It should not have Partial Dependency.

#### To apply the 2nd NF:

1-To remove all partial dependencies by break the relation to two tables.

2-The primary key of the new relation is a combination of the primary key of the original relation plus an attribute from the newly created relation for unique identification.

#### 1)Teachers:

The table(Teachers) should remove all partial dependencies by splitting the relationship into three tables. The new relationship primary key is (Id\_teachers) a combination of the original relationship primary key plus an attribute from the newly created relationship of the unique identifier.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Id | Id\_school | F\_name | L\_name | Email | City | Street\_name | phone |
| 10 | 1 | Hussein | Al-hamada | [Hussein@gmail.com](mailto:Hussein@gmail.com) | Amman | Qais Street | 0788658586 |
| 10 | 1 | Hussein | Al-hamada | [Hussein@gmail.com](mailto:Hussein@gmail.com) | Jerash | Sabik Street | 0787856222 |
| 12 | 2 | Hammed | Al-jebor | [jebor@gmail.com](mailto:jebor@gmail.com) | Zarqa | Quran Street | 0772334543 |
| 12 | 2 | Hammed | Al-jebor | [jebor@gmail.com](mailto:jebor@gmail.com) | Amman | Hussein Street | 0784387767 |

Teachers:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Id\_school** | **F\_name** | **L\_name** | **email** |
| 10 | 1 | Hussein | Al-hamada | [Hussein@gmail.com](mailto:Hussein@gmail.com) |
| 12 | 2 | hammed | Al-jebor | [jebor@gmail.com](mailto:jebor@gmail.com) |



|  |  |
| --- | --- |
| **Id\_teachers** | **phone** |
| **10** | 0788658586 |
| **10** | 0787856222 |
| **12** | 0772334543 |
| **12** | 0784387767 |

Teachers\_address: Teachers\_phones:

|  |  |  |
| --- | --- | --- |
| **Id\_teachers** | **City** | **Street\_name** |
| 10 | Amman | Qais Street |
| 10 | Jerash | Sabik Street |
| 12 | Zarqa | Quran Street |
| 12 | Amman | Hussein Street |

#### **2)** Students:

The table(Students) should remove all partial dependencies by splitting the relationship into three tables. The new relationship primary key is (Id\_students) a combination of the original relationship primary key plus an attribute from the newly created relationship of the unique identifier.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Id\_school** | **F\_name** | **L\_name** | **Data\_of\_birth** | **City** | **Street\_name** | **Health\_status** | **Building\_num** | **phone** |
| **111** | **1** | **Hasan** | **Al-hwietat** | **2002-2-12** | **Amman** | **Karama Street** | **Good Health** | **10** | **0784554554** |
| **111** | **1** | **Hasan** | **Al-hwietat** | **2002-2-12** | **Zarqa** | **Al Hussein Street** | **Good Health** | **37** | **0784344511** |
| **113** | **2** | **Fadi** | **Al-skafi** | **2002-6-24** | **Jerash** | **Al Rahma Street** | **Mental illness** | **15** | **0798888662** |
| **113** | **2** | **Fadi** | **Al-skafi** | **2002-6-24** | **Amman** | **Al Fayha Street** | **Mental illness** | **22** | **0795568622** |

Students:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Id** | **F\_name** | **L\_name** | **Data\_of\_birth** | **Health\_status** | **Id\_school** |
| **111** | **Hasan** | **Al-hwietat** | **2002-2-12** | **Good Health** | **1** |
| **113** | **Fadi** | **Al-skafi** | **2002-6-24** | **Mental illness** | **2** |

Students\_address: Students\_phones:

|  |  |
| --- | --- |
| **Id\_students** | **phones** |
| **111** | **0784554554** |
| **111** | **0784344511** |
| **113** | **0798888662** |
| **113** | **0795568622** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Id\_students** | **City** | **Street\_name** | **Building\_num** |
| **111** | **Amman** | **Karama Street** | **10** |
| **111** | **Zarqa** | **Al Hussein Street** | **37** |
| **113** | **Jerash** | **Al Rahma Street** | **15** |
| **113** | **Amman** | **Al Fayha Street** | **22** |

#### **3)** Courses:

The table(courses) should remove all partial dependencies by splitting the relationship into two tables. The new relationship primary key is (Id\_courses) a combination of the original relationship primary key plus an attribute from the newly created relationship of the unique identifier.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Name** | **Id\_classroom** | **Id\_teachers** | **Id\_students** |
| **12345** | **Math** | **1001** | **10** | **111** |
| **12344** | **Physics** | **1002** | **12** | **111** |
| **12345** | **Math** | **1001** | **10** | **113** |
| **12344** | **Physics** | **1002** | **12** | **113** |

Courses: courses\_students:

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Name** | **Id\_classroom** | **Id\_teachers** |
| **12345** | **Math** | **1001** | **10** |
| **12344** | **physics** | **1002** | **12** |

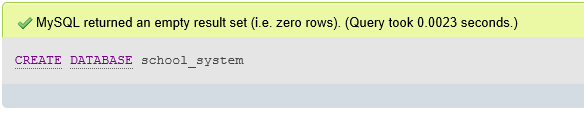
|  |  |
| --- | --- |
| **Id\_** **Courses** | **Id\_students** |
| **12345** | **111** |
| **12344** | **111** |
| **12345** | **113** |
| **12344** | **113** |

# 3 Development:

## 3.1 create database, tables, insert:

### 3.1.1 create database:

CREATE DATABASE school\_system;



### 3.1.2 tables:

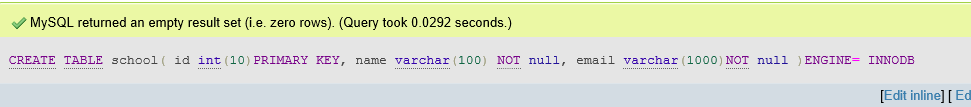
CREATE TABLE school(

id int (10) PRIMARY KEY,

name varchar(100) NOT null,

email varchar(1000) NOT null

) ENGINE= INNODB;



CREATE TABLE school\_address(

bulding\_num int(100) not null,

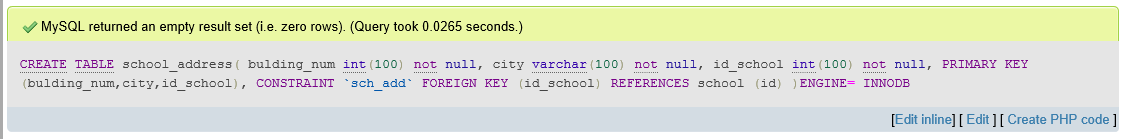
city varchar(100) not null,

id\_school int(100) not null,

PRIMARY KEY(bulding\_num,city,id\_school),

CONSTRAINT `sch\_add` FOREIGN KEY(id\_school) REFERENCES school(id)

)ENGINE= INNODB;



CREATE TABLE class\_rooms(

id int (100) PRIMARY KEY,

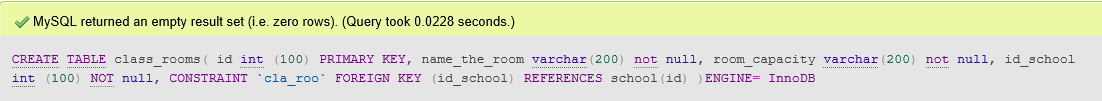
name\_the\_room varchar(200) not null,

room\_capacity varchar(200) not null,

id\_school int (100) NOT null,

CONSTRAINT `cla\_roo` FOREIGN KEY (id\_school) REFERENCES school(id)

)ENGINE= InnoDB;



CREATE TABLE teachers(

id int(100) PRIMARY KEY,

f\_name varchar(200) not null,

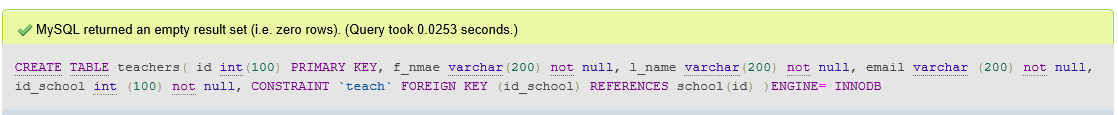
l\_name varchar(200) not null,

email varchar (200) not null,

id\_school int (100) not null,

CONSTRAINT `teach` FOREIGN KEY (id\_school) REFERENCES school(id)

)ENGINE= INNODB;



*CREATE TABLE teachers\_address(*

*city\_address varchar (150) not null,*

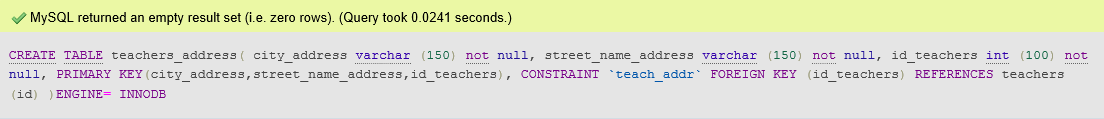
*street\_name\_address varchar (150) not null,*

*id\_teachers int (100) not null,*

*PRIMARY KEY(city\_address,street\_name\_address,id\_teachers),*

*CONSTRAINT `teach\_addr` FOREIGN KEY (id\_teachers) REFERENCES teachers (id)*

*)ENGINE= INNODB;*



CREATE TABLE teachers\_phones(

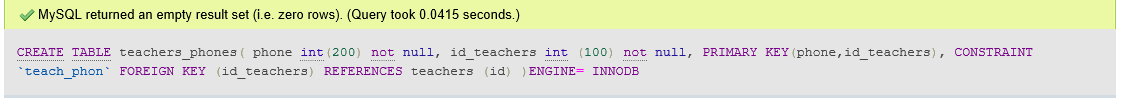
phone int(200) not null,

id\_teachers int (100) not null,

PRIMARY KEY(phone,id\_teachers),

CONSTRAINT `teach\_phon` FOREIGN KEY (id\_teachers) REFERENCES teachers (id)

)ENGINE= INNODB;



CREATE TABLE students(

id int(100) PRIMARY KEY,

f\_name varchar(200) not null,

l\_name varchar(200) not null,

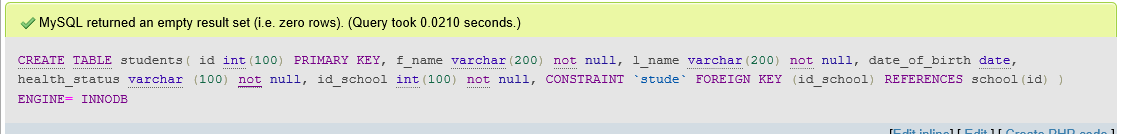
date\_of\_birth date,

health\_status varchar (100) not null,

id\_school int(100) not null,

CONSTRAINT `stude` FOREIGN KEY (id\_school) REFERENCES school(id)

)ENGINE= INNODB;



CREATE TABLE students\_address(

city\_address varchar (180) DEFAULT 'Amman',

street\_name\_address varchar (160) not null,

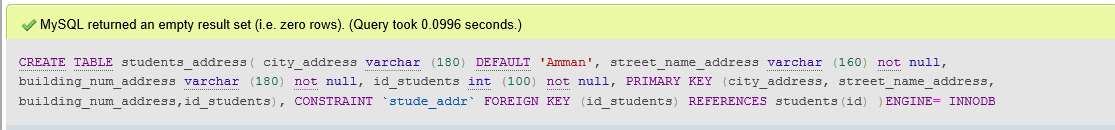
building\_num\_address varchar (180) not null,

id\_students int (100) not null,

PRIMARY KEY (city\_address, street\_name\_address, building\_num\_address,id\_students),

CONSTRAINT `stude\_addr` FOREIGN KEY (id\_students) REFERENCES students(id)

)ENGINE= INNODB;



CREATE TABLE students\_phones(

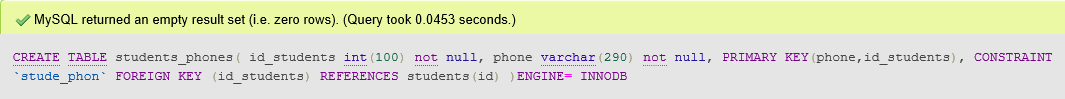
id\_students int(100) not null,

phone varchar(290) not null,

PRIMARY KEY(phone,id\_students),

CONSTRAINT `stude\_phon` FOREIGN KEY (id\_students) REFERENCES students(id)

)ENGINE= INNODB;



CREATE TABLE courses(

id int(100) PRIMARY KEY,

name varchar (210) not null,

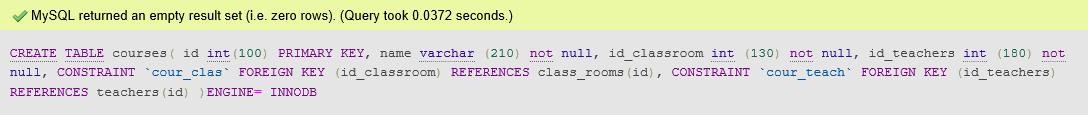
id\_classroom int (130) not null,

id\_teachers int (180) not null,

CONSTRAINT `cour\_clas` FOREIGN KEY (id\_classroom) REFERENCES class\_rooms(id),

CONSTRAINT `cour\_teach` FOREIGN KEY (id\_teachers) REFERENCES teachers(id)

)ENGINE= INNODB;



CREATE TABLE courses\_students(

id\_courses int (130) not null,

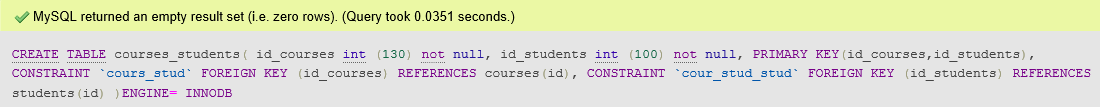
id\_students int (100) not null,

PRIMARY KEY(id\_courses,id\_students),

CONSTRAINT `cours\_stud` FOREIGN KEY (id\_courses) REFERENCES courses(id),

CONSTRAINT `cour\_stud\_stud` FOREIGN KEY (id\_students) REFERENCES students(id)

)ENGINE= INNODB;



CREATE TABLE assignments(

id int (150) PRIMARY KEY,

assignment\_tag float(100,30) not null,

name\_assignment varchar (140)not null,

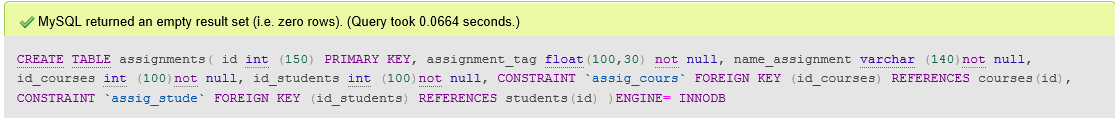
id\_courses int (100)not null,

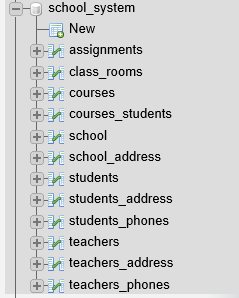
id\_students int (100)not null,

CONSTRAINT `assig\_cours` FOREIGN KEY (id\_courses) REFERENCES courses(id),

CONSTRAINT `assig\_stude` FOREIGN KEY (id\_students) REFERENCES students(id)

)ENGINE= INNODB;





### 3.1.3 insert:

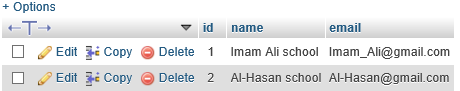
INSERT INTO school

VALUES(1,'Imam Ali school','Imam\_Ali@gmail.com');

INSERT INTO school

VALUES(2,'Al-Hasan school','Al-Hasan@gmail.com');

INSERT INTO school



INSERT INTO school\_address

VALUES(10,'Amman',1);

INSERT INTO school\_address

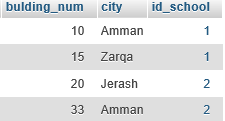
VALUES(20,'Jerash',2);

INSERT INTO school\_address

VALUES(15,'Zarqa',1);

INSERT INTO school\_address

VALUES(33,'Amman',2);



INSERT INTO class\_rooms

VALUES(1001,'Drawing room',30,1);

INSERT INTO class\_rooms

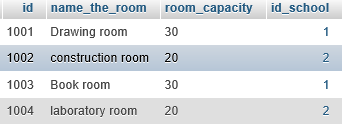
VALUES(1002,'construction room',20,2);

INSERT INTO class\_rooms

VALUES(1003,'Book room',30,1);

INSERT INTO class\_rooms

VALUES(1004,'laboratory room',20,2);

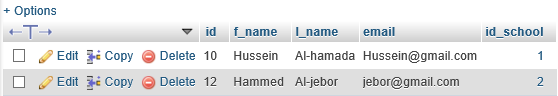


INSERT INTO teachers

VALUES(10,'Hussein','Al-hamada','Hussein@gmail.com',1);

INSERT INTO teachers

VALUES(12, 'Hammed','Al-jebor','jebor@gmail.com',2);



INSERT INTO teachers\_address

VALUES('Amman','Qais Street',10);

INSERT INTO teachers\_address

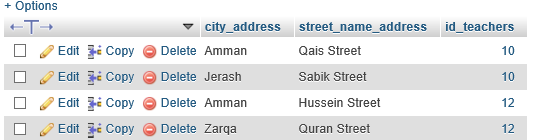
VALUES('Jerash','Sabik Street',10);

INSERT INTO teachers\_address

VALUES('Zarqa','Quran Street',12);

INSERT INTO teachers\_address

VALUES('Amman','Hussein Street',12);



INSERT INTO teachers\_phones

VALUES(0788658586,10);

INSERT INTO teachers\_phones

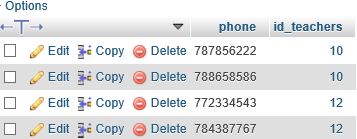
VALUES(0787856222,10);

INSERT INTO teachers\_phones

VALUES(0772334543,12);

INSERT INTO teachers\_phones

VALUES(0784387767,12);



INSERT INTO students

VALUES(111,'Hasan','Al-hwietat', '2002-02-12','Good Health',1);

INSERT INTO students

VALUES (113, 'Fadi', 'Al-skafi', '2002-6-24','Mental illness',2);



INSERT INTO students\_address

VALUES('Amman','karama Street',10,111);

INSERT INTO students\_address

VALUES('Zarqa','Al Hussein Street',37,111);

INSERT INTO students\_address

VALUES('Jerash','Al Rahma Street',15,113);

INSERT INTO students\_address

VALUES('Amman','Al-Fayha Street',22,113);



INSERT INTO students\_phones

VALUES(111,0784554554);

INSERT INTO students\_phones

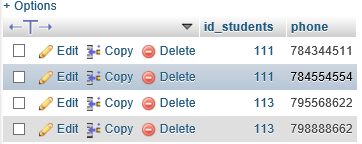
VALUES(111,0784344511);

INSERT INTO students\_phones

VALUES (112,0798888662);

INSERT INTO students\_phones

VALUES (112,0795568622);

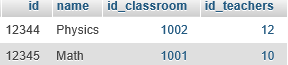


INSERT INTO courses

VALUES(12345,'Math',1001,10);

INSERT INTO courses

VALUES(12344,'Physics',1002,12);



INSERT INTO courses\_students

VALUES(12345,111);

INSERT INTO courses\_students

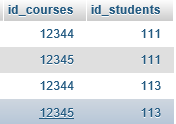
VALUES(12344,111);

INSERT INTO courses\_students

VALUES (12345,113);

INSERT INTO courses\_students

VALUES (12344,113);



INSERT INTO assignments

VALUES(200,90,'Half Math',12345,111);

INSERT INTO assignments

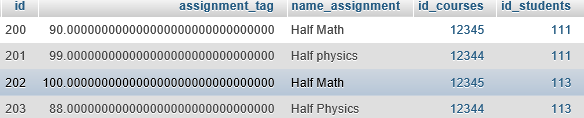
VALUES(201,90,'Half physics',12344,111);

INSERT INTO assignments

VALUES (202,100,'Half Math',12345,113);

INSERT INTO assignments

VALUES (203,88,'Half Physics',12344,113);

****

## 3.2 create users and give privileges + view and procedures

### 3.2.1 view:

CREATE VIEW school\_info AS

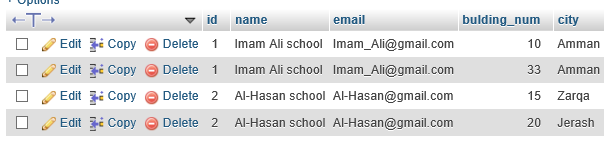
SELECT school.id,school.name,school.email, school\_address.bulding\_num,school\_address.city

FROM school INNER JOIN school\_address

ON school.id=school\_address.id\_school

ORDER BY school.id ASC;

SELECT \* FROM school\_info;



CREATE VIEW teachers\_info1 AS

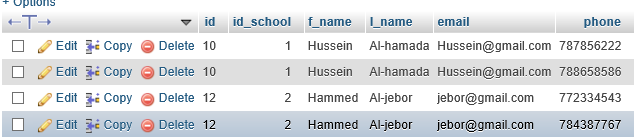
SELECT teachers.id,teachers.id\_school,teachers.f\_name, teachers.l\_name,teachers.email,teachers\_phones.phone

FROM teachers INNER JOIN teachers\_phones

ON teachers.id=teachers\_phones.id\_teachers

ORDER BY teachers.id ASC;

SELECT \* FROM teachers\_info1;



CREATE VIEW teachers\_info2 AS

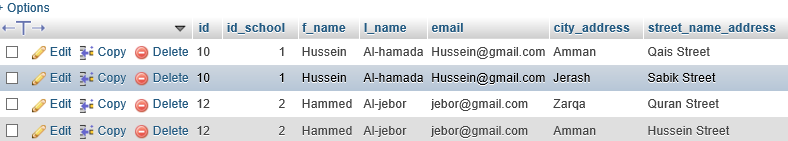
SELECT teachers.id,teachers.id\_school,teachers.f\_name, teachers.l\_name,teachers.email,teachers\_address.city\_address,teachers\_address.street\_name\_address

FROM teachers INNER JOIN teachers\_address

ON teachers.id=teachers\_address.id\_teachers

ORDER BY teachers.id ASC;

SELECT \* FROM teachers\_info2;



CREATE VIEW stud\_assig AS

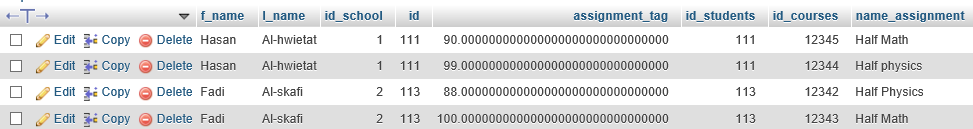
SELECT students.f\_name, students.l\_name, students.id\_school, students.id, assignments.assignment\_tag, assignments.id\_students, assignments.id\_courses, assignments.name\_assignment

FROM students INNER JOIN assignments

ON students.id=assignments.id\_students

ORDER BY students.id ASC;

SELECT \* FROM stud\_assig;



CREATE VIEW stud\_cou AS

SELECT courses\_students.id\_courses,students.id,students.f\_name,students.l\_name,students.date\_of\_birth,students.health\_status,students.id\_school

FROM students INNER JOIN courses\_students

ON students.id= courses\_students.id\_students

ORDER BY students.id ASC;

SELECT \* FROM stud\_cou;



CREATE VIEW students\_info1 AS

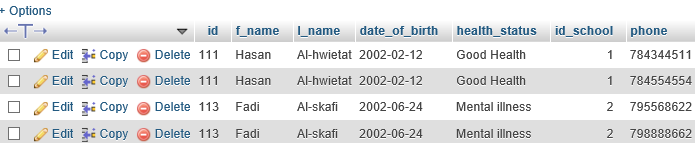
SELECT students.id,students.f\_name,students.l\_name,students.date\_of\_birth,students.health\_status,students.id\_school, students\_phones.phone

FROM students INNER JOIN students\_phones

ON students.id=students\_phones.id\_students

ORDER BY students.id ASC;

SELECT \* FROM students\_info1;



CREATE VIEW students\_info2 AS

SELECT students.id,students.f\_name,students.l\_name,students.date\_of\_birth,students.health\_status,students.id\_school, students\_address.city\_address,students\_address.street\_name\_address,students\_address.building\_num\_address

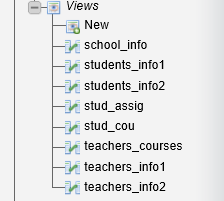
FROM students INNER JOIN students\_address

ON students.id=students\_address.id\_students

ORDER BY students.id ASC;

SELECT \* FROM students\_info2;





### 3.2.1procedures:

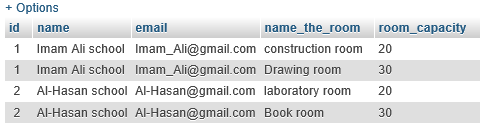
CREATE PROCEDURE fadi()

SELECT school.id,school.name,school.email,class\_rooms.name\_the\_room,class\_rooms.room\_capacity FROM school INNER JOIN class\_rooms

ON school.id=class\_rooms.id\_school

ORDER BY school.id ASC;

CALL fadi();



### 3.2.2 create users:

CREATE USER 'teachers'@'localhost' IDENTIFIED BY '4453';

meaningful data can be extracted from user teachers:

1- He can update his data, add his data to the base, show and delete all data related to him, and know the name of the school in which he teaches.

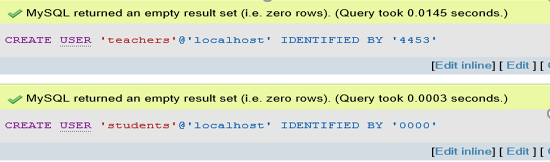
2-He can see the student's data, take exams and download the grades on the site, and he can know the number of students in this subject he gives and know the best student in this subject, the worst student and the average students.

CREATE USER 'students'@'localhost' IDENTIFIED BY '0000';

meaningful data can be extracted from user students:

1- A student can fill in the data related to him, update and remove it.

2- A student can choose the teacher who wants to give him the material, know the marks of the subjects he is studying, and know the name of the school in which he is studying.



### 3.2.2 privileges:

GRANT SELECT ON school\_system.courses TO 'students'@'localhost';

GRANT SELECT,INSERT,UPDATE,DELETE ON school\_system.courses\_students TO 'students'@'localhost';

GRANT SELECT ON school\_system.assignments TO 'students'@'localhost';

GRANT INSERT,UPDATE,DELETE ON school\_system.students TO 'students'@'localhost';

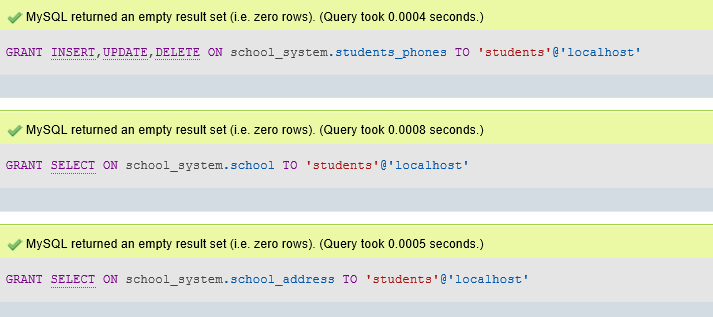
GRANT INSERT,UPDATE,DELETE ON school\_system.students\_address TO 'students'@'localhost';

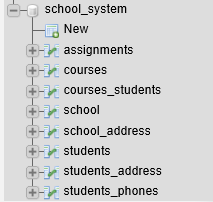
GRANT INSERT,UPDATE,DELETE ON school\_system.students\_phones TO 'students'@'localhost';

GRANT SELECT ON school\_system.school TO 'students'@'localhost';

GRANT SELECT ON school\_system.school\_address TO 'students'@'localhost';







GRANT SELECT ON school\_system.students TO 'teachers'@'localhost';

GRANT INSERT,SELECT,UPDATE,DELETE ON school\_system.teachers TO 'teachers'@'localhost';

GRANT INSERT,SELECT,UPDATE,DELETE ON school\_system.teachers\_phones TO 'teachers'@'localhost';

GRANT INSERT,SELECT,UPDATE,DELETE ON school\_system.teachers\_address TO 'teachers'@'localhost';

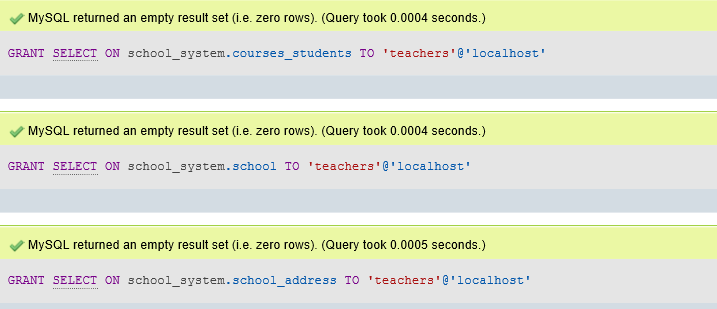
GRANT INSERT,SELECT,UPDATE,DELETE ON school\_system.assignments TO 'teachers'@'localhost';

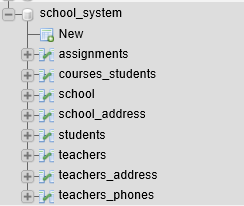
GRANT SELECT ON school\_system.courses\_students TO 'teachers'@'localhost';

GRANT SELECT ON school\_system.school TO 'teachers'@'localhost';

GRANT SELECT ON school\_system.school\_address TO 'teachers'@'localhost';







# 4 user and technical documentation:

## 4.1 User documentation:

User documentation, Written or other visual information consists of instructions and information to users who will interact with the system. It includes user manuals, Help screens, and tutorials.

Scenario:

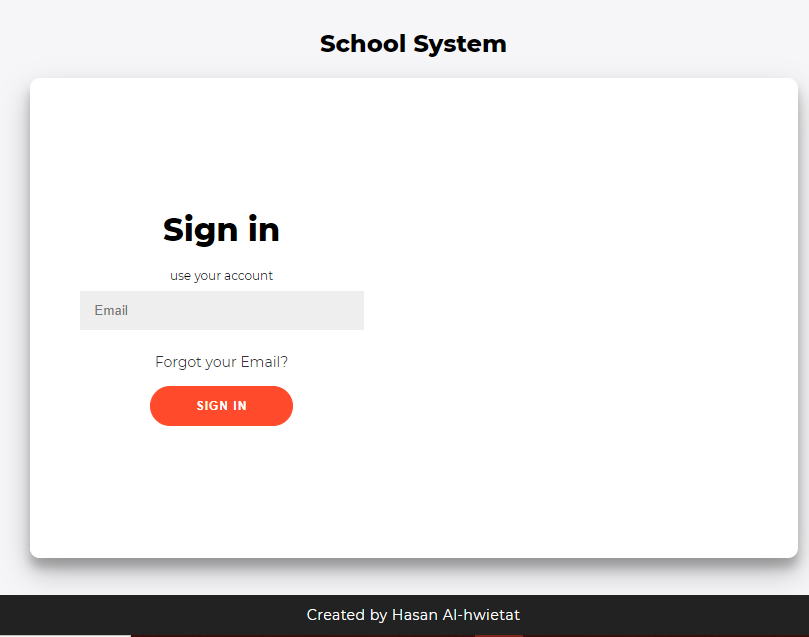
school database

The database keeps the information about the school, like (Id, name, email, building num\_address, city\_address). many teachers work in the school, the teacher teaches many courses. the school has many classrooms. Each classroom has more than one course and the course is given in one room. A student can take many courses. One course is taken by more than one student. the course has many assignments. Each student has many assignments. The database keeps the information about teachers, such as (id, f\_name, l\_name, phones, city\_address, street name\_address, email). The specified student information is kept in the database (ID, name, l\_name, city\_address, street\_address, building address\_number, health status, date of birth, phones). Also, classroom information is kept in the database (ID, room name, room capacity). The database keeps the information about courses, like (ID, name). The database keeps the information about assignments, such as (id, name the assignment, assignment tag)

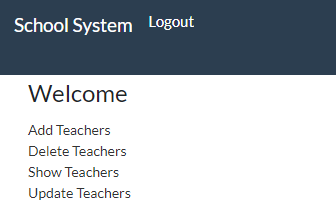
1.2 User groups and Business needs:

1. Teachers: storing their data They can browse information about their students, ask them for homework, and send their grades to Administrative staff or download over the school website, and know the name of the school in which he teaches.
2. Students: can choose the teacher they want and can the courses schedule either over the phone or using the school's website, and know the name of the school in which he studies. also should be able to access the database using a web browser to request grades and submit assignments.
3. root: He can do all the things he wants from seeing the number of students in the school, the number of teachers, the subjects that are given in this school and the first ones who got the highest marks.

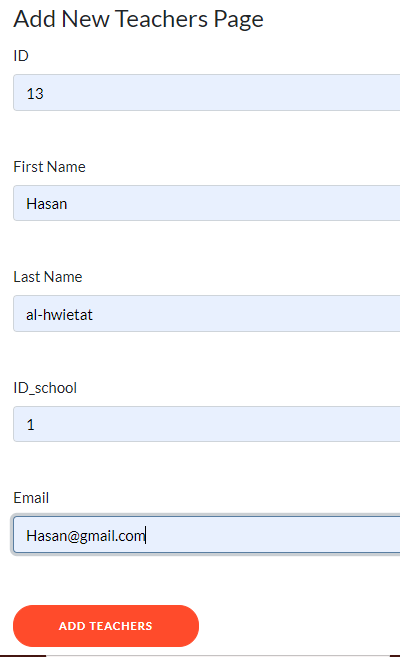
Home Page You must enter your Email to enter the next page.



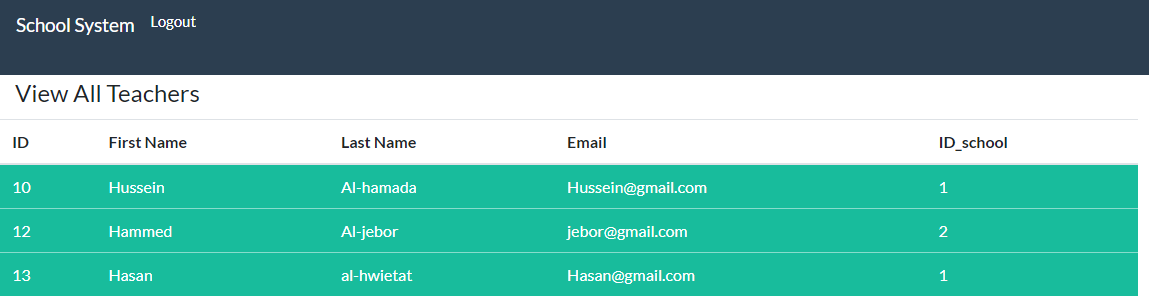
The page that can do anything you want to do from adding a teacher, removing a teacher, showing all teachers, updating the data to move to the next page, or exiting and returning to the home page.



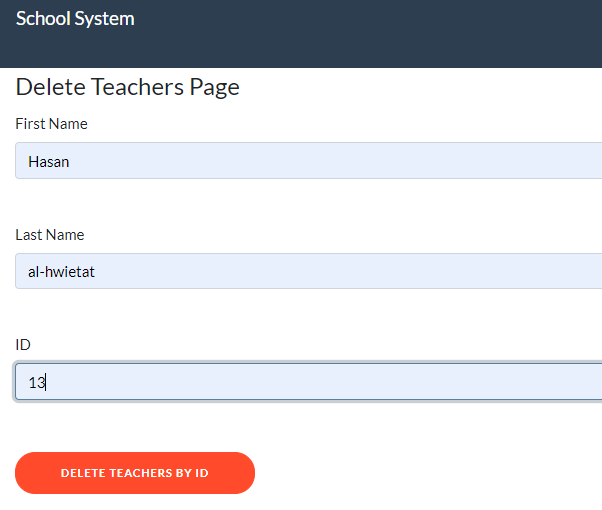
On this page you can add a feature and then return to the previous page.



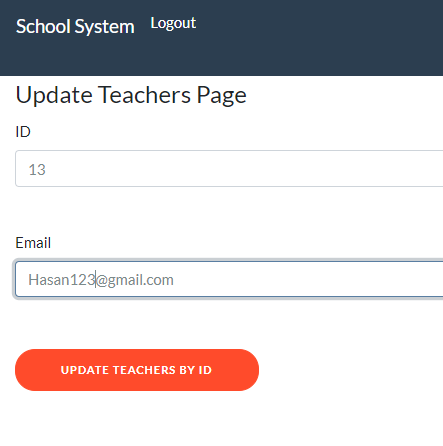
On this page you can see all the teachers in this school and then click logout to return to the main page.



On this page, you can remove the teacher from this school and then click on Delete to return to the previous page.



On this page, you can update the teacher's email from this school, then click on Update and return to the previous page.



Flowcharts:

is a diagram that depicts a computer process, system, or algorithm. They are used extensively in multiple fields to document, study, plan, improve and communicate often complex processes in clear, easy-to-understand diagrams.

T

F

Database to update teachers

Database to show teachers

Database to delete teachers

Database to add teachers

Click to Update Teachers

logout

T

Click to Show Teachers

Click to Delete Teachers

T

Click to Add Teachers

T

T

T

F

Check Email

Enter the Email

If(finished)

Diagrams: A diagram is a line drawing comprised of symbols that represent key objects and actions that occur within a system. The type of system often determines the set or library of symbols used within a particular diagram.



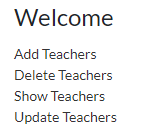
Open the website.



Sign in to the site using your email.

Go to the home page and click on one of the options you want.

Logout





Go to the page you choose.

## 4.2 Technical documentation:

Technical documentation refers to documents containing information and data related to a product, containing details of a technical product or service that is under development or already in use. Simply put, technical documentation is all about presenting the product in a way that is easy to use, read and understand.

Scenario:

school database

The database keeps the information about the school, like (Id, name, email, building num\_address, city\_address). many teachers work in the school, the teacher teaches many courses. the school has many classrooms. Each classroom has more than one course and the course is given in one room. A student can take many courses. One course is taken by more than one student. the course has many assignments. Each student has many assignments. The database keeps the information about teachers, such as (id, f\_name, l\_name, phones, city\_address, street name\_address, email). The specified student information is kept in the database (ID, name, l\_name, city\_address, street\_address, building address\_number, health status, date of birth, phones). Also, classroom information is kept in the database (ID, room name, room capacity). The database keeps the information about courses, like (ID, name). The database keeps the information about assignments, such as (id, name the assignment, assignment tag).

User groups and Business needs:

1-Teachers: storing their data They can browse information about their students, ask them for homework, and send their grades to Administrative staff or download over the school website, and know the name of the school in which he teaches.

2-Students: can choose the teacher they want and can the courses schedule either over the phone or using the school's website, and know the name of the school in which he studies. also should be able to access the database using a web browser to request grades and submit assignments.

3-root: He can do all the things he wants from seeing the number of students in the school, the number of teachers, the subjects that are given in this school and the first ones who got the highest marks.

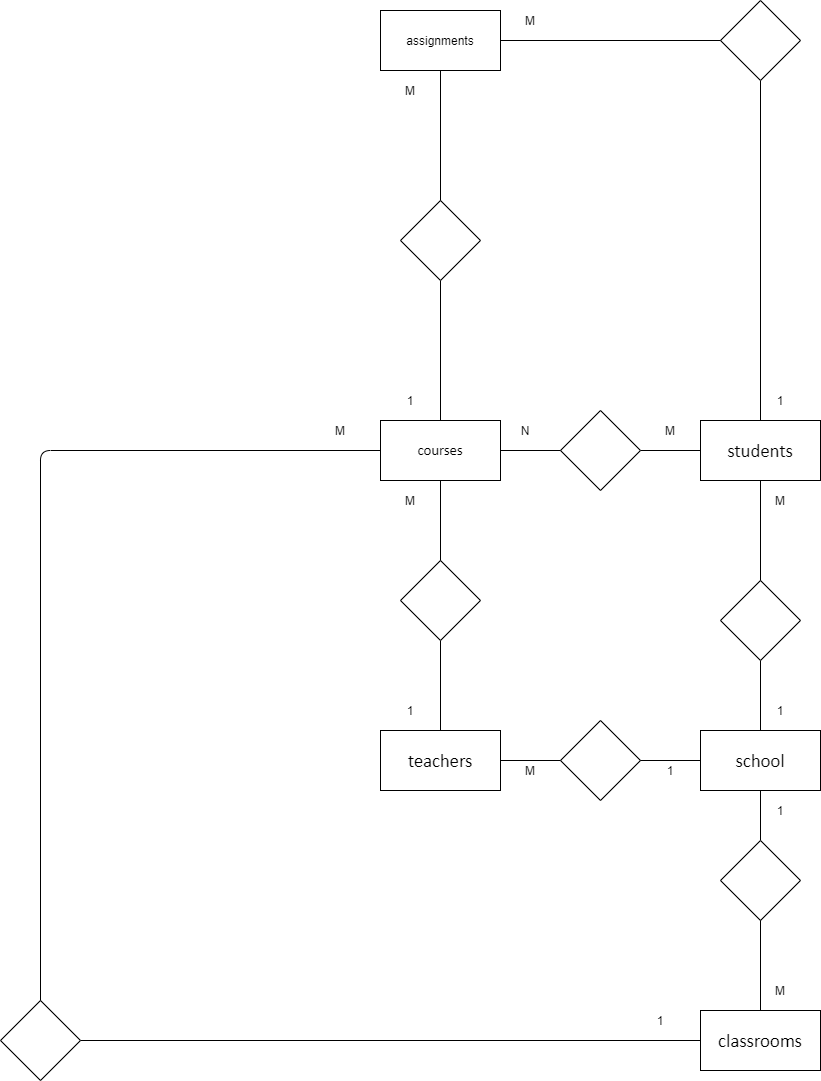
System requirements:

1-Many computers or laptops are required for teachers and administration to store data.

2-The building must have an internet connection to save data related to the school on its official website.

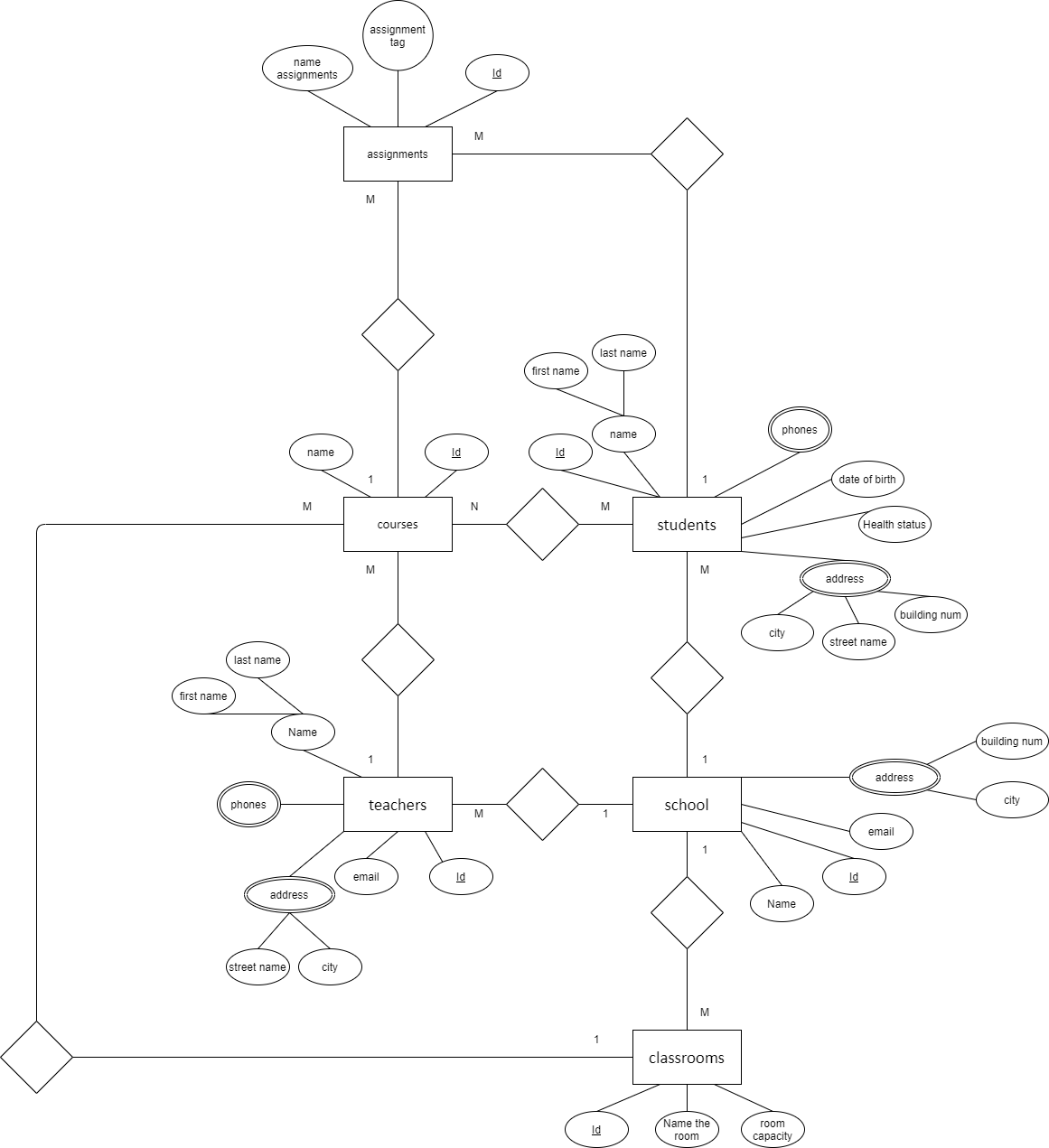
Conceptual schema:

A model that develops the meaning of the basic concepts of a problem area, defines its collective structure and identifies the appropriate vocabulary necessary to communicate about it constantly.



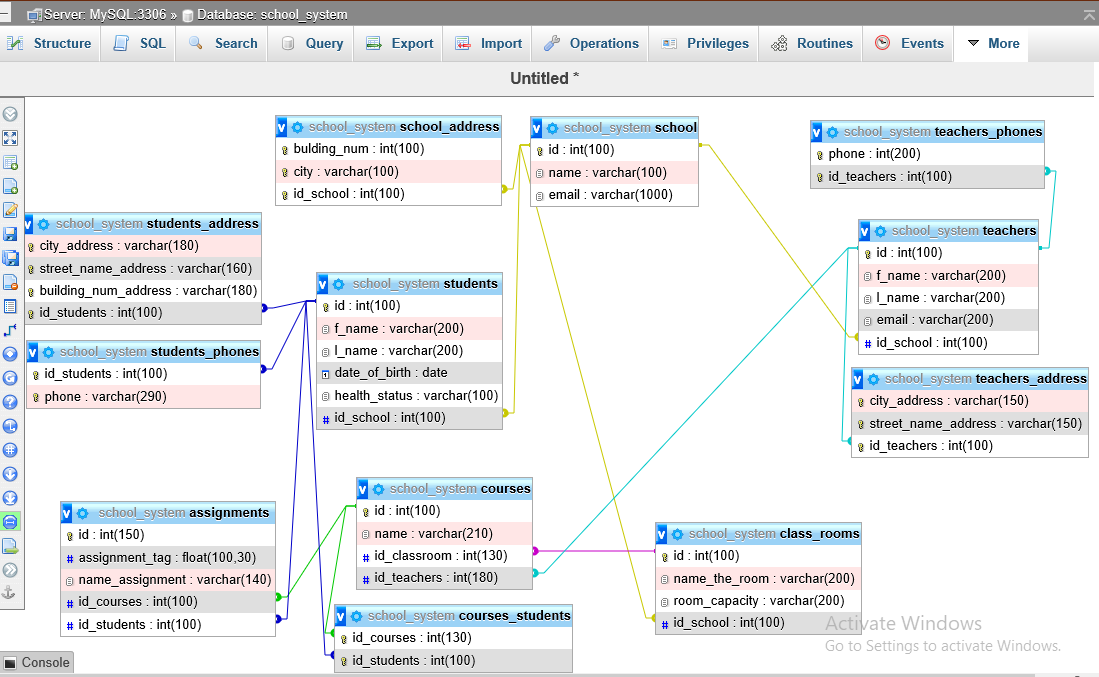
Logical Schema:

A logical data model describes data in as much detail as possible, regardless of how it is physical implemented in the database.



Physical Schema:

It represents how the model will be built into the database. Shows all entity structures, attribute data type, attribute constraints, primary key, foreign key, and relationships between tables.



Mapping:

The process of matching fields from multiple data sets in a schema or central database. Data mapping is required for data migration, ingestion, processing, and data management. Ultimately, the goal of data mapping is to consolidate multiple data sets into one.

School (id, name, email)

School\_address (bulding\_num, city, id\_school)

Classrooms (id, name\_the\_room,room\_capacity,id\_school)

Teachers (id, f\_name, l\_name, email, id\_school)

Teachers\_address (city\_address, street\_name\_address, id\_teachers)

Teachers\_phones (phone, id\_teachers)

Students (id, f\_name, l\_name, date\_of\_birth, health\_status, id\_school)

Students\_address (city\_address, street\_name\_address, building\_num\_address, id\_students)

Students\_phones (id\_students, phone)

Courses (id, name, id\_classroom, id\_teachers)

Courses\_Students (id\_courses, id\_students)

Assignments (id, assignment\_tag, name\_ assignment, id\_courses, id\_students)

Normalization:

It is a way of organizing data in a database that helps you to avoid data duplication, insertion, updating and deletion of data. This involves creating tables and creating relationships between those tables according to rules designed to protect data and to make the database more resilient by eliminating redundancy and inconsistent dependency.

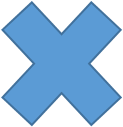
First normal form:

1. Each cell in the table must contain one value (the table is Atomic).
2. Each record needs to be unique.
3. Values stored in a column should be of the same domain
4. All the columns in a table should have unique names.

To apply the **1st** NF:

1-If there is any multi-valued or complex attribute on table, we should Extract them to another table (new table) using one-to-many relationship (Making everything **Atomic**).

2-The primary key of the new relation is a combination of the primary key of the original relation plus an attribute from the newly created relation for unique identification.

1)School:

The school has many bulding\_num and many cities. Because there are multi-valued on the table, we should Extract them to another table. Create a new table School\_address (id\_school, city, bulding\_num) and three column are primary key in school\_address.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Name | Bulding\_num | City | email |
| 1 | Imam Ali School | 10 | Amman | Imam\_Ali@gmail.com |
| 2 | Al-Hasan School | 20 | Jerash | Al-Hasan@gmail.com |
| 1 | Imam Ali School | 15 | Zarqa | Imam\_Ali@gmail.com |
| 2 | Al-Hasan School | 33 | Amman | Al-Hasan@gmail.com |

school: School\_address:

|  |  |  |
| --- | --- | --- |
| Id | Name | email |
| 1 | Imam Ali School | Imam\_Ali@gmail.com |
| 2 | Al-Hasan School | Al-Hasan@gmail.com |

|  |  |  |
| --- | --- | --- |
| Id\_school | Bulding\_num | city |
| 1 | 10 | Amman |
| 2 | 20 | Jerash |
| 1 | 15 | Zarqa |
| 2 | 33 | Amman |

2)classrooms:

1NF conditions don't apply to The classroom that doesn't have multiple values ​​so it shouldn't be split into another table.

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Name\_the\_room | Room\_capacity | Id\_school |
| 1001 | Drawing room | 30 | 1 |
| 1002 | construction room | 20 | 2 |
| 1003 | Book room | 30 | 1 |
| 1004 | laboratory room | 20 | 2 |

3)Assignments:

1NF conditions don't apply to The Assignments that don't have multiple values so they shouldn't be split into another table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Assignment\_tag** | **Name\_ assignment** | **Id\_courses** | **Id\_students** |
| **200** | **90** | **Half Math** | **12345** | **111** |
| **201** | **99** | **Half Physics** | **12344** | **111** |
| **202** | **100** | **Half Math** | **12345** | **113** |
| **203** | **88** | **Half Physics** | **12344** | **113** |

Second normal form:

1-Relational (table) must be in 1st NF.

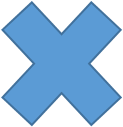
2-All non-key attributes are fully functional dependent on the primary key.  It should not have Partial Dependency.

To apply the 2nd NF:

1-To remove all partial dependencies by break the relation to two tables.

2-The primary key of the new relation is a combination of the primary key of the original relation plus an attribute from the newly created relation for unique identification.

1)Teachers:

The table(Teachers) should remove all partial dependencies by splitting the relationship into three tables. The new relationship primary key is (Id\_teachers) a combination of the original relationship primary key plus an attribute from the newly created relationship of the unique identifier.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Id | Id\_school | F\_name | L\_name | Email | City | Street\_name | phone |
| 10 | 1 | Hussein | Al-hamada | [Hussein@gmail.com](mailto:Hussein@gmail.com) | Amman | Qais Street | 0788658586 |
| 10 | 1 | Hussein | Al-hamada | [Hussein@gmail.com](mailto:Hussein@gmail.com) | Jerash | Sabik Street | 0787856222 |
| 12 | 2 | Hammed | Al-jebor | [jebor@gmail.com](mailto:jebor@gmail.com) | Zarqa | Quran Street | 0772334543 |
| 12 | 2 | Hammed | Al-jebor | [jebor@gmail.com](mailto:jebor@gmail.com) | Amman | Hussein Street | 0784387767 |

Teachers:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Id\_school** | **F\_name** | **L\_name** | **email** |
| 10 | 1 | Hussein | Al-hamada | [Hussein@gmail.com](mailto:Hussein@gmail.com) |
| 12 | 2 | hammed | Al-jebor | [jebor@gmail.com](mailto:jebor@gmail.com) |

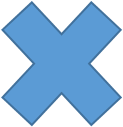


|  |  |
| --- | --- |
| **Id\_teachers** | **phone** |
| **10** | 0788658586 |
| **10** | 0787856222 |
| **12** | 0772334543 |
| **12** | 0784387767 |

Teachers\_address: Teachers\_phones:

|  |  |  |
| --- | --- | --- |
| **Id\_teachers** | **City** | **Street\_name** |
| 10 | Amman | Qais Street |
| 10 | Jerash | Sabik Street |
| 12 | Zarqa | Quran Street |
| 12 | Amman | Hussein Street |

**2)** Students:

The table(Students) should remove all partial dependencies by splitting the relationship into three tables. The new relationship primary key is (Id\_students) a combination of the original relationship primary key plus an attribute from the newly created relationship of the unique identifier.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Id\_school** | **F\_name** | **L\_name** | **Data\_of\_birth** | **City** | **Street\_name** | **Health\_status** | **Building\_num** | **phone** |
| **111** | **1** | **Hasan** | **Al-hwietat** | **2002-2-12** | **Amman** | **Karama Street** | **Good Health** | **10** | **0784554554** |
| **111** | **1** | **Hasan** | **Al-hwietat** | **2002-2-12** | **Zarqa** | **Al Hussein Street** | **Good Health** | **37** | **0784344511** |
| **113** | **2** | **Fadi** | **Al-skafi** | **2002-6-24** | **Jerash** | **Al Rahma Street** | **Mental illness** | **15** | **0798888662** |
| **113** | **2** | **Fadi** | **Al-skafi** | **2002-6-24** | **Amman** | **Al Fayha Street** | **Mental illness** | **22** | **0795568622** |

Students:

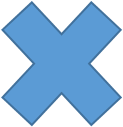
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Id** | **F\_name** | **L\_name** | **Data\_of\_birth** | **Health\_status** | **Id\_school** |
| **111** | **Hasan** | **Al-hwietat** | **2002-2-12** | **Good Health** | **1** |
| **113** | **Fadi** | **Al-skafi** | **2002-6-24** | **Mental illness** | **2** |



Students\_address: Students\_phones:

|  |  |
| --- | --- |
| **Id\_students** | **phones** |
| **111** | **0784554554** |
| **111** | **0784344511** |
| **113** | **0798888662** |
| **113** | **0795568622** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Id\_students** | **City** | **Street\_name** | **Building\_num** |
| **111** | **Amman** | **Karama Street** | **10** |
| **111** | **Zarqa** | **Al Hussein Street** | **37** |
| **113** | **Jerash** | **Al Rahma Street** | **15** |
| **113** | **Amman** | **Al Fayha Street** | **22** |

**3)** Courses:

The table(courses) should remove all partial dependencies by splitting the relationship into two tables. The new relationship primary key is (Id\_courses) a combination of the original relationship primary key plus an attribute from the newly created relationship of the unique identifier.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Name** | **Id\_classroom** | **Id\_teachers** | **Id\_students** |
| **12345** | **Math** | **1001** | **10** | **111** |
| **12344** | **Physics** | **1002** | **12** | **111** |
| **12345** | **Math** | **1001** | **10** | **113** |
| **12344** | **Physics** | **1002** | **12** | **113** |

Courses: courses\_students:

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Name** | **Id\_classroom** | **Id\_teachers** |
| **12345** | **Math** | **1001** | **10** |
| **12344** | **physics** | **1002** | **12** |

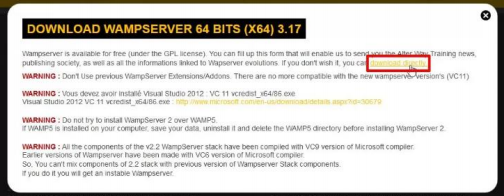
|  |  |
| --- | --- |
| **Id\_** **Courses** | **Id\_students** |
| **12345** | **111** |
| **12344** | **111** |
| **12345** | **113** |
| **12344** | **113** |

How to install WAMP - step by step:

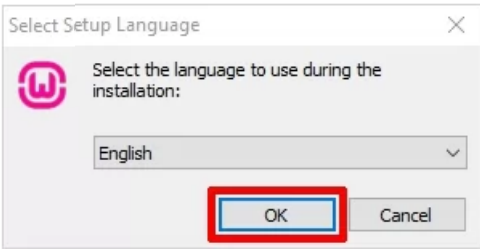
Step 1: Download WAMP Server Go to the official website http://www.wampserver.com/en/ and download the WampServer setup. There are two versions of WampServer available for example. 64-bits and 32-bits Choose according to the specifications of your computer.



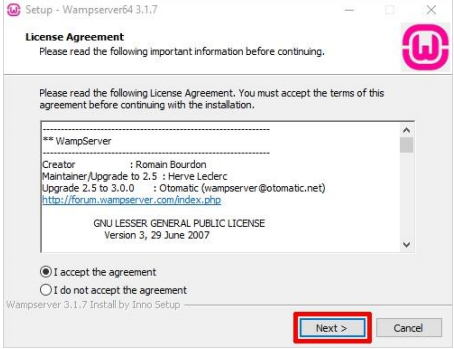
Click on the download option, a popup will appear showing some warnings. Just don't worry about these warnings, instead, just click the "Download Directly" link as shown below and proceed with the download process.



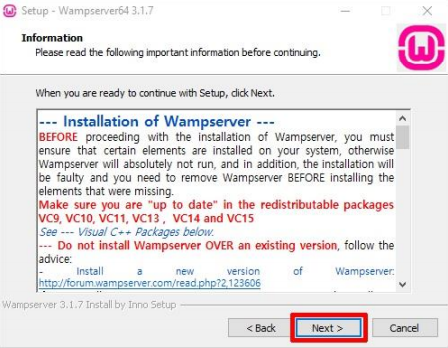
Step 2: will Start the WAMP Server installation process Shortly after you click on the downloaded file, you will be to choose the language and then click on the “Ok” button. Because I chose English here. see below



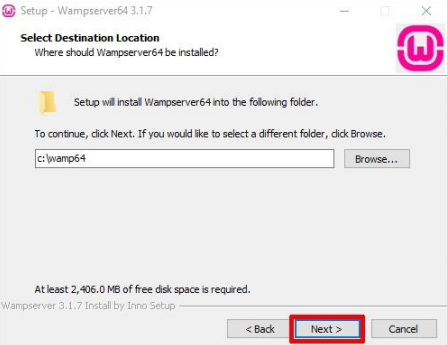
Then after selecting the language the license agreement will appear. Click the "I accept the agreement" button and then the "Next" button to proceed with the installation described below.



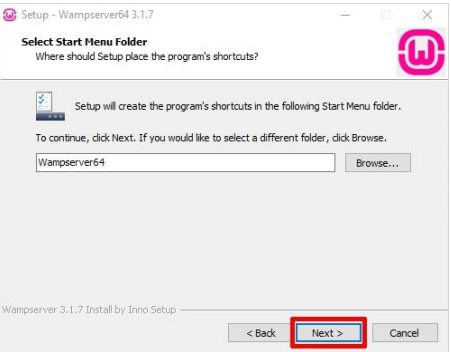
The next screen will tell you the required components that must be present in your computer system for the proper functioning of WampServer. The main purpose of this step is to check for the suitable version of Microsoft VC++ redistributable package. Click on the ‘Next’ to continue.

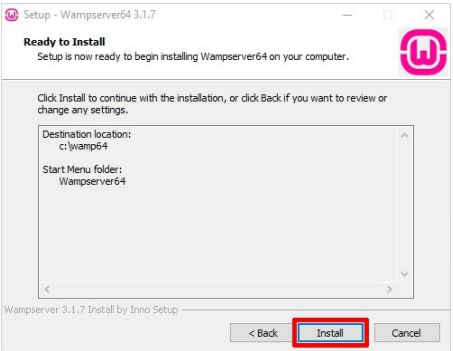


Step 3: Select the location to install WAMP, you need to select the location to install WAMP on your computer. However, you can accept the default location to continue the installation, press the "Next" button.



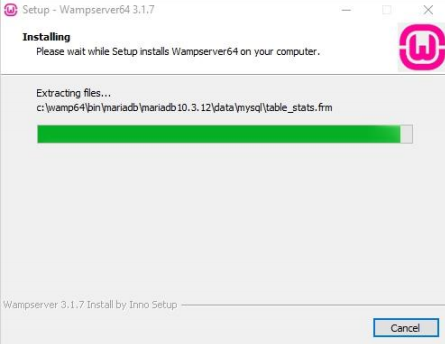
Step 4: Start Menu Folder must be selected to install WAMP. You have to select any folder of your choice or you can proceed with the default option. After selecting the folder, click "Next" to continue.

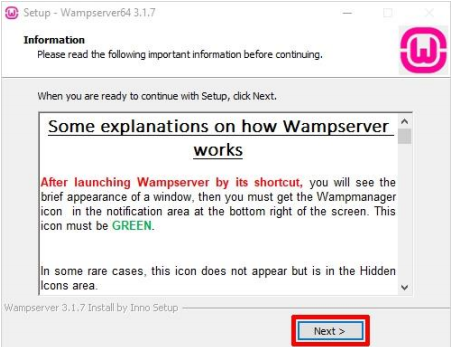


Step 5: Ready to Install WAMP Finally, you will see a ready-to-install wizard with a setup ready to begin the installation. Just click on the "Install" button to start the installation procedure.

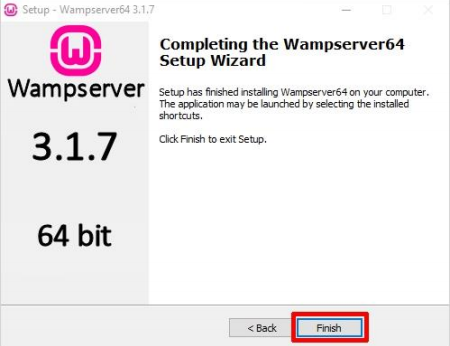
Now, just be patient because WampServer extracts the files to your specified location. Leave

The process is complete.



Once the status bar turns completely green, an information screen like the one shown below will appear. Click Next to continue.

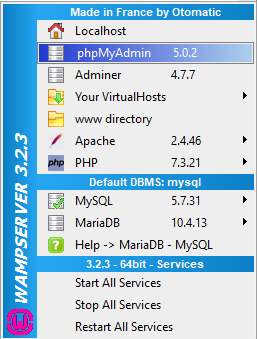
Step 6: WAMP installation is complete, just click Finish to exit Setup.



That’s all. You are ready with the WAMP server now

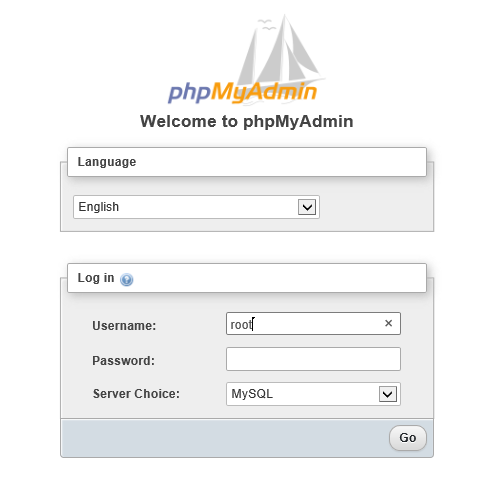
How to run a wamp server 64 program?

First, click on the program on the desktop.



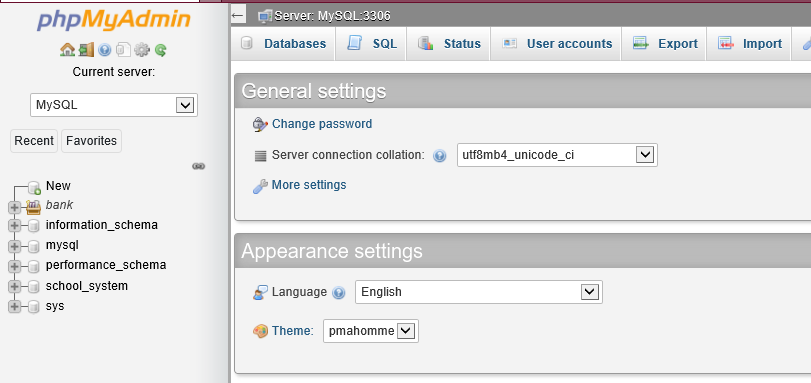
Second, we click on it from here.

Third, We click on phpMyAdmin.



Fourth, we type the username and then we click on Go.

Fifthly, the program's home page appears.



# 5 test the system against user and system requirements + output and input validation:

Tests:

It aims to detect flaws within the system as well as check if the application is behaving as expected and as documented in the requirements analysis stage.

## output and input validation:

**Primary key:** The candidate key best suited to be the table's primary reference key (the chosen key).

1-Can not accept null values.

2-Only one primary key in a table.

3-uniquely identify a record in the table.

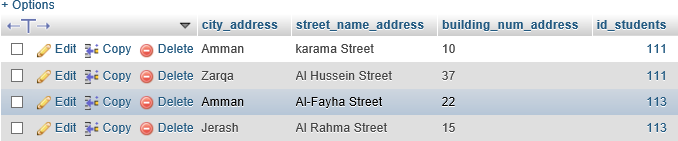


INSERT INTO students

VALUES(111,'Sara','Al-hgftm', '2003-02-18','Bad Health',1);



Default: Provides a default value for a column when no value is available when a record is inserted into a table.

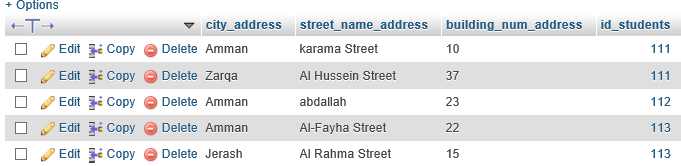


INSERT INTO students\_address(street\_name\_address, building\_num\_address, id\_students)

VALUES ('abdallah',23,112);



NOT NULL: makes sure that a column does not hold NULL value.



INSERT INTO students\_address

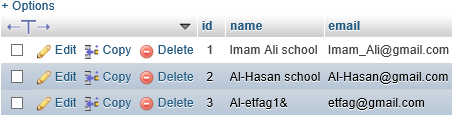
VALUES('Maan',32,112);



**VARCHAR(size):** A variable-length string (can contain letters, numbers, and special characters). Size specifies the maximum column length in characters - can range from 0 to 6553.

INSERT INTO school

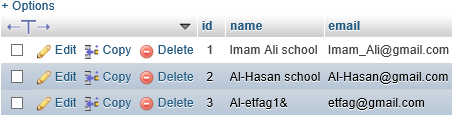
VALUES(3,'Al-etfag1&','etfag@gmail.com');



**INT(*size*):** An average integer. The signed range is from -2147483648 to 2147483647. The unsigned range is 0 to 4294967295. The size parameter specifies the maximum width of the width (which is 255).

INSERT INTO school

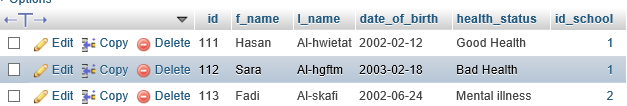
VALUES(3,'Al-etfag1&','etfag@gmail.com');



Date: a date. Format: YYYY-MM-DD. The supported range is from "1000-01-01" to "9999-12-31".

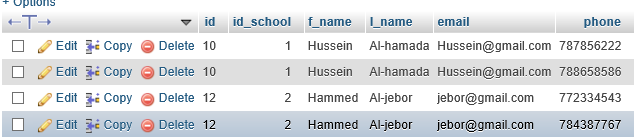
INSERT INTO students

VALUES (113,'Sara','Al-hgftm', '2003-02-18','Bad Health',1);

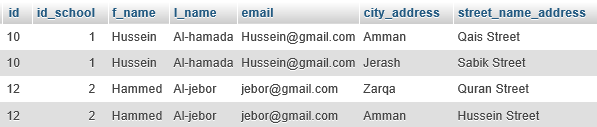


## Test the created views and see if they work correctly:

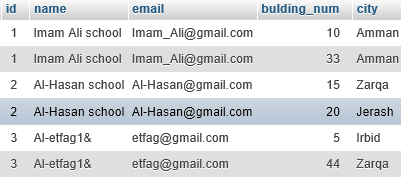
SELECT \* FROM teachers\_info1;



SELECT \* FROM teachers\_info2;

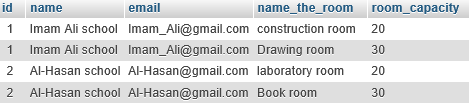


SELECT \* FROM school\_info;



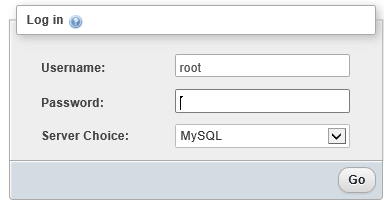
## Test the created procedures and see if they work correctly:

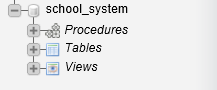
CALL fadi();



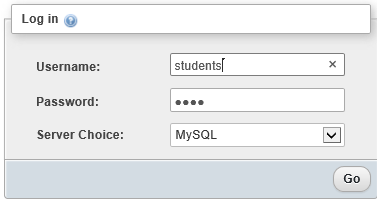
## Test the created Users and see if they work correctly:

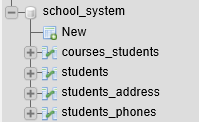
The root controls the entire database.



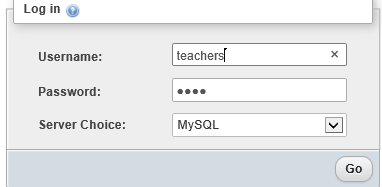


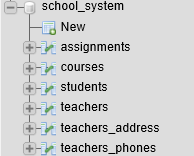
Students: can choose the teacher they want and can the courses schedule either over the phone or using the school's website. also should be able to access the database using a web browser to request grades and submit assignments.





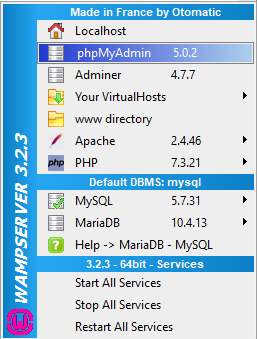
Teachers: storing their data They can browse information about their students, ask them for homework, and send their grades to Administrative staff or download over the school website.





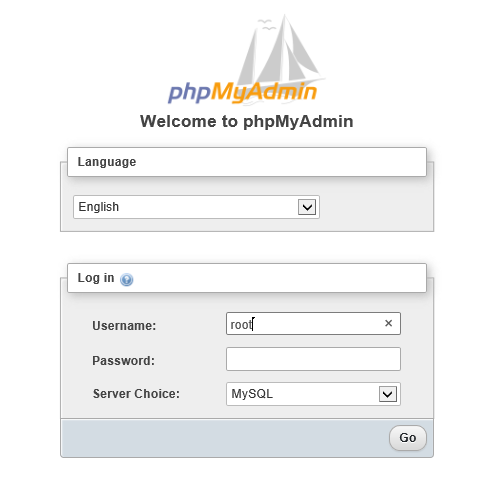
## How to run a wamp server 64 program?

First, click on the program on the desktop.

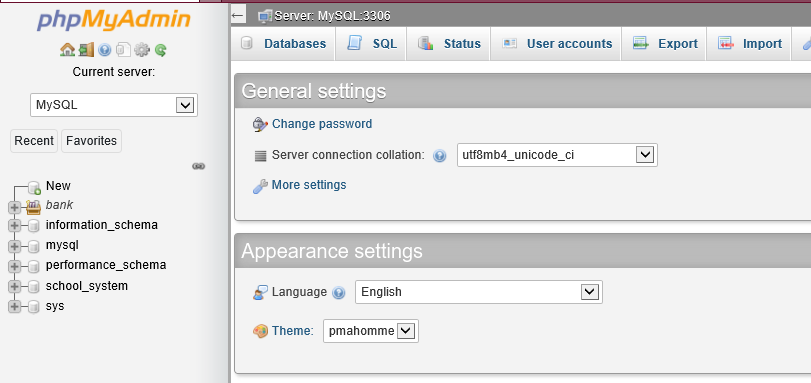


Second, we click on it from here.

Third, We click on phpMyAdmin.

Fourth, we type the username and then we click on Go.

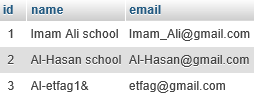
Fifthly, the program's home page appears.



## A test to show the data contained in the database:

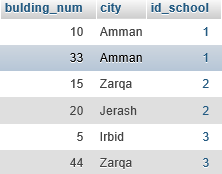
Show the first 3 schools in the database.

SELECT \* FROM school LIMIT 3;



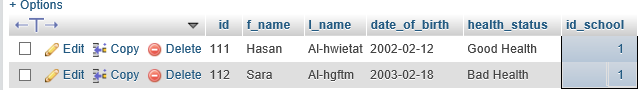
Show all addresses related to the schools.

SELECT \* FROM school\_address;



Show all data students when id\_school=1.

SELECT \* FROM students WHERE id\_school= 1;

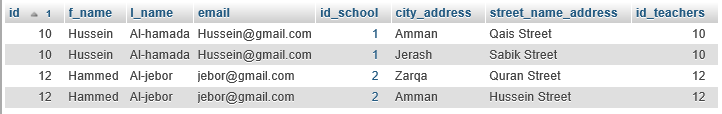


Sort data for teachers in descending order.

SELECT \* FROM teachers INNER JOIN teachers\_address

ON teachers.id=teachers\_address.id\_teachers

ORDER BY teachers.id ASC;



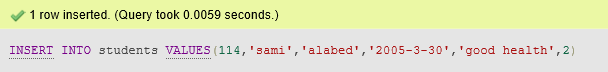
## Test the system against user and system requirements:

A student can enter his data and update it if necessary, and he cannot see the information of other students and he cannot see information related to the teacher.

The student is allowed to insert the first and second name, date of birth, health status and school ID.

INSERT INTO students

VALUES(114,'sami','alabed','2005-3-30','good health',2);



A student is not allowed to see all other students' data.

SELECT \* FROM students;



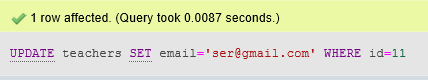
A teacher can enter his data and update it if necessary, and he can see student information, but he cannot see student information such as phone and place of residence, and he can see the number of students who have registered in the course he gives.

The teacher can update his information.

UPDATE teachers

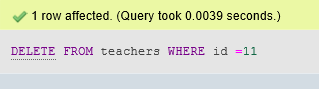
SET email='ser@gmail.com'

WHERE id=11;



The teacher can delete his information.

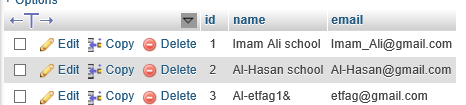
DELETE FROM teachers WHERE id =11;



The root can enter the data of students and teachers and distribute them to schools and to know the distinguished students and he can see all the information related to the students and he can see all the information related to the teachers and he can see the number of students who have registered in the course given by the teacher and he can count the number of students in the school the one.

root can create views and create procedures and create tables and insert and select and update and delete data.

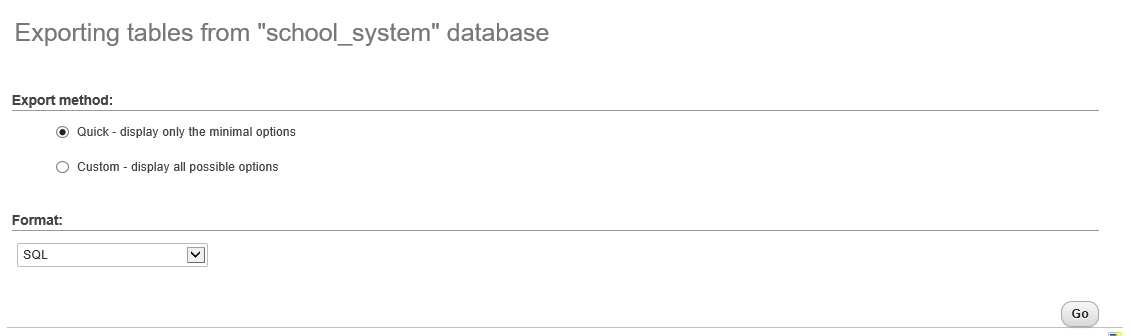
Select \* from school;



# 6database maintenance:

To get a backup of your database, you should follow the next steps:

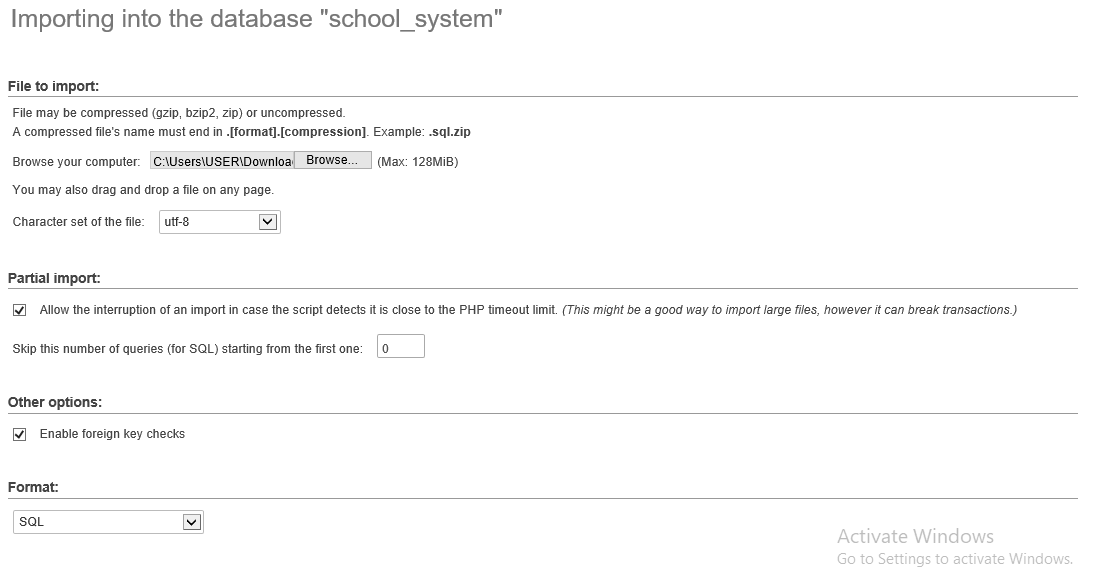
1. Run the **WampServer** and then open the **phpMyadmin** in your browser.
2. On the left, **select** the database that you will be working.
3. Click **Export** in the top menu.
4. Under File **format**, make sure you have selected the **SQL** option.
5. Click **Go** at the bottom right to **export** the database SQL file.
6. When the database has been exported **successfully**, you should see the **downloaded file** in the download folder.





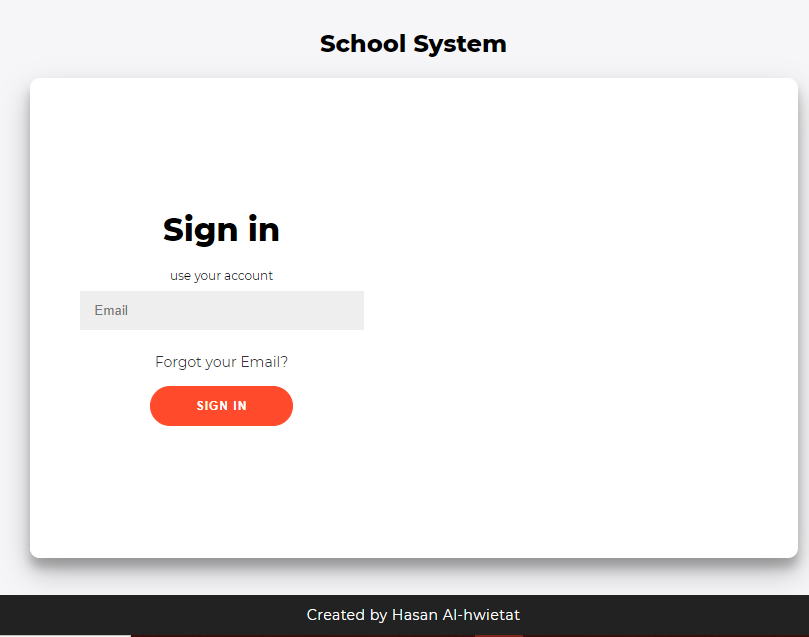
To import a backup of your database, you should follow the next steps:

1. Run the WampServer and then open the phpMyadmin in your browser.
2. Click SQL in the top menu.
3. Run a statement that creates a new database.
4. On the left, select the database that you created.
5. Click Import in the top menu.
6. Under File to Import, click Browse and select the db.sql file.
7. Click Go at the bottom right to import the database file.
8. When the database has been imported successfully, you should see a message at the top of the page similar to: Import has been successfully finished, ## queries executed.

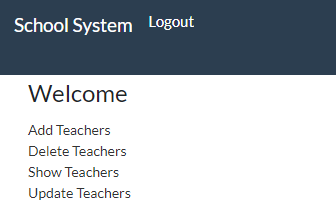


# 7 interface and output designs:

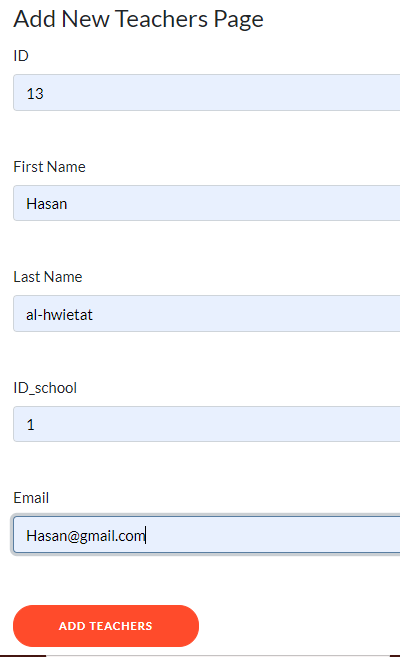
Home Page You must enter your Email to enter the next page.



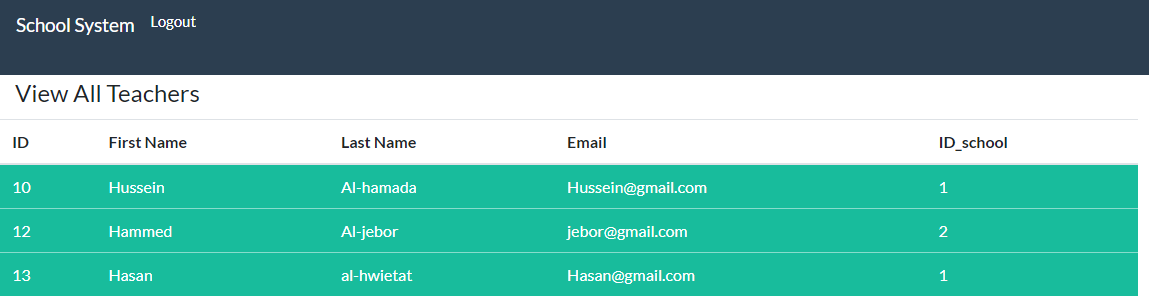
The page that can do anything you want to do from adding a teacher, removing a teacher, showing all teachers, updating the data to move to the next page, or exiting and returning to the home page.



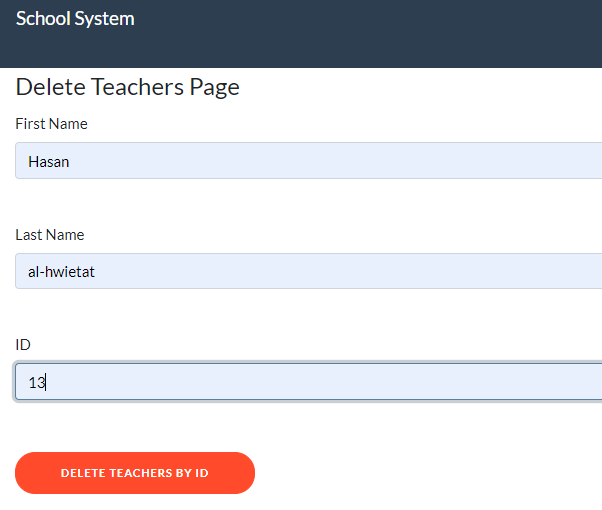
On this page you can add a feature and then return to the previous page.



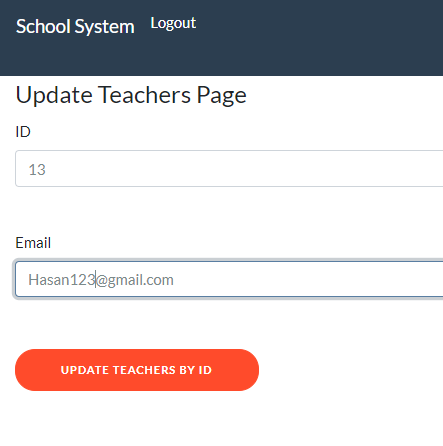
On this page you can see all the teachers in this school and then click logout to return to the main page.



On this page, you can remove the teacher from this school and then click on Delete to return to the previous page.



On this page, you can update the teacher's email from this school, then click on Update and return to the previous page.



# 8 Diagrams and flowcharts:

Flowcharts:

is a diagram that depicts a computer process, system, or algorithm. They are used extensively in multiple fields to document, study, plan, improve and communicate often complex processes in clear, easy-to-understand diagrams.

T

F

Database to update teachers

Database to show teachers

Database to delete teachers

Database to add teachers

Click to Update Teachers

logout

T

Click to Show Teachers

Click to Delete Teachers

T

Click to Add Teachers

T

T

T

F

Check Email

Enter the Email

If(finished)

Diagrams: A diagram is a line drawing comprised of symbols that represent key objects and actions that occur within a system. The type of system often determines the set or library of symbols used within a particular diagram.



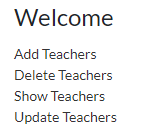
Open the website.



Sign in to the site using your email.

Go to the home page and click on one of the options you want.

Logout





Go to the page you choose.

# 9 Evaluate the effectiveness of the design in relation to user and system requirements.

The design was suitable for the needs of the user. I put in the database three users of the system, they are

1. Teachers: storing their data They can browse information about their students, ask them for homework, and send their grades to Administrative staff or download over the school website, and know the name of the school in which he teaches.
2. Students: can choose the teacher they want and can the courses schedule either over the phone or using the school's website, and know the name of the school in which he studies. also should be able to access the database using a web browser to request grades and submit assignments.
3. root: He can do all the things he wants from seeing the number of students in the school, the number of teachers, the subjects that are given in this school and the first ones who got the highest marks.

All the users were positioned correctly. The graphical User Interface was created based on the User Needs, the design correctly came based on the connection between the table and other tables. Also, the entire system requirements were present correctly and within the needs of the users and the graphical User Interface.

# 10 Evaluate the effectiveness of the database solution in relation to user and system requirements, and suggest improvements.

Database development is a series of stages or steps, each of which focuses on one aspect of development, so any data system always needs to be developed until it is permanently discontinued from use. Ideally, you would validate each stage of development before moving on to the next. Therefore, one of the most important principles of good database design is to avoid data redundancy, validity, accuracy and integrity of information. A feature of good design is to divide the information into tables with headings to reduce redundancy and to organize the data well. Database systems are basically developed for large amounts of data. When dealing with a huge amount of data, there are two things that require optimization: Storage of data and retrieval of data.

# 11 Evaluate the database in terms of improvements needed to ensure the continued effectiveness of the system.

In order to develop and improve the data that I have provided so that it is fully functional and to ensure its continuity, I suggest that I make a new table containing the installments to be paid by the student. I mean the accounts department may know that this student is how much he has to pay this month's fee and the total amount of the amount owed. It is also possible to add new attributes to the tables in the data, which increases the users' permissions, but the more data, the more users and more devices that use the data. Which makes the data meaningful and sophisticated, but all this will not affect the data base and make it not last for a long time. So, it is possible to add a new table or attribute and it will not affect the data and it will work for a long time.