

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **XYZ Software Solutions**  23/12/2018 | |  | | |
|  |  |  | | |  | |
|  |  | Trinity Database  Institutional record management system | | |  | |
|  |  | | | | |  |
|  | **Introduction**  Trinity Music School based in Colombo currently practices disorganized and out of date, data processing methods in order to meet the demands in the growth of the school and flexibility, the school has request XYZ Software Solutions to create a relational database system to completely replace their school based record management system.  This report will attempt to justify, explain and clarify the specifics and semantics of the relational database that has been created per their request.  Author: K.P.I. Shenesh Perera  IDM | | | | |  |
|  |  | |  | | | |

**Contents**

[Modelling the Database. 3](#_Toc1410083)

[Entity Relationship Diagram. 3](#_Toc1410084)

[Use Case Diagram. 5](#_Toc1410085)

[Specification for the intended users of the system. 6](#_Toc1410086)

[Identified requirements of the system. 6](#_Toc1410087)

[Data Normalization 7](#_Toc1410088)

[Class Table: 8](#_Toc1410089)

[Supplier Table: 8](#_Toc1410090)

[Instrument Table: 8](#_Toc1410091)

[Student Table: 9](#_Toc1410092)

[Teacher Table: 9](#_Toc1410093)

[Attendance Table: 9](#_Toc1410094)

[Class\_subjects Table: 10](#_Toc1410095)

[System Designs 10](#_Toc1410096)

# Modelling the Database.

## Entity Relationship Diagram.

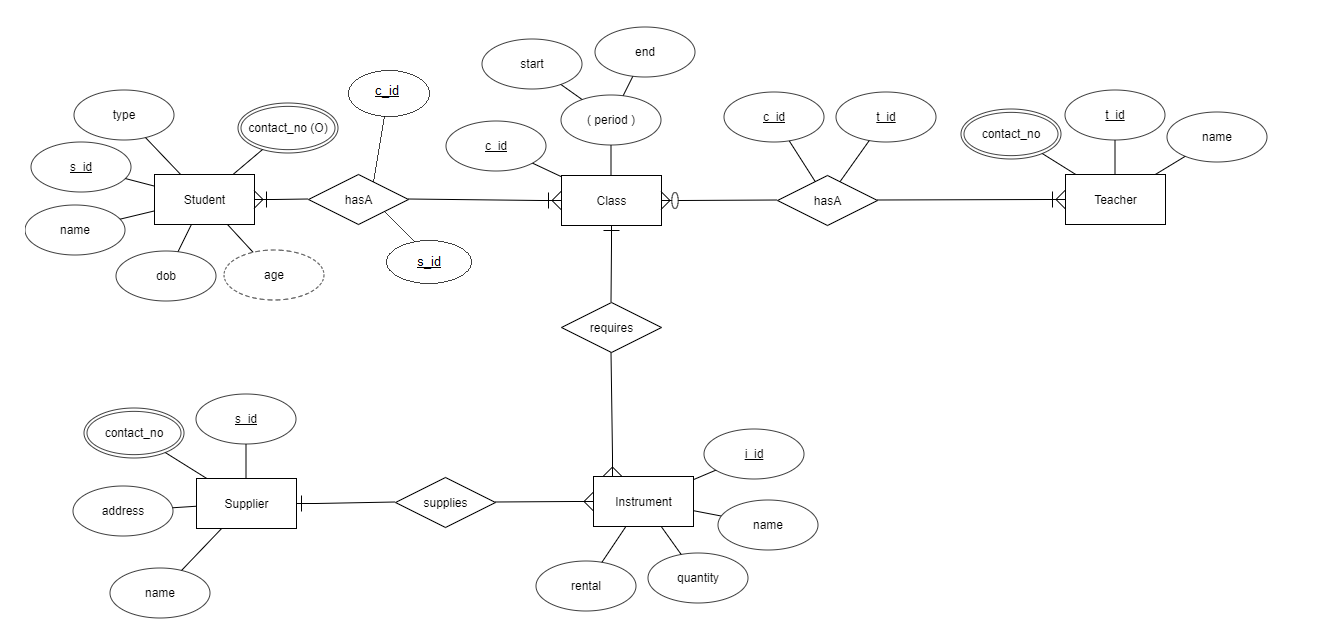


Figure 1.0, Shenesh Perera, 15/12/2018

The Entity Relationship Diagram (ERD) was used as modelling technique to model the database. During the modelling stage, per the school’s scenario and the requirements stated, 5 different entities and their respective attributes were identified. Upon closer inspection, it was found that these entities had relationships in between them. After thorough inspection, the relationships were determined and appropriately modelled into the diagram.

The 5 entities and their properties:

1. Student – Is in a many-to-many mandatory participation relationship named “hasA” with entity Class.
2. Class – Is in a many-to-many mandatory participation relationship named “hasA” with entity Student**,** is a many-to-many optional participation relationship named “hasA” with entity Teacher **and** Is in a one-to-many relationship named “requires” with entity Instrument.
3. Teacher – Is in a many-to-many mandatory participation relationship named “hasA” with entity Class.
4. Instrument – Is in a one-to-many relationship named “requires” with entity Class **and** Is in a one-to- many relationship named “supplies” with entity Supplier.
5. Supplier – Is in a one-to-many relationship named “supplies” with entity Supplier.

However, during data normalization and the actual development of the database the entities, their relationships, attributes and certain other factors changed in order to ensure best functionality.

## Use Case Diagram.

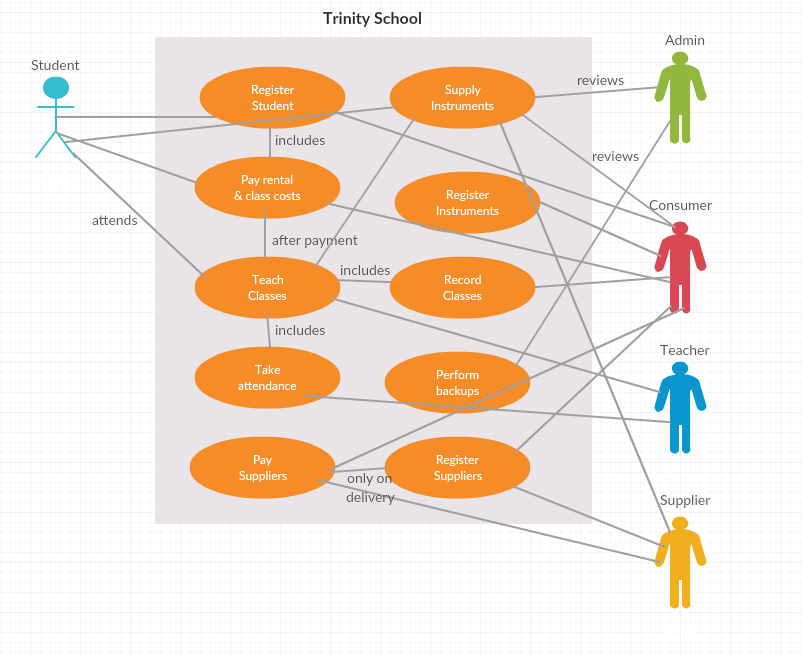


Figure 1.1, Shenesh Perera, 15/12/2018

The Use Case Diagram (UCD) was used as modelling technique to model the database. This technique was used to identify potential users of the system, classify them and identify how and where the data flows within the school. In this modelling stage, users were classified. The actors displayed in the diagram perform each of those actions.

During this modelling stage, 3 users (Admin, Consumer, Supporter) were identified in addition to the 3 stated in the scenario (Student, Teacher, Supplier).

The supporter user is not demonstrated here as that type of user is intended to not make any interactions with the system, but only exist as a side-hand. This type of user was concluded to be optional.

The admins and consumers will make direct interactions with the system, while teachers and students will be the users that will influence the actions of the formerly stated admins and consumers.

However, it is best to note that the actual system may or may not follow these same declared data flows or support for the declared actors.

# Specification for the intended users of the system.

All users of this system must generally satisfy the following criteria:

1. Have a history of decent exposure to other well-known application software.
2. Must strictly remain within their scope of action.
3. Have a primary sense of reporting any and all strange behaviors to the school’s IT staff.
4. Must obey all school practices.

Users have been categorized to 3 groups, depending on the role they have to perform in order to extract the best of the system. Namely, Admins, Consumers and Supporters.

Admin users have read/write permissions, can change passwords of consumers and other admins, can create consumer accounts, can backup & restore the database and can completely erase all data within the database. Admin users are suggested to not frequently operate the system. Admins can have consumer accounts to participate in read/write actions instead.

Consumer users have read/write permissions for all forms across the system. These users are the ones that will actively and frequently operate the system by interacting with students, suppliers and all other external audiences.

Support users do not interact with the system at all. They exist in order to ensure that the system functions well and to serve the purpose of reporting management level or the IT staff when an error of sufficient magnitude has occurred.

It is strictly suggested that these 3 types of users are granted to operatives within the company. In addition to the IT staff and front office operatives only teachers may be bestowed upon one of these user roles.

# Identified requirements for the system.

Upon processing the scenario, before the modelling stages of the system several requirements were identified.

Trinity’s primary requirement is to be provided with a database to keep records of:

1. Classes.
2. Teachers that teach on a class.
3. Students that participate in a class.
4. Instruments rented for a particular class.

Trinity’s secondary requirements that were identified:

1. Class records must contain class code, instrument taught, teachers, number of instruments rented, start/end data and timing attributes.
2. Instrument records must contain instrument type code, instrument name, supplier company and contact number attributes.
3. Student records must contain student name, student type and class attended attributes.

Trinity’s tertiary requirements that were identified:

1. Enforce each type of musical instrument being supplied by only one company.
2. Enforce rental formula as “**NumberOfInstruments** \* **Period**”.
3. Enforce one class can have many students.
4. Enforce one student can have many classes.
5. Enforce student classification of types “Full\_Time”, “Part\_Time” and “Part\_Time\_Evening”.
6. Enforce one teacher can have many classes.
7. Enforce one class can have many teachers.

All primary, secondary and tertiary requirements have been attempted to be factored into the system by creating a model that respects the nature of these requirements, and a database that can obey the nature of the model.

# Data Normalization

Data normalization is a systematic process that organizes data within a database, by decomposing tables to get rid of data repetition and other problematic anomalies the main 3 Insertion, Update and Deletion anomalies. This process consists of a lot of steps by which performing a particular step will give a table that takes one step towards a better structure.

1st normal form (1NF):

1. Must only have atomic valued columns.
2. All values in a column must be of the same data type.
3. All columns in the table must have unique names.
4. The order of data storage does not matter.

2nd normal form (2NF):

1. Must be in 1NF.
2. Must not contain any attributes depending only on part of a composite primary key.

3rd normal form (3NF):

1. Must be in 2NF.
2. All attributes in the table must only depend on the prime attributes of the primary key.

There are 2 primary advantages of having a normalized database:

* Increased consistency: Information is stored in one place and one place only, reducing the possibility of inconsistent data.
* Easier object-to-data mapping: Highly-normalized data schemas in general are closer conceptually to object-oriented schemas because the object-oriented goals of promoting high cohesion and loose coupling between classes results in similar solutions.

You’d most likely want to have normalized data stores and data warehouses.

The primary disadvantage of normalization is slower reporting performance. You’d want to have a denormalized schema for efficient reporting, particularly in data marts.

All tables will be normalized to the third normal form. Below each table that exists in the database will be visualized, normalized and justified.

## Class Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Class | | | |
| c\_id (Integer) | c\_start (datetime) | c\_end (datetime) | c\_type (varchar(50)) |
| 1111 | 20181018 08:00:00 AM | 20181018 01:34:09 PM | FULL\_TIME |

Primary key: c\_id

This table is in the 3rd normal because:

1NF – All values are atomic, have the same data type and columns have unique names.

2NF – Is already in 1NF and all attributes depend on the primary key c\_id.

3NF – Is already in 2NF and all attributes depend only on c\_id.

## Supplier Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Supplier | | | |
| supp\_id  (integer) | supp\_address  (varchar) | supp\_name  (varchar(100)) | supp\_contact\_no  (varchar(20)) |
| 100 | St Joseph''s St, Negombo | Yamaha Dealer | 031892834 |

Primary key: supp\_id

This table is in the 3rd normal because:

1NF – All values are atomic, have the same data type and columns have unique names.

2NF – Is already in 1NF and all attributes depend on the primary key supp\_id.

3NF – Is already in 2NF and all attributes depend only on supp\_id.

## Instrument Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Instrument | | | | |
| i\_id  (Integer) | supp\_id  (Integer) | i\_name  (varchar) | i\_quantity  (Integer) | i\_rental  (money) |
| 103 | 100 | Violin | 13 | 20398.00 |

Primary Key: i\_id

This table is in the 3rd normal because:

1NF – All values are atomic, have the same data type and columns have unique names.

2NF – Is already in 1NF and all attributes depend on the primary key i\_id.

3NF – Is already in 2NF and all attributes depend only on i\_id.

Supp\_id is a foreign key referencing the column supp\_id from the supplier table, but this column depends entirely on the i\_id in this table.

## Student Table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Student | | | | | | |
| s\_id  (Integer) | i\_id  (Integer) | s\_name  (varchar) | s\_type  (varchar) | s\_regist\_date (datetime) | s\_dob (datetime) | s\_contact\_no  (varchar) |
| 1 | 103 | Saline Man | FULL\_TIME | 20181018 00:00:00 AM | 20000928 00:00:00 AM | 076 1234567 |

Primary key: s\_id

1NF – All values are atomic, have the same data type and columns have unique names.

2NF – Is already in 1NF and all attributes depend on the primary key s\_id.

3NF – Is already in 2NF and all attributes depend only on s\_id.

i\_id is a foreign key referencing the column i\_id from the instrument table, but this column depends entirely on the s\_id in this table.

## Teacher Table:

|  |  |  |
| --- | --- | --- |
| Teacher | | |
| t\_id (Integer) | t\_name (varchar) | t\_contact\_no (varchar) |
| 10001 | Dumbledore McField | 079 123467 |

Primary key: t\_id

1NF – All values are atomic, have the same data type and columns have unique names.

2NF – Is already in 1NF and all attributes depend on the primary key t\_id.

3NF – Is already in 2NF and all attributes depend only on t\_id.

## Attendance Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attendance | | | | |
| a\_index | c\_id  (Integer) | s\_id  (Integer) | attend\_time  (datetime) | daily\_cost  (money) |
| 10 | 1111 | 1 | 20181018 08:05:43 AM | 1000.00 |

This table has been created to save the primary keys of class and student as class and student have a many -to-many relationship, as such they’re foreign keys. The primary key is a\_index and all attributes depend on it.

1NF – All values are atomic, have the same data type and columns have unique names.

2NF – Is already in 1NF and all attributes depend on the primary key a\_index.

3NF – Is already in 2NF and all attributes depend on the primary key a\_index.

## Class\_subjects Table:

|  |  |  |  |
| --- | --- | --- | --- |
| class\_subjects | | | |
| cs\_index | c\_id | t\_id | c\_subject |
| 10 | 1111 | 10001 | Cleaning Instruments |

This table has been created to save the primary keys of class and teacher as class and teacher have a many -to-many relationship, as such they’re foreign keys.

1NF – All values are atomic, have the same data type and columns have unique names.

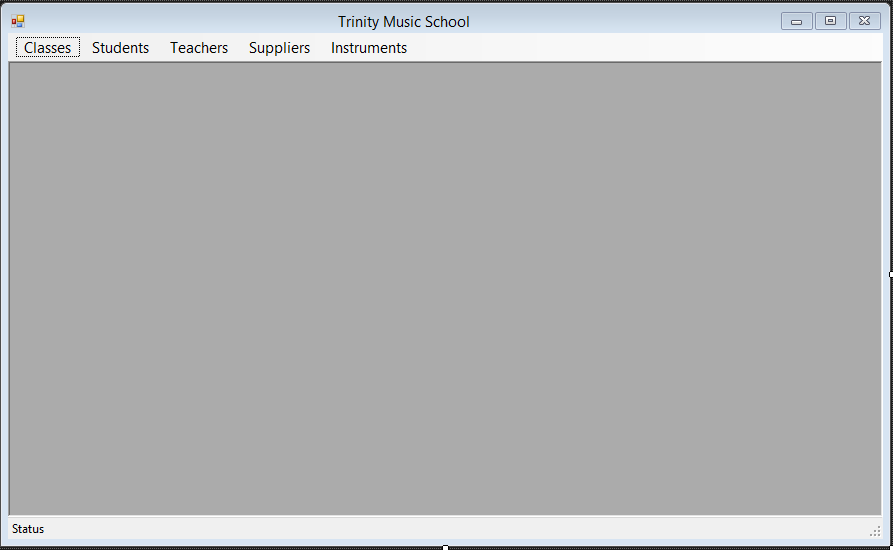
2NF – Is already in 1NF and all attributes depend on primary key cs\_index.

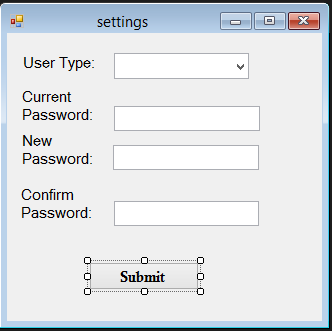
3NF – Is already in 2NF and all attributes depend on the primary key cs\_index.

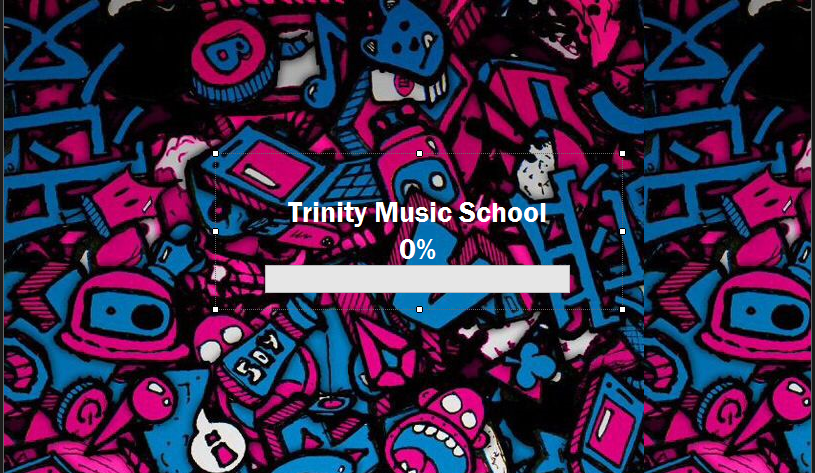
Justification for c\_subject, one may argue that c\_subject only depends on the class but not the teacher, but one most know that this table can be used to identify what subject that a teacher teaches in a particular class, so c\_subject depends on both c\_id and t\_id and therefore is **not** a transitive dependency.

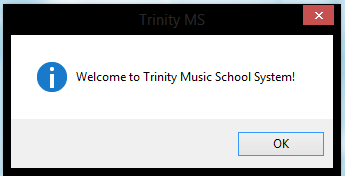
# System Designs

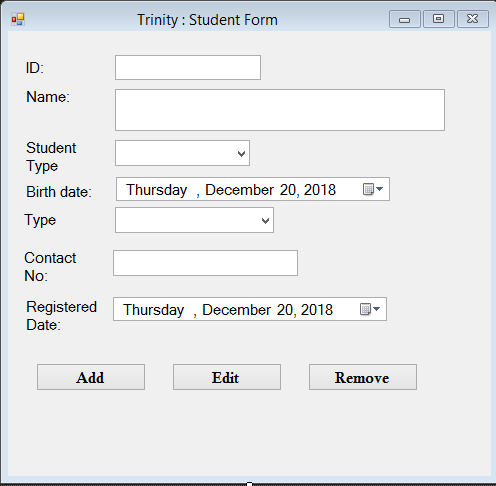
The designs are self-explanatory and have been created to be as minimal as possible while maintaining user-friendliness. Following are some screenshots of the design of the system:











Justification of system design

Introduction

This will be a brief report on how much the design of the system have stood by the requirements of both the user and the system.

*What is system design?*

System design is the process of designing the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system.

Assessment on the effectiveness of the design

User Requirements

The design has been done in a way that anyone with previous experience to some sort of application software like one from the Microsoft office suite will be capable of easily navigating through the system and perform whatever task they wish to perform.

Whenever the system encounters an error the user will be notified using a message box, the user must then be capable of relaying this to a supporter or a knowledgeable IT staff member that can provide them the necessary support, or refer to the technical documentation for assistance.

With this relational database system, all hassles that were experienced by Trinity will be effectively overcome but in order to this, the users must be trained to use the system.

Referring to the user documentation during the initial few days of exposure to the system is highly recommended.

A satisfactory attempt was made to satisfy all user requirements demonstrated in the scenario.

System Requirements

All primary requirements have been achieved due to the existence of the respective tables that would replace Trinity’s records. A table has been created to satisfy each problem in the primary requirements.

Data validation has been achieved through enforcing datatypes by embedding of a schema to each table in the database. Only data of the defined data type can be stored within each table.

Further constraints like NOT NULL, UNIQUE, PRIMARY KEY or FOREIGN KEY have been enforced in order to make sure that the system performs ideally and the data stored is consistent.

In order to satisfy all secondary requirements, during modelling stages respective entities were assigned with the attributes that have been declared in the scenario and were later converted to columns of a particular table in the development stages.

All enforcements described as tertiary requirements have been achieved through FOREIGN KEY constraints, composite primary keys and at times separate tables in order to ensure the various relationships between each entity of the modelling stage.

Since all 3 types of system requirements have been achieved, it can be said that the system has been designed to satisfy all requirements.

As the design of the system satisfies both the user and system requirements, it is perfectly valid to say that the system is capable of performing all demands of Trinity Music School ideally and effectively.

# Bibliography

* Medium.com
* Stackoverflow.com