# Database Project

CS3014 – Information Management – Ruth Brennan – 17329846

## Description of the database

I made a database for the Trinity CoderDojo. CoderDojo is a global network of free, volunteer-led, community-based programming clubs for young people. Anyone aged seven to seventeen can visit a Dojo where they can learn to code, build a website, create an app or a game, and explore technology in an informal, creative, and social environment. Trinity CoderDojo is one such club.

This Dojo runs weekly events called sessions as well as one of events throughout the year. All of these events have names, topic covered, a director, a date and time, a location and an ID. For example, the session this Saturday the 30th of November takes place from 10-12, in room 1.07 in the Lloyd Institute, Mary will be the director, the topics covered will be Scratch, Makey Makey, Python and P5Js and the event ID will be SE51. Where SE signifies session and 51 is a number to uniquely identify the event.

The dojo practices self-directed learning though the dojo tries to focus the kids on topics which they are prepared for. Such topics are selected as mentors are well versed in them, have rough syllabuses designed, have resources/equipment available readily. As such each topic has a syllabus director, this is someone who is in charge of updating the syllabus, gathering resources/equipment for this topic and training new mentors in this area. The Ninjas are asked to bring their own laptops so the only equipment that the dojo provides would be extra to that for example Arduino or Makey Makey kits and the accessories pursuant to those.

The kids that attend the dojo sessions are referred to as Ninjas. The dojo keeps a record of their name, age, date of birth (dob), topics they are interested in as well as information on one parent/guardian. The dojo keeps contact information for that parent so that they can contact them about upcoming events or in case of an emergency.

The volunteers at Trinity CoderDojo are all students in Trinity with technical experience relevant to the content that they mentor the ninjas in. The dojo keeps a record of mentor’s name, the topic they are comfortable and ready to mentor in and an ID code to uniquely identify them. They also keep track of a mentor’s level based on a system structured as follows:

Junior Mentor

Yellow Belt

Senior Mentor

Green Belt

Committee Mentor

Black Belt

Coordinating Mentor

Purple Belt

Contributing Mentor

White Belt

They use the mentor hierarchy when assigning responsibilities and mentoring opportunities. It is primarily based on how often mentors attend events as well as what organizational aspects they help with outside of the events. The Committee Members are mentors who have worked their way through the hierarchy and hold one of the six positions on the committee. Champion, Information Officer, Promotional Officer, Treasurer and Child Protection Officer. Each with responsibilities pursuant to their title. Champion is the CoderDojo Organization’s term for a president of the club. The club holds meetings for the mentors to plan, organize and restructure the dojo.

## Relational Model

A picture containing sky

Description automatically generated

\*Underlined Attribute indicates a Primary Key

## Functional dependancy

A screenshot of a video game

Description automatically generated

\*Underlined Attribute indicates a Primary Key (Red Box), Italic indicates a foreign key (Orange Box)

In my initial design, I planned to keep track of parents and ninjas in the same table. However, one parent can have multiple children so to reduce repeat data and redundancy I split them into two tables. Using the Parent ID to uniquely identify the parent in the Ninja table.

## Semantic Constraints

There are a few semantic constraints that apply to this database.

### Entity Integrity Constraints

The entity integrity constraint states that primary key value can't be null. This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows. A table can contain a null value other than the primary key field.

### Referential Integrity Constraints

These are defined as part of an association between two entity types. The definition specifies: The principal end of the constraint: the foreign key.

ALTER TABLE Ninjas ADD CONSTRAINT parent\_id FOREIGN KEY (parent\_id) REFERENCES Parents(phone\_number);

ALTER TABLE Ninjas ADD CONSTRAINT ninja\_group\_id FOREIGN KEY (group\_id) REFERENCES Ninja\_group(id);

ALTER TABLE Committee\_position ADD CONSTRAINT mentor\_position FOREIGN KEY (position) REFERENCES Committee\_position(id);

ALTER TABLE Topics ADD CONSTRAINT topic\_director FOREIGN KEY (syllabus\_director) REFERENCES Mentors(id);

ALTER TABLE Meetings ADD CONSTRAINT meeting\_chair FOREIGN KEY (chairperson) REFERENCES Mentors(id);

ALTER TABLE Meetings ADD CONSTRAINT meeting\_minutes FOREIGN KEY (minute\_taker) REFERENCES Mentors(id);

ALTER TABLE Events ADD CONSTRAINT event\_director FOREIGN KEY (director) REFERENCES Mentors(id);

ALTER TABLE Events ADD CONSTRAINT event\_topic FOREIGN KEY (topic) REFERENCES Topics(name);

ALTER TABLE Events ADD CONSTRAINT event\_group FOREIGN KEY (group\_id) REFERENCES Ninja\_group(id);

ALTER TABLE Ninja\_group ADD CONSTRAINT group\_topic FOREIGN KEY (topic) REFERENCES Topics(name);

### Table Constraints

1. CHECK

We start out by looking at the ID constraints:

An event’s and meetings ID is a five-digit number, where the first two digits represent the year eg 2019 – “19101”. The last two digits accumulate start with 00. If the last number is 1 it signifies an event if it is 2 it signifies a meeting.

The Id for the Ninjas (the kids that attend the dojo) is another five-digit number where the first two digits correspond like above to the year in which they joined the dojo. And the last digit is a 3. I considered using PPS numbers but the collection of that information felt unnecessary and would cause issues with GDPR.

The Id for Mentors is their TCD student number, this is a unique eight-digit number generated by the college e.g. 17329846. This is collected during application as all mentors have to be Trinity Students because the dojo is sponsored by the college.

We can use CHECK to maintain correctness in our number (int) attributes.

ALTER TABLE Ninjas ADD CONSTRAINT ninja\_age CHECK (TIMESTAMPDIFF(YEAR, dob, '2019-11-06') > 5 AND TIMESTAMPDIFF(YEAR, dob, '2019-11-06') < 17);

There are also a number of constraints on String attributes.

There are a fixed number of committee positions, 6. The Champion, the Treasurer, the Information Officer, the Promotion Officer, the Child Protection Officer and 5 OCM (Ordinary Committee Member) positions. The 5 titled positions have to be filled but the OCMs are optional.

There are a fixed number of levels which the mentors can achieve. In this case 5 as listed above; White belt, Yellow Belt, Green Belt, Purple Belt and Black Belt.

In a similar manner there are a fixed level that the ninjas can reach with their topic; 1-creator, 2-builder, 3-developer, 4-maker.

ALTER TABLE Committee\_position ADD CONSTRAINT committee\_id CHECK (id<7 AND id>0);

ALTER TABLE Mentors ADD CONSTRAINT position\_id CHECK (position<7 AND position>0);

ALTER TABLE Mentors ADD CONSTRAINT mentor\_level CHECK(level IN ('White', 'Yellow', 'Green', 'Purple', 'Black'));

ALTER TABLE Committee\_position ADD CONSTRAINT position\_name CHECK(position IN ('Champion', 'IO', 'CPO', 'PRO', 'PRO', 'Treasurer', 'OCM'));

ALTER TABLE Committee\_position ADD CONSTRAINT committee\_id CHECK (id<7 AND id>0);

ALTER TABLE Topics ADD CONSTRAINT topic\_director CHECK(level IN ('Creator', 'Builder', 'Maker', 'Developer'));

ALTER TABLE Ninjas ADD CONSTRAINT ninja\_age CHECK (TIMESTAMPDIFF(YEAR, dob, '2019-11-06') > 5 AND TIMESTAMPDIFF(YEAR, dob, '2019-11-06') < 17);

## Database Security Commands for Access and Security Policy

We now take a look at database security. Integrity is concerned with accidental corruption. Security is concerned with deliberate corruption. To deal with this we look to some examples and to describe the intended security policy.

We will first look at relation level privileges. We do this by allowing only each committee member to have a user in order to prevent unauthorized access to the data. We would restrict read privileges to committee members, restrict read and modification and reference privileges to the three senior committee positions: Champion, Information Officer and the Child Protection Officer. Read Privileges give an account the ability to use SELECT to retrieve rows from this relation. Modification Privileges give an account the ability to use INSERT, UPDATE and DELETE to modify rows in this relation. Reference Privileges give an account the ability to refer to this relation when specifying integrity constraints.

The Database administrator positions (DBA) would be taken on by the information officer and the champion. They would create users for each of the committee members -Champion, IO, CPO, PRO, treasurer and 5 OCMs.

GRANT SELECT on Mentors TO treasurer;

GRANT CREATE TABLE TO champion WITH GRANT OPTION;

From this the treasurer has the ability to access and filter the mentor table and champion has the ability to create a new table and to share that privilege with other users, the treasurer cannot share their privilege with other users.

If we wanted to take away a privilege, we would use the following code.

REVOKE SELECT on Mentors FROM treasurer;

Adopting this type of approach and allowing different users to have different privileges tailored to their job within the dojo will ensure greater security for the data stored in the database. The creation of views from tables further enhances the security on the information stored in the database as it allows the DBA (Database Administrator) to restrict the attributed that can be seen by the different staff members.

## examples