COMPX323 – Project

* Kevin Han 1521885
* Bedir Asici 1539000
* Tetsusaburo Kato

Table of Contents

Table of Contents 1

1. Project Milestone 1 2

1.1 Application Description 2

1.2 Revised ER Diagram 3

2. Project Milestone 2 4

2.1 Relation Schema 4

2.2 Table Definitions 4

2.3 Dataset 8

2.4 Application 11

3. Project Milestone 3 11

3.1 Indexing and Querying Optimization 13

4. COMPX323-22A Project Checklist 19

5. Code, SQL and Data 20

# Project Milestone 1

## Application Description

The application focuses on sports event information storage. It keeps track of which events happened at which location at which time. Users can use it as a watch list to as they will have a watchlist dedicated to their id, so that they can keep a track of all events that they have watched. If a person is in a team they have access to an additional page that shows matches that they have personally participated in as well as their current members. It does not keep track of wins and losses as that brings up potential storage and relational issues in which the win storage location will have to be decided e.g. attached to team or event.

The sport platform wants to have a new database to store information about sport events. The platform stores. Sport events can be hosted in-person, virtual or both. The platform wants to store information about the players, organizers, viewers, the team the players are in and the sport. The team only records the current roster and players can only be assigned to at most 1 team at any given time. Details about the Organization are stored in the organizers. The Sport Event is created when it is organized by an organizer.

Explanation of relationships:

Team - Plays - Sport: 1 team can play 1 and only 1 sport; A sport will have at least 1 team, to many teams that play it.

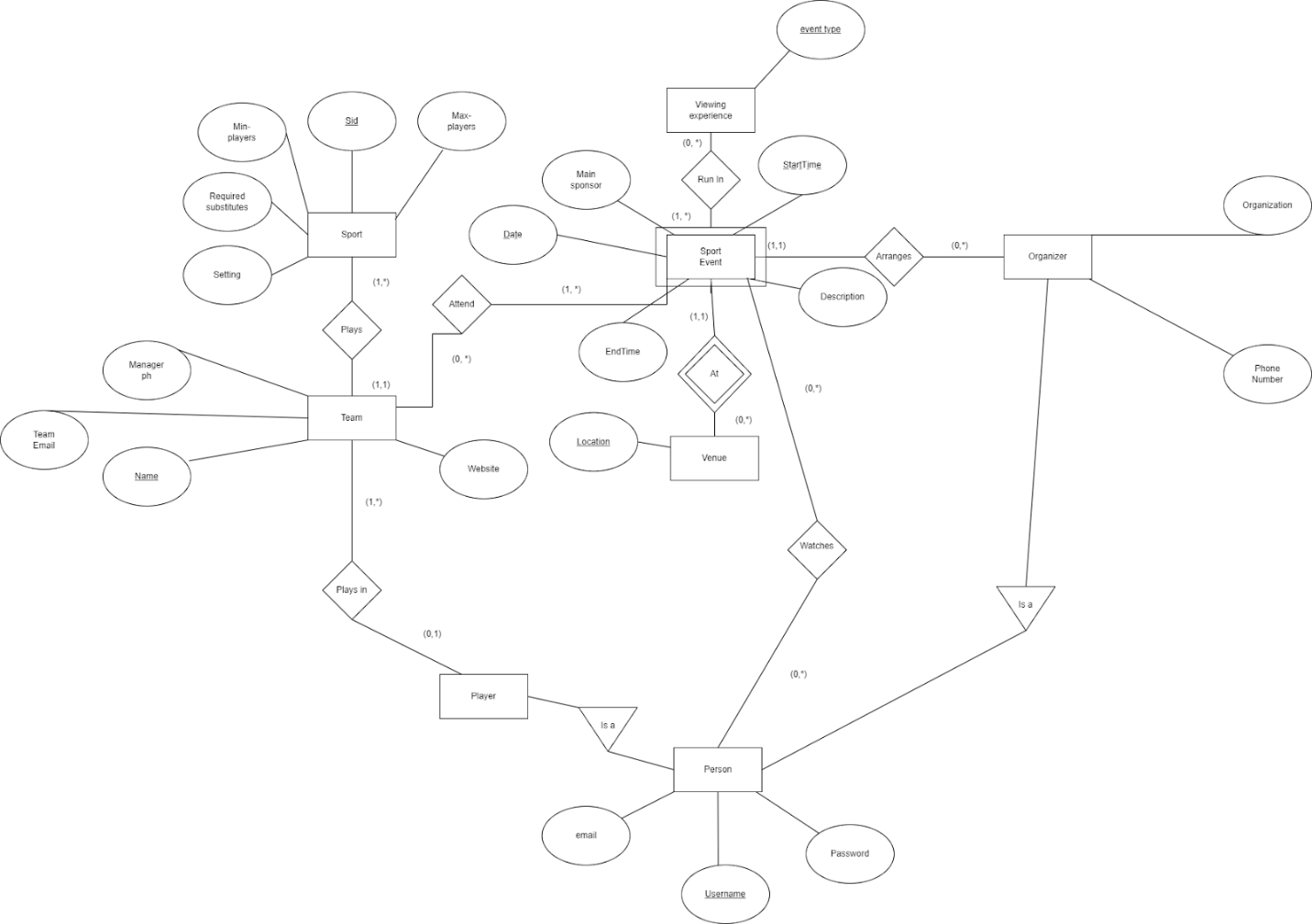
Sport - Sport - Event: A sport can take place at no events, to many events; A sport event can hold one to many sports

Person - Attends - Sport Event: 1 viewer can attend 0 events or many events; A sport event can have 0 to many members in attendance

Sport-event - arranges - Organizer: A sport event is organized by one organizer and only 1, An organizer can organize 0 to many events.

A sporting event is a generalization between virtual esports and other forms of sports that do not have to be played from a shared location.

## Revised ER Diagram



# Project Milestone 2

## Relation Schema

**Person**

Username, Email, Password, First Name, Last Name

**Player (Inherit from Person)**

Username, TeamID

**Organizer (Inherit from Person)**

Username, Organization, Phone Number

**Team**

TeamID, Name, TeamEmail, Phone Number, Website, Sport

**Sport**

Name, MinPlayer, MaxPlayer, Required Substitute, Setting

**Setting**

Setting

**Sport Event**

Date, Start Time, Location, End Time, Main Sponsor, Description, Organizer

**Venue**

Location

**Viewing experience**

Event-type

**Run In**

Date, Start Time, Organizer, Event-type

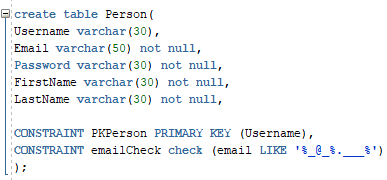
**Attends**

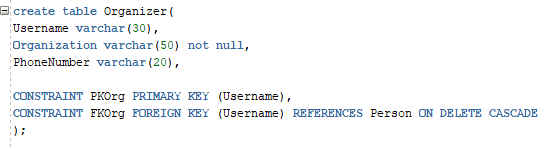
TeamID, Date, Start Time, Location

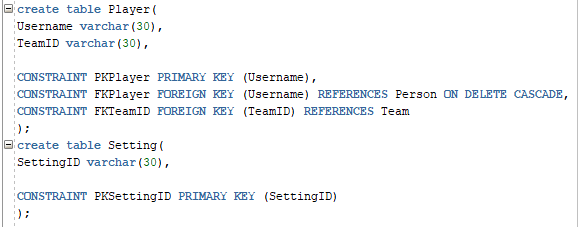
**Watches**

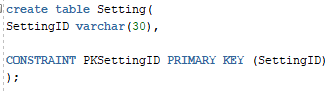
Person, Date, Start Time, Location

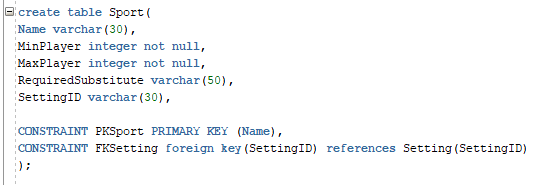
## Table Definitions





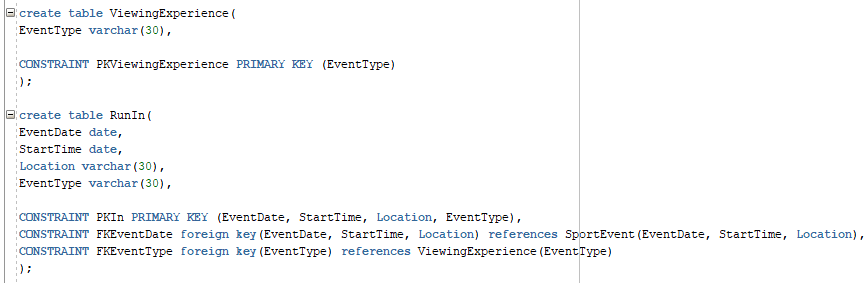


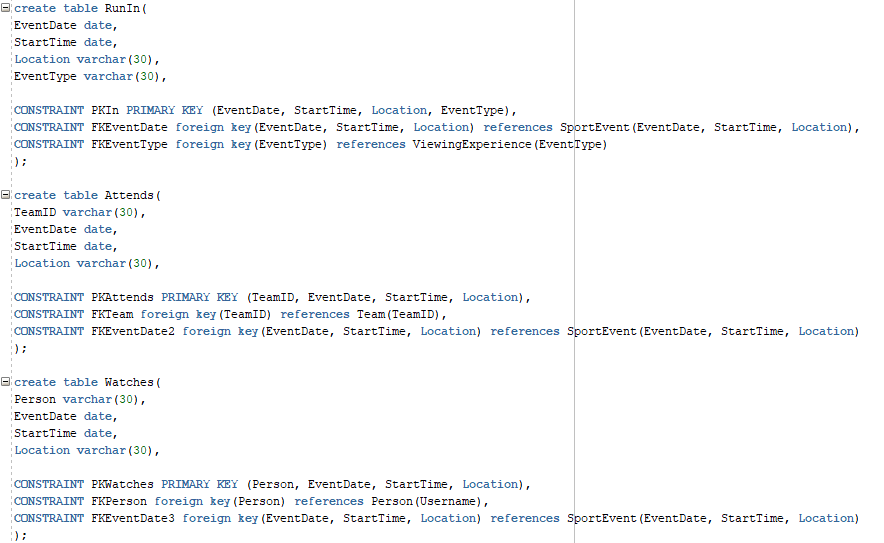






s

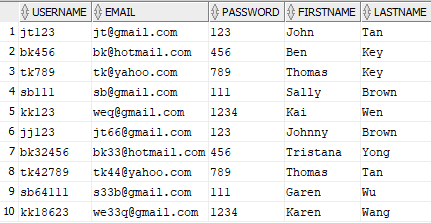




## Dataset

### Small

Person



Organizer

Text

Description automatically generated with low confidence

Player

Table

Description automatically generated

Setting

Graphical user interface, application

Description automatically generated

Sport



Team

Text, table

Description automatically generated

Venue

Graphical user interface, text

Description automatically generated

Table, Excel

Description automatically generatedSport Event

Viewing Experience

Graphical user interface, text, application

Description automatically generated

Run In

Table

Description automatically generated

Attends

Table

Description automatically generated

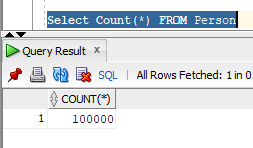
Watches

Table

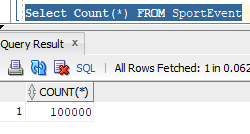
Description automatically generated

### Large

Person Table



Sport Event Tables

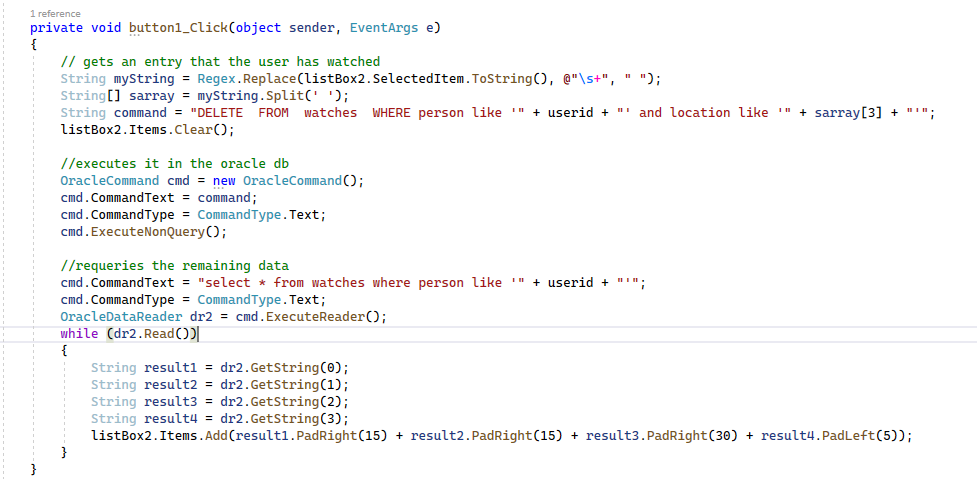


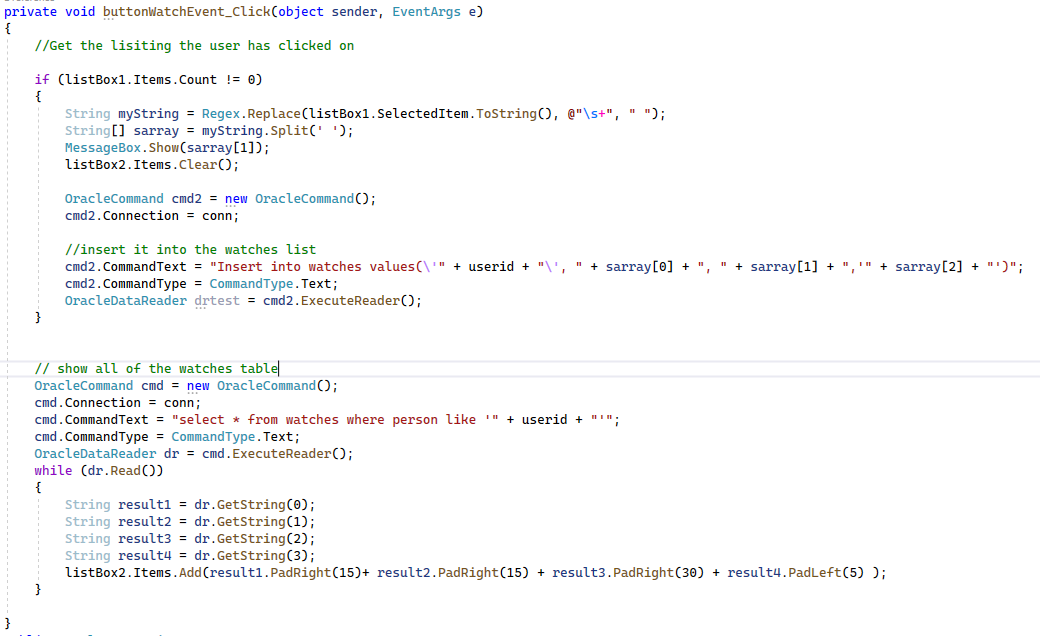
The screenshot above are the two main table.

The dataset is randomly generated using a java program generating random string of chars/integer.

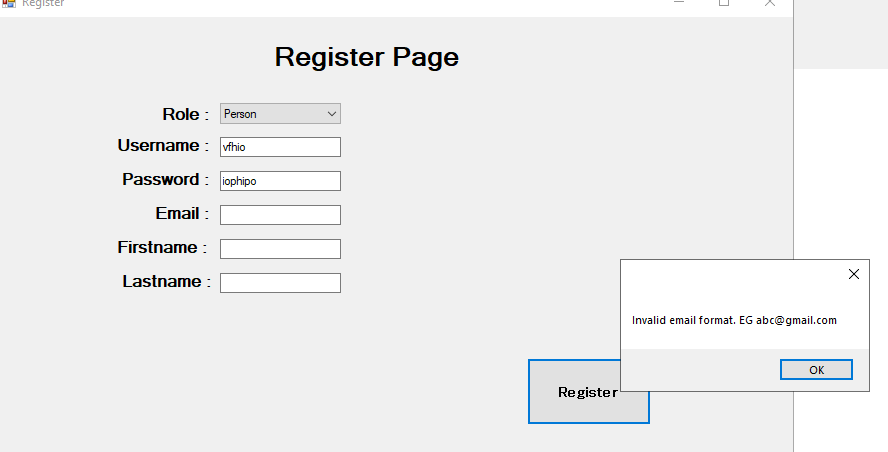
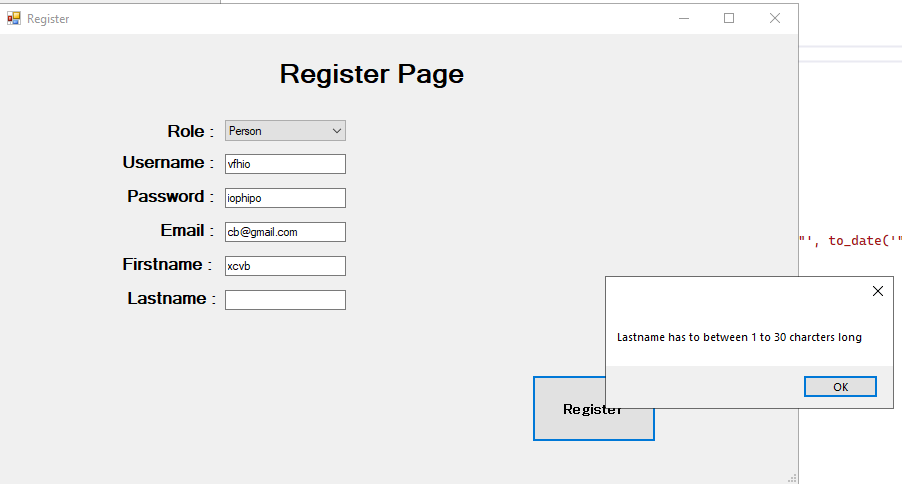
## Application

Showing that the data can be deleted, with user interaction.



User can insert data into the database

Error handling



# Project Milestone 3

## Indexing and Querying Optimization

### Text Description automatically generatedQueries and SQL script to create index

### Why was it chosen?

B-Tree:

We decided to choose first name on Person table. It allows us to search for different group of people with some similarities with their first name.

Hash-Cluster:

We chose organizer as the hash for the Sport Event table. We felt that there will be lot of queries trying to find the events that are organized by certain organizer. Using hash allow for quick direct to all event organized by the one organizer.

### Performance Measurement

B-Tree Index Before Implement

Text, table, email

Description automatically generated

Table

Description automatically generated

B-Tree Index After Implement

Table

Description automatically generated

Hash-Cluster Index Before Implement

Text, table

Description automatically generated

Hash-Cluster Index After Implement

Text, application, table, email

Description automatically generated

### Discussion of Performance Measurement

Using indexing it was demonstrated that the cost for finding the value that is being searched for is less costly. This is important as with large datasets such as user information storage, the more information you store the slower the retrieval is. As time delays are an important factor to consider therefore indexing becomes increasing important based on the dataset. Indexing allows for faster querying ability however it has downsides of being expensive to insert new data. Because of this it is good for applications in which user retention and activity are high but user intake is low. Comparing the specific indexes shows that there are benefits to each and that they have specific upsides to consider.

## Mongo db

### Structure



For our mongo db structure we chose to have person as the main outermost element, then within the person document we chose to list all the related elements the person had interacted with within the same document. This means that for the persons that had not watched an event or had participated in an event they will have had smaller documents. Because mongo db is a NoSQL structure there are no joins, storing everything within the single document style is beneficial for our application as sport events won’t exist if no players attended it. Therefore, we do not need additional tables to store sport event or other tables. Concludingly as the existence of teams, organizers, sport event is based on the existence of persons having all the information nested within is person is appropriate for our application.

We implemented functionality for the login phase of our application use using Mongo db. We did not implement the modification of the database using mongo dB, therefore it cannot be inserted into or deleted from.

Comparing to the storage method in our SQL the mongo dB will have larger storage size requirements because of duplicate data, however for the purpose of the common user (non admin) the queries will be much faster as they will be making queries with low traversal cost.

# COMPX323-22A Project Checklist

Kevin Han: 55 (Contributions %)

Bedir Asici: 35 (Contributions %)

Tetsusaburo Kato:10 (Contributions %)

|  |  |  |
| --- | --- | --- |
| Project Milestone 1 | |  |
| 1 | 1a. Clear structure of milestone material, including headings, sections, readable screenshots with captions. This checklist should be included and filled in. | ✔ |
| 2 | 1b. Database application description. | ✔ |
| 6 | 1c. Revised ER Diagram. | ✔ |
| Project Milestone 2 | |  |
| 4 | 2. Relational schema for your ER Diagram. | ✔ |
| 6 | 3. Table definitions in Oracle, include SQL script which creates relevant tables etc. | ✔ |
| 3 | 4a. Dataset: small (screenshots of dataset successfully loaded). | ✔ |
| 4 | 4b. Dataset: large.  Description of how data was created (incl code if relevant)  Screenshot of large dataset successfully loaded (use count). | ✔ |
|  | 5. Application:  Functionality to display and modify the database.  System should be error proof with appropriate user messages.  Screenshots showing functionality, with appropriate descriptions. | ✔ |
| Project Milestone 3 | |  |
|  | 6. Indexing and Query Optimization:  Show queries used and SQL script that creates the indexes.  Discussion of why these indexes were chosen to optimize the queries.  Performance measurements with and without indexes (with query plan).  Discussion of performance measurements. | ✔ |
|  | 7. Application: extend with MongoDB.  MongoDB version of database (show structure) + small dataset (screenshot).  Explanation of the data structures you have chosen and comparison to your SQL version.  Core functionality of application in second tab/area, using MongoDB (screenshots). |  |

# Code, SQL and Data

<https://github.com/Lecreator-KH/COMPX323App.git>