- COS20031 Team Space -

About

This Confluence homepage is the primary landing site for our COS20031 project. We are currently implementing high quality data-management solutions for our range of customers.

Mission and vision

Our team mission is to ensure high quality data management solutions are delivered to all our clients and projects!

Meet the team

Melvin Goxhaj

Backend Developer

In charge of ensuring robust systems are in place for processing, cleaning and managing database and information realtional systems.

Nikhil Mohite

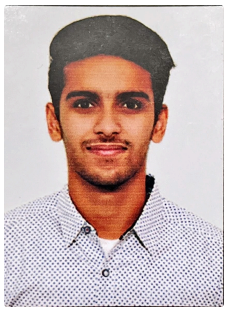
Front end/UX Lead

Creating and maintaining user interfaces for clients to interact with the webpage.

Arsh Khanna

Testing Lead

Must ensure code is unit tested, functional and without bugs



Pulkit Pannu

Team Organiser/Coordinator

Responsible for organising group meetings and making sure all the tasks are done on time

Upek Isuranga

System Administrator

Make sure the system runs and operates smoothly

Contact us

How can someone reach out to your team?

- 104549772@student.swin.edu.au
- 104540526@student.swin.edu.au
- 104100124@student.swin.edu.au
- 104093910@student.swin.edu.au
- 104532824@student.swin.edu.au

Important Pages

- Project Plan
- Working Agreement
- Entity Diagram

Roles and Responsibilities

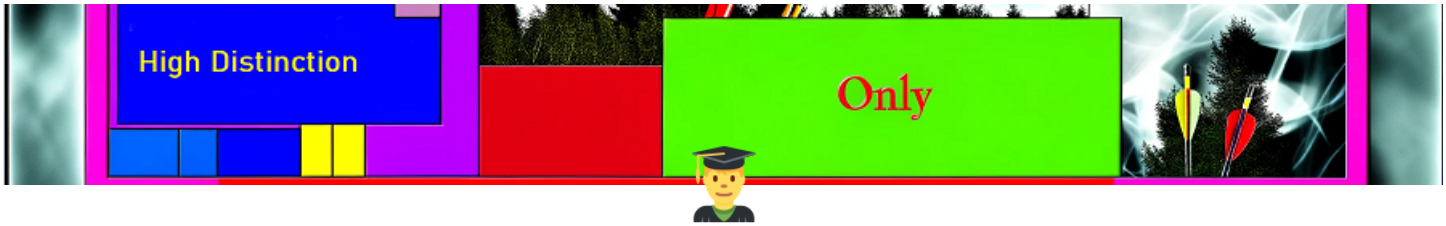
🎓 Roles and Responsibilities

Meeting notes

📖 Group Meeting Notes

Team home page

👩‍🎓 High Distinction Only



High Distinction Only

👋 Welcome to the High Distinction Only homepage!

📈 Team metrics

Database Uptime “Five 9’s“#1
Updated May 1, 2024
ON TRACK

Archery Club Database Access #2
Updated May 8, 2024
ON TRACK

✨ About High Distinction Only

We are a collective of Swinburne Students aiming to implement a high quality and secure SQL databse for a local archery club.

📸 Meet the team

Arsh Khanna
Testing Head

Nikhil Mohite
Frontend/UX Lead

Melvin Goxhaj
Backend/Database Engineer

Pulkit Pannu
Team Organiser + Misc

🗓️ Meetings

Date	Attendees	Agenda	Notes, decisions and action items
Feb 28, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Team Building and Team Agreement	• First meeting of the group. • Introductions were made and contacts exchanged for team collaboration and follow-ups. • Confluence was set up and a Team Agreement was made with mutual consent.
Mar 6, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Linking Jira to Confluence	• Week 2 meeting. • Confluence was linked to Jira by @Pulkit • Issues were made with the future of the project kept in mind. • A thorough discussion of the project brief was done so that we all had a good understanding of the project requirements and could clarify any doubts with our tutor • Started working on other confluence pages. • The Project plan was made with a group discussion. • Roles and responsibilities were divided although everyone is expected to work together and help out a team member if they are having problems. • Risk Assessment Matrix was also made.
Mar 13, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Entity Modelling	• Week 3 meeting. • Confluence pages made in the previous meeting were completed and polished according to the feedback provided by our tutor. • The project brief was read again for a better understanding. • Had an introduction to the draw.io software. • Started on the Entity Relationship diagram for our database.
Mar 20, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Personas, Empathy Mapping and Product Requirements	• Week 4 meeting. • Finished up on the ER diagram and got feedback on it from our tutor. • Discussed as a group about the various personas suitable for our product. • Personas was created in relation to the entity diagram and our project situation. • Empathy Maps was created for the previously created personas. • Product Requirements were verified.
Mar 27, 2024	@MelvinGoxhaj @Upek Isuranga @Pulkit @ARSH KHANNA	Progress Check and Resolving Issues	• Week 5 meeting • Checking many-to-many relationship • Investigation for database's nominal form • Updating Jira
Apr 10, 2024	@MelvinGoxhaj @Nikhil mohite @Upek Isuranga @Pulkit	Create Database And Tables	• Week 6 meeting • Creation of Database by @MelvinGoxhaj • Creating tables in the database
Apr 17, 2024	@ARSH KHANNA @Pulkit @Nikhil mohite @Upek Isuranga @MelvinGoxhaj	Define nullable values and Enter Dummy data	• week 7 meeting • Defining nullable values in the database • Creation of dummy data • Decide on the way to upload the data. • Uploading the data to the table.
May 6, 2024	@Pulkit @MelvinGoxhaj @ARSH KHANNA	Populate the table with data for the unit tests to successfully run. Finalize group account and personal account tables.	• Tables are complete, some group members need to adjust minor tweaks such as PK and FK combinations • Unit test command have been created & run against tables, some are not currently passing due to minor schematic differences.
May 15, 2024	@Pulkit @ARSH KHANNA	Explore topics related to majors and data management	• Tasks for different majors were explored in simple forms, focusing on the relevance to data management. • Games and Software Development tasks focused on implementing the main use case of

			the project.
May 22, 2024	@Pulkit @Upek Isuranga @ARSH KHANNA @MelvinGoxhaj @Nikhil mohite	Finalize the product deliverables and finish the report.	• Front-end is working and enables user input for archer's database. • Demonstration video has been recorded outlining the project. • Confluence report has been updated and finalized for submitting. • Self and peer assessments have been completed and submitted.

Resources

Search

Search

Where to find us

You can contact the team anytime by following the links below:

Discord - A New Way to Chat with Friends & Communities

104540526@student.swin.edu.au

Our Group Working Agreement

👥 Team Preferences

Team Member	Arsh	Melvin	Nikhil	Pulkit	Upek	**Final Agreed Upon Terms**
Working location and timezone	Melbourne, AEDT	Melbourne, AEDT	Melbourne, AEDT	Melbourne, AEDT	Melbourne, AEDT	**Melbourne**
Working hours and commitments	9am - 8pm	9am to 9pm	Free 24/7	8am -7pm	10 am - 7 pm	**12 - 7pm**
Working environment and preferences	From Uni	I work from home and sometimes the latelab	Meetings face to face and completing work from home.	From Uni or Online	From Uni or Online work	**From Uni**
How I like receiving feedback	Direct Feedback is preferred	Direct feedback face to face is my preferred method.	After assignments & submissions	Face to Face	Direct Feedback in person is preferred.	**Direct Feedback**
Context about me	I am an international student and I have a particular interest in cloud based cybersecurity services.	I live in Shepparton. So im always 3 hours at the quickest from Swinburne	Live down in waterways, majoring in AI.	I'm interested in the fields of Cyber and AI.	I have moved from New Zealand and I have interests in programming and game creation.	**Five dudes**

💬 Communication Channels

Channel	Purpose	Audience	Standards
Discord Server	Day to day work related conversating and planning	Core team	Completely open and informal, no strict protocol of communication to adhere too.
Instagram	Reserved for reaching those who do not respond to discord group messages after a threshold of time (>7 days)	Core team only still	Must be respectful and communicate lack of work/interaction with fellow team members.
Email	Used as last resort for formal communication of either a) Removal from group due to extended period of time with no communication (>1 month) b) Message from tutor for important change.	Core + Tutor	Formal in nature

🗓️ Meetings

Meeting Number	1 (February 28, 2024)	2 (March 6, 2024)	3 (March 13, 2024)	4 (March 20, 2024)	5 (March 27, 2024)	6 (April 10, 2024)	7 (April 17, 2024)	8 (May 1, 2024)	9 (May 8, 2024)	10 (May 15, 2024)	11 (May 22, 2024)
Objective	Team Building and Team Agreement	Finishing up on Confluence pages and setting up Jira	Planning & Wrapping up Project Proposal	Completing the project proposal and getting feedback	Resolving issues and update Jira and existing backlog	Updating the ER diagram and creating the database	Creating the database and inserting dummy data	Starting on developing the front-end of the website	Progress review and planning for final stages	Explore different majors in simple forms and focusing on the relevance to data management.	Final project review and preparation for submission
Outcomes	Team was	Getting a filled	Getting a	Finished the	Discussed how the	Some progres	Properly defined	Initiated front-	Reviewed progress	Collaborated with	Conducted final review

	built and started on the working agreement	out and completed	completed project proposal by the start of week 3	project proposal after getting feedback	ER diagram can be improved and updated Jira	s made regarding changes to be made to the ER diagram	code written with correct data types and nullable values identified. Ran into some errors while running the script. Couldn't perform data insertion.	end development tasks and assigned responsibilities for UI design and implementation.	on front-end development. Team also discussed remaining tasks and set deadlines for completion.	members from various majors leveraging their expertise to accomplish tasks related to AI, Cybersecurity, Games to develop our front-end.	of the product and addressed any last-minute issues and prepared documentation for submission. Video demonstration was also created.
Format	In-person	In-person	In person, as always.	In-person	In-person	In-person	In-person	In-person	In-person	In-person	In-person and discord
Who	Melvin, Nikhil, Pulkit, Arsh	Melvin, Nikhil, Pulkit (Arsh was sick)	Melvin, Nikhil, Pulkit, Arsh, Upek(joined the group)	Melvin, Nikhil, Arsh, Upek (Pulkit couldn't join offline due to a medical condition but joined via discord call)	Melvin, Nikhil, Arsh, Pulkit, Upek	Melvin, Nikhil, Arsh, Pulkit, Upek	Melvin, Nikhil, Arsh, Pulkit, Upek	Melvin, Nikhil, Arsh, Pulkit, Upek	Melvin, Nikhil, Arsh, Pulkit, Upek	Melvin, Nikhil, Arsh, Pulkit, Upek	Melvin, Nikhil, Arsh, Pulkit, Upek
Resources	Confluence	Confluence, Jira	Jira & Confluence	Jira & Confluence	Jira & Confluence	Jira & Confluence	Jira & Confluence	Jira & Confluence	Jira & Confluence	Jira & Confluence	Jira & Confluence
How will we show up?	On campus	On campus in class	On campus during works	On campus during worksh op and	On campus during workshop and in	On campus during worksho p and in	On campus during workshop and in	On campus during worksh op and	On campus during workshop and in	On campus during workshop and in	On campus during workshop and online.

			hop and in latelab after	in latelab after	latelab after	latelab after	latelab after	in latelab after	latelab after	latelab after	
How will we manage follow up?	Discord	Discord	Discord Server for group	Discord Server for group	Discord Server for group	Discord Server for group	Discord Server for group	Discord Server for group	Discord Server for group	Discord Server for group	Discord Server for group

⬆️ Escalation Process

Decider	How	Transparency	Feedback Loop
Pulkit/Arsh/Melvin/Nikhil. Whoever initates the conflict procedure is responible for seeing its resolution out.	DACI via Confluence.	The entire team will be informed of proceedings via discord server admin notification.	All team members will be kept in the loop with every update.
Mutual team decisions on project development and designs and feedback will be taken by a detailed group discussion focusing of the problem in question.	In-person/Discord team call	All the meetings will preferably be in the presence of all the team members but if someone is unable to attend in-person or online, a detailed description of the meeting will be provided to the team member via Discord and the meeting notes will also be available.	All decisions made in regards of the project will be mutual and agreed upon by every team member. Moreover, we ask for feedback from our tutor before moving on to the next step of our project.

💡 Continuous Improvement

Purpose	How	Standards
Sharing feedback	Face to face	• Positive criticism is encouraged.
Double checking / Testing	In-person/Online	• Providing valuable second opinions and error preventions via detailed and precise testing
Documentation Review	Shared Online Workspace	• Regularly updating and reviewing project documentation for clarity,

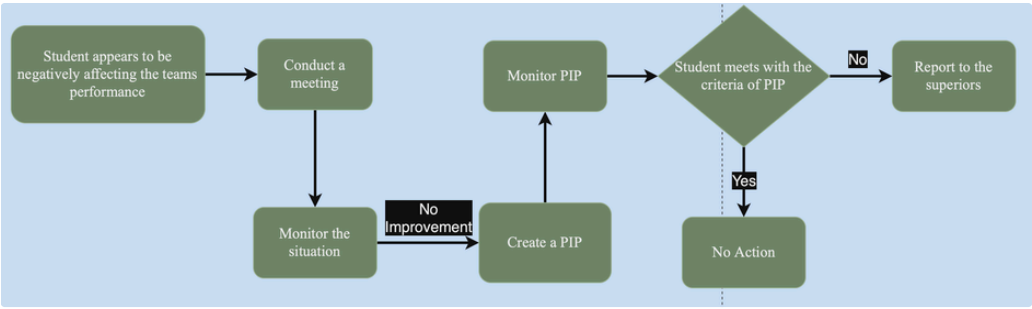
		accuracy, and completeness to ensure alignment with project objectives.
Reflection Sessions	End of Sprint meetings	• Holding reflective discussions to review what worked well and what could be improved, fostering a culture of learning and adaptation

👽Underperformance Provision

Provision for the case if a team member contributes poorly to the work to be done as a group.

Team member behaviour	Action taken
Communicates less but makes efforts to contribute to the team but eventually leads to low efficiency in work done.	• Conduct a meeting with the team member and politely explain the situation and offer to help and explain any queries or concerns.
Failed to attend more than one of the team meetings without making effort to inform before hand or to catch up with the work done.	• Take note of the events. • Ask politely about the reasons for not showing up and report to the tutor the said reason and the situation. • Offer to help and create a *performance improvement plan*. • Monitor the behaviour.
Failed to attend most of the group meetings, didn't make any efforts to communicate afterwards and failed to contribute to most of the group work.	• Conduct a meeting with the team member along with the supervising tutor and inform the details to the unit convenor. • Explain in detail about the course of events and the lack of communication and underperformance of the team member. • Offer to help and create a *performance improvement plan*. • Monitor the behaviour.
All of the above and still no improvement in the behaviour of the individual.	• Inform the tutor and the unit convenor and ask for the immediate expulsion of the team member and a special consideration for the work done by the group.

A GENERAL DIAGRAM REPRESENTING STEP OF ACTIONS FOR UNDERPERFORMANCE


Student appears to be negatively affecting the teams performance
Conduct a meeting
Monitor the situation
No Improvement
Create a PIP
Monitor PIP
Student meets with the criteria of PIP
No
Report to the superiors
Yes
No Action

😎

Project Plan for COS20031 Team

Driver	Arsh Khanna
Approver	Pulkit Pannu/Melvin Goxhaj
Contributors	Arsh, Melvin, Nikhil, Pulkit, Upek
Informed	@ARSH KHANNA @Nikhil mohite @MelvinGoxhaj @Pulkit @Upek Isuranga
Objective	Implement efficient and high-quality data management solutions using MySQL scripting language for a local archery club, ensuring archers can see their scores and club recorders to ensure the scores are inserted correctly.
Due date	May 13, 2024
Key outcomes	1. Archery Club Satisfaction Feedback 2. Database integrity metrics 3. Database incident levels metric/downtime 4. Database query times.
Status	COMPLETED

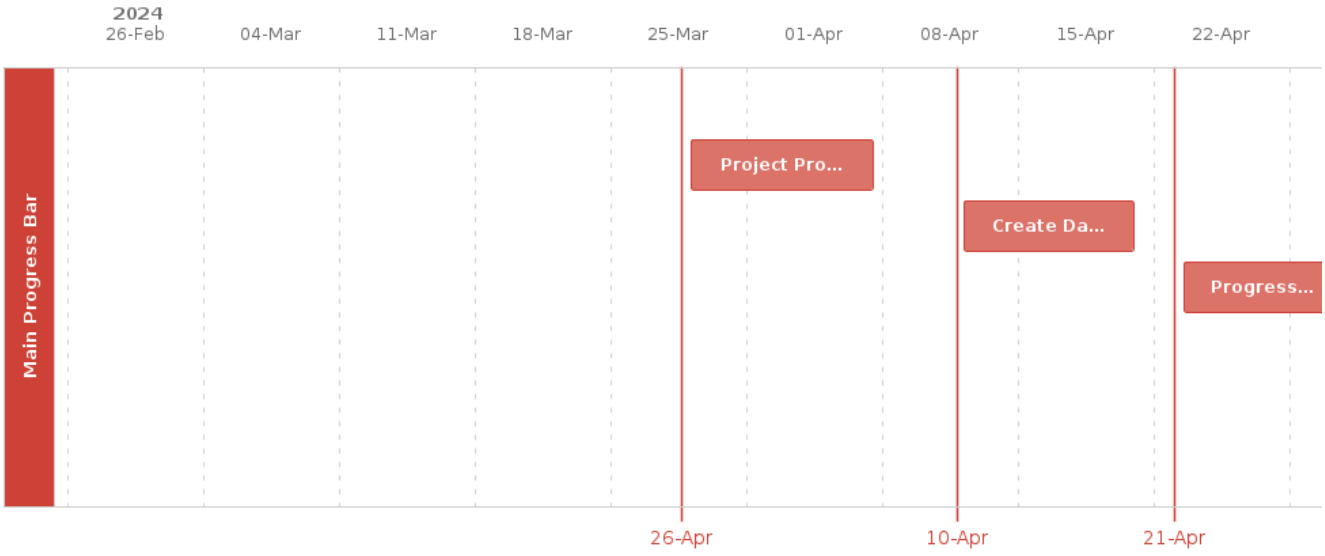
🤔 Problem Statement

Our problem is storing data for a local archery club in a realational manner using RDMS, making inserts, reads and deletes quick & easy and by using the correct software for the job whilst coordinating with other team members and stake holders to ensure the database is delivered on time and as promised with a working front-end so that the archery club can easily access their data and a working backend to necessary changes can be made by admins too.

🎯 Scope of Project

Must have:	• Must be able to store data in a realtional manner • Must maintain data integrity and security • Robust user authentication and authorization for data access.
Nice to have:	• As fast as possible read, writes & deletes. • A nice web client GUI for the database • Regular backup and disaster recovery capabilities. • Mobile-responsive design for the web client GUI.
Not in scope:	• Multiple databases in different scripting languages such as NoSQL etc. • Development of a custom database engine. • Support for legacy systems and outdated protocols.

🗓️ Timeline for Implementation



🚩 Team deadlines (Group tasks only)

Milestone	Owner	Deadline	Status
Project Proposal (20%)	@MelvinGoxhaj @Pulkit @Nikhil mohite @ARSH KHANNA @Upek Isuranga	Mar 26, 2024	COMPLETED
Progress Report (5%)	@MelvinGoxhaj @Pulkit @Nikhil mohite @ARSH KHANNA @Upek Isuranga	Apr 19, 2024	COMPLETED
Final Report and Product Deliverable (25%)	@MelvinGoxhaj @ARSH KHANNA @Nikhil mohite @Pulkit	May 26, 2024	COMPLETED

	@Upek Isuranga		

Reference materials

- https://swinburne.instructure.com/courses/56946/assignments/587183 Project Proposal)
- https://swinburne.instructure.com/courses/56946/assignments/587181 Progress Report
- https://swinburne.instructure.com/courses/56946/assignments/587176)

🎓 Roles and Responsibilities

📋 Overview

Identify and discuss team responsibilities by following the instructions for the Roles and Responsibilities Play.

Team	High Distinction Only
Team members	Arsh, Melvin, Nikhil, Pulkit, Upek
Date	Mar 4, 2024
Team mission	Design and implement a functional MySQL database in order to organize the realtional data needs of a local archery club, whilst ensuring we maintain the highest possible level of data integrity and security.

📘 Roles and responsibilities

Roles	Responsibilities (what others think)	Responsibilities (what I think)	Comments
Team Organiser/Coordinator	• Setting meeting reminders for the group • Notify if there are any conflicts.	• Updating the 'meetings' time table within 'Group working agreement' page.	Pulkit has been assigned with the responbility.
Backend Developer (Database)	• Ensuring SQL queries are optimsed and running performantly • Making sure the archery club database is secure and without leaks	• Database has to have low downtime, high level of security, follow ACID proprties and best practices • Must be integrated well into the front-end and user accesible	Melvin has been assigned to this role.
Front end/UX Lead	• Creating and maintaining user interfaces for clients to interact with the webpage. • Voice any changes in backend to frontend developer	• Creating clean and efficient HTML and CSS code • Design a webpage visitors that is easy for visitors to navigate around.	Nikhil has been assigned this role
Testing	• Must ensure code is unit tested, functional and without bugs • Should make sure that the front end and back end are both secure, functional and	• The product should be throghly tested without any bugs and when delivered to end user (archery club) fully functional.	Arsh has been assigned to this role

	without any possible user misfunctions.	• All code should be unit tested as well, following OOP best practices and standards	
System Administrator	• Manage server configurations and stability • Troubleshoot technical issues and provide support to users	• Make sure the system runs and operates smoothly. • Implementing and enforce security policies to protect sensitive data and prevent data breaches	Upek has been assigned to this role.

Unassigned responsibilities
• Graphics/Logo for the team
• Coordinating team meetings is left to the collective group to agree upon weekly times when everyone is free without work, uni & or other prior commitment
• Caching/Speed/Optimization for the database

Risk Assessment Matrix

📋 Background

Some of the risks involved in our project are related to data security, integrity and containment. All archery club data should be isolated and unable to be read, inserted or deleted by any unauthorised user entry, the databse should also have correct user and admin authentication levels to ensure that happens so.

🐓 Risks management

Below we have analysed and pondered what the least to most severe potential risks could be to the project as whole, the database alone and or the site as welll. We've rated them on a scale from Low to Extreme and elaborated on each point as to why they would lead to risks arising.

Risk ratings table

LOW	MEDIUM	HIGH	EXTREME
• Database is not visible to ALL archery club users and therefore not *fully* functional	• Database has weak password/username combination, suscetpible to bruteforce attacks • Some parts of the database are visible to read but not write , to users without admin access. Not a direct threat security threat but poor practice	• Major database downtime and or conflicting databse results such as queries resulting in multiple results • Admin account comprised leading to any user being able to access the databse internals • Front end non functional for archery club users i.e javascript/css non responsive	• Read/writes/inserts/deletes are completly avaible to any user. • Admin login details complelty exposed.

Below we have outlined the severity of some potential risks in our database implementation.

	SEVERITY			
LIKELIHOOD	**ACCEPTABLE** *Little to no effect on event*	**TOLERABLE** *Effects are felt, but not critical to outcome*	**UNDESIRABLE** *Serious impact to the course of action and outcome*	**INTOLERABLE** *Could result in disaster*
IMPROBABLE *Risk is unlikely to occur*		System outage due to hardware failure.	Data loss due to lack of regular backups	Database is attacked by sophiscated attackers using zero-day vulnerabilites, leading to irreversible compromisation.
POSSIBLE *Risk will likely occur*		Database injection attacks due to insufficient input validation.	Database not fully functional for all users.	Major database downtime or conflicting database results, and admin account compromised.
PROBABLE *Risk will occur*			Weak password/username combinations and parts of the database visible to read but not write for non-admin users.	Database operations fully available to any user and admin login details exposed

✅ Action items

- [ ] Secure the availability of the database from outsiders by creating a better authentication and authorisation policies @Upek Isuranga @Nikhil mohite
- [ ] Validate input values into the database to prevent from SQL injection attacks @MelvinGoxhaj @ARSH KHANNA
- [ ] Ensure availability of redundant resources in case major database downtime @Pulkit @MelvinGoxhaj

🕵️ Persona 1

Persona name	**Jack Churchill**
Persona role	Archer
Job description	Jack is a competitive archer who frequently competes locally and regionally. His goal is for him to become more proficient and set personal records in every competition.

🏢 Company

Company name	The Archer Co.
Company size	2,000 Employees
Industry	Sports

👥 Demographic information

Age	30
Gender	Male
Income	$60,000

Education level	Bachelor Of Physical Education
Residential environment	Rural

🖋️ Personal quote

“Ain't bout the size of the dog in a fight, its bout the size of the fight in a dog!“

📜 Biography

Born into a small town with a long history of archery, Seaspray, Jack Churchill was enthralled with the grace and accuracy of the sport at an early age. After years of practice, he was a natural fit for competitive archery. Now, he looks forward to his hard focus periods at the range, where he aims to get better with every shot.

Professional goals	**Motivators**
• Skill Mastery • Personal Beats • Recognition • Tournament Wins	• Thrill Of Competition • Personal Growth • Howard Hill (A famous archer)
Challenges	**Sources of information**
• Equipment Cost • Balancing Training And Life	• Archery Associations • Online Forums

🕵️ Persona 2

Persona name	John Thompson
Persona role	Score Recorder
Job description	John is the designated score recorder of the Archery Club and uses the archery club on a day to day basis. He puts in the scores for all the athletes from their staging table into the official database table and checks in all the competition scores.

🏛️ Company

Company name	The Archer Co.
Company size	2,000 Employees
Industry	Sports

👥 Demographic information

Age	45
Gender	Male
Income	$150,000
Education level	Master in Business Administration
Residential environment	Urban

🖋️ Personal quote

“It’s gonna be Legend-wait for it-dary.”

📜 Biography

John Thompson, a countryside native, stumbled upon archery during his outdoor adventures and quickly fell in love with its precision and sense of community. Becoming a regular at the local archery club, he found his niche as the designated score recorder, where he diligently inputs competition scores into the database with care and accuracy. Outside the club, John works as a data analyst, but his true joy lies in the simplicity of shooting arrows and being part of the archery community.

Professional goals	**Motivators**
• Excel as a score recorder • Streamline database efficiency • Contribute to the archery club	• Passion for archery • Sharing Knowledge
Challenges	**Sources of information**
• Accuracy and human error • Time Management • Huge data sets	• National Archery Association • Networking with archers and officials

🕵️ Persona 3

Persona name	Selena Smith
Persona role	Archery enthusiast
Job description	Selena is a primary school teacher who is also an archery enthusiast and regularly goes to watch archery competitions held in her local archery club. She follows all the competitions regularly and tries her best to stay up to date with any new events.

🏛️ Company

Company name	Hypothetical Primary School
Company size	150 Employees
Industry	Education

👥 Demographic information

Age	28
Gender	Female
Income	$70,000
Education level	Graduate Certificate Of Education Research

Residential environment	Metropolitan

🖋️ Personal quote

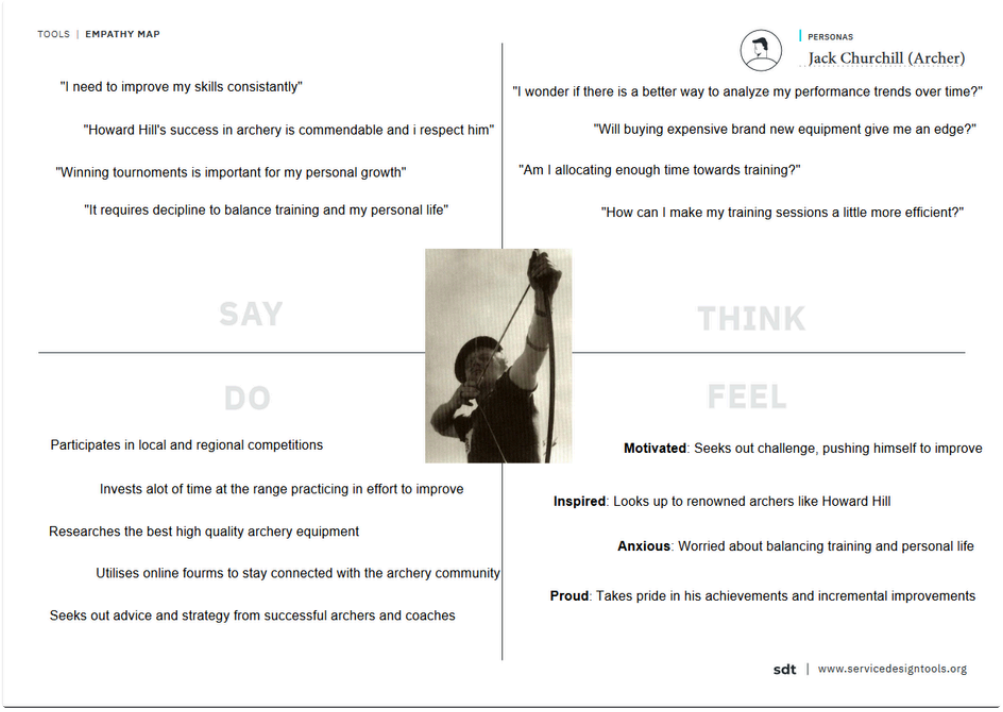
“A lie is just a great story that someone ruined with the truth.”

📜 Biography

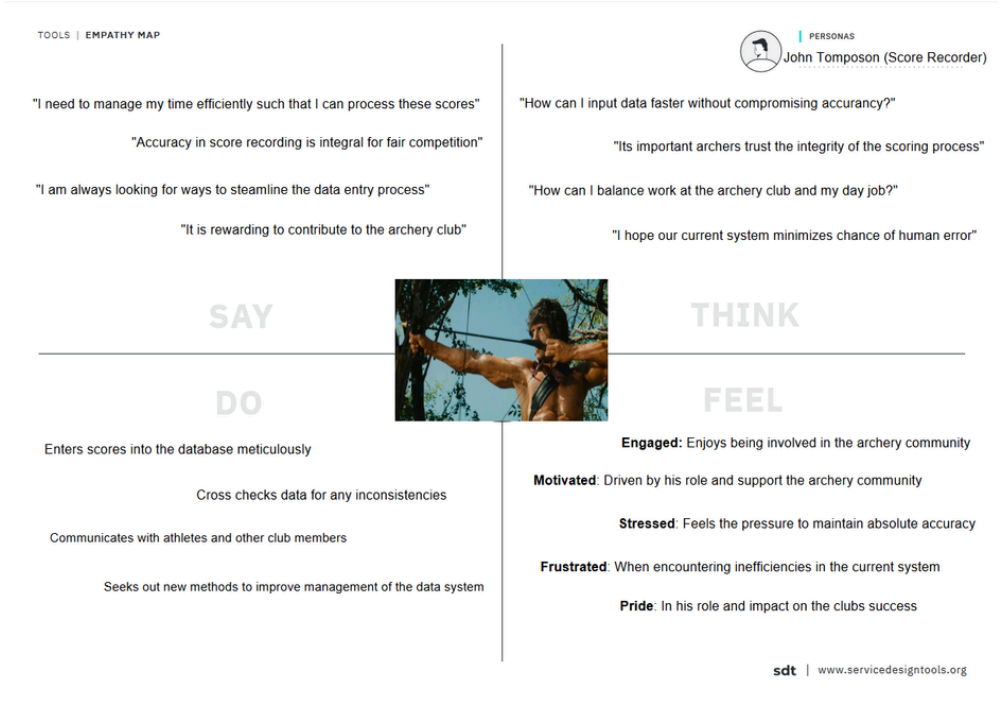
Born and raised in the centre of Melbourne City, Selena Smith developed an intense love for sports and accuracy as a child. She discovered her passion for archery at a young age. Despite her busy schedule, she regularly attends local competitions at the archery club, staying updated on new events. Selena incorporates archery's discipline and focus into her teaching, sharing its values with her students. Outside of work, she enjoys the tranquility of nature, finding joy in the simplicity of drawing her bow.

Professional goals	**Motivators**
• Provide good mentorship to her students • Make a positive impact on the lives of her students • Live her life to the fullest	• Her altruistic nature • General desire to help people and make a positive change
Challenges	**Sources of information**
• Work - Life balance • Day to day blues	• Online forums • News and Google • Social media

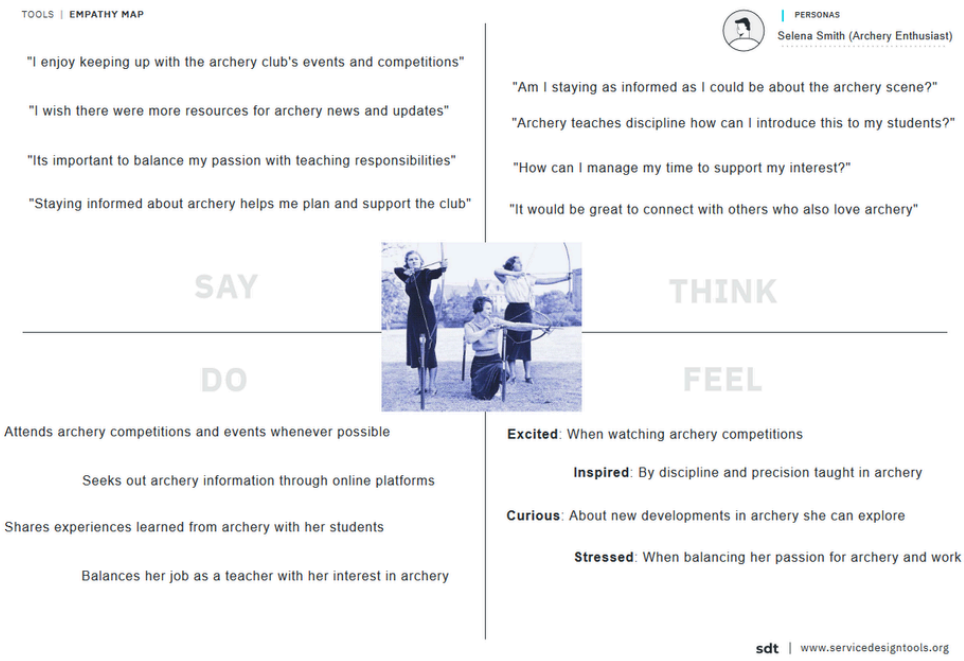
❤️ Empathy Maps



Jack Churchill (Archer)

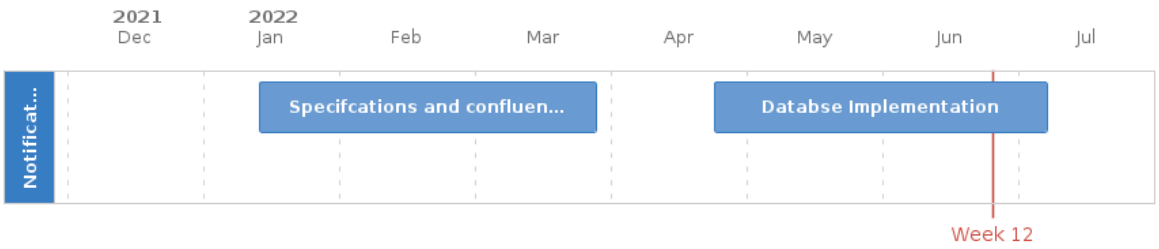


John Tompson (Score Recorder)



Selena Smith (Archery Enthusiast)

Product Requirements - HD Only



Target release	May 20, 2024
Epic	https://student-104093910.atlassian.net/browse/ACDHO-7 Can't find link
Document status	completed
Document owner	@Pulkit @Upek Isuranga @MelvinGoxhaj @ARSH KHANNA @Nikhil mohite
Designer	@MelvinGoxhaj @Pulkit
Tech lead	@ARSH KHANNA @MelvinGoxhaj
Technical writers	@Nikhil mohite @Upek Isuranga
QA	

🎯 Objective

The objective of this project is to learn how to plan, design and create a relational database and apply design thinking approach to understand and solve a stakeholder problem in a professional manner. Moreover, learning to design and implement a project plan using industry standard collaboration and management tools, and applying ethical and technical considerations in the development of the project solution are also the major key takeaways of this project.

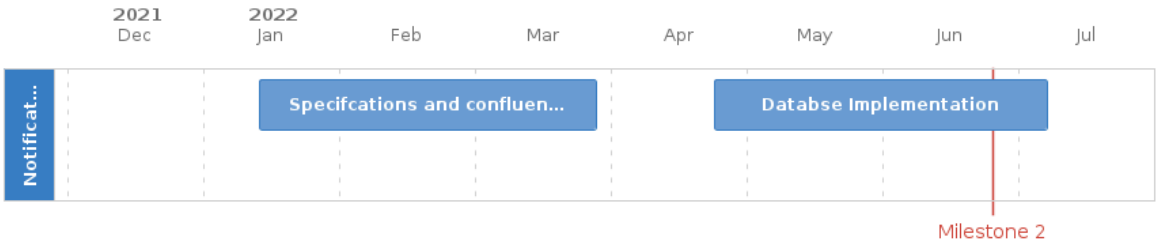
📊 Success metrics

Goal	Metric
Simplify user experience	Easier for the archers to access their scores and details about other competitions
Faster reads, writes and deletes	Quick access to the details under high traffic flow, i.e. during a busy competition etc
User authentication and authorisation	Secure access for clients and users

🤔 Assumptions

- The archers will have access to handheld devices or mobile apps to enter their scores.
- The club recorder will have administrative privileges to manage archer records, rounds, and competitions.
- Archery Australia's rules and round definitions may change over time, requiring updates to the database.

🌟 Milestones

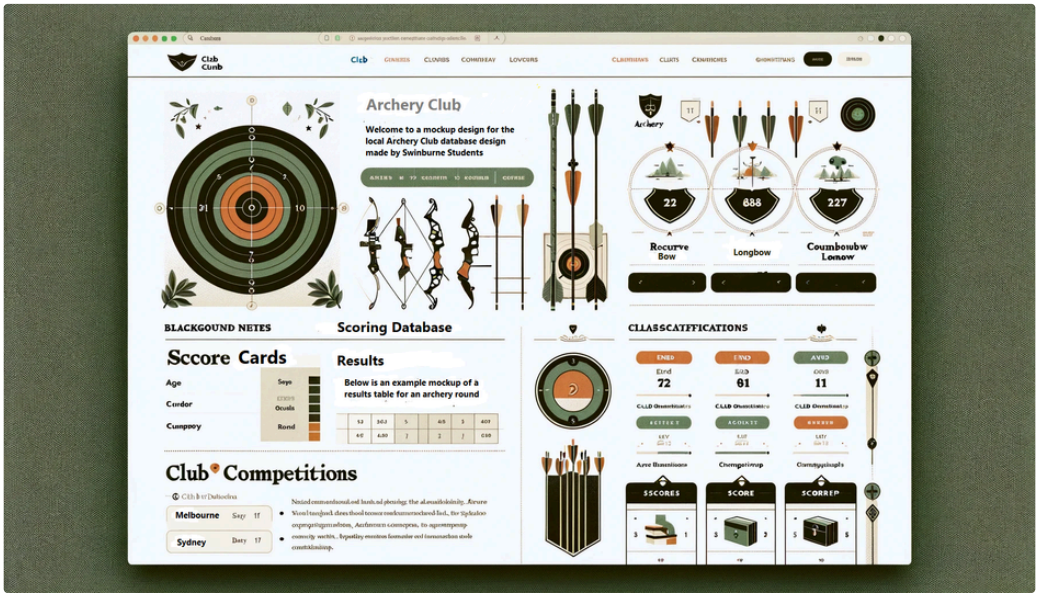


🗒️ Requirements

Requirement	User Story	Importance	Jira Issue	Notes
Archers can view their scores	As an archer, I want to view my past scores so that I can track my progress.	HIGH	ARC-101	Scores should be filterable by date range and round type.
Archers can enter scores	As an archer, I want to enter my scores from a handheld device so that I can easily record my performance.	MEDIUM	ARC-102	Scores should include date, time, round, and equipment details.
Club recorders can manage archer records	As a club recorder, I want to manage archer records so that I can keep track of archer details and divisions.	MEDIUM	ARC-103	Archer records should include personal information, age, gender, and default equipment.
Club recorders can manage rounds	As a club recorder, I want to manage rounds so that I can define new rounds or update existing ones.	MEDIUM	ARC-104	Rounds should include definitions

				for distances, number of ends, target faces, and total arrows.
Club recorders can manage club championships	As a club recorder, I want to manage club championships so that I can define participating rounds and calculate overall standings.	HIGH	ARC-205	Club championships should include participating rounds and scoring rules.

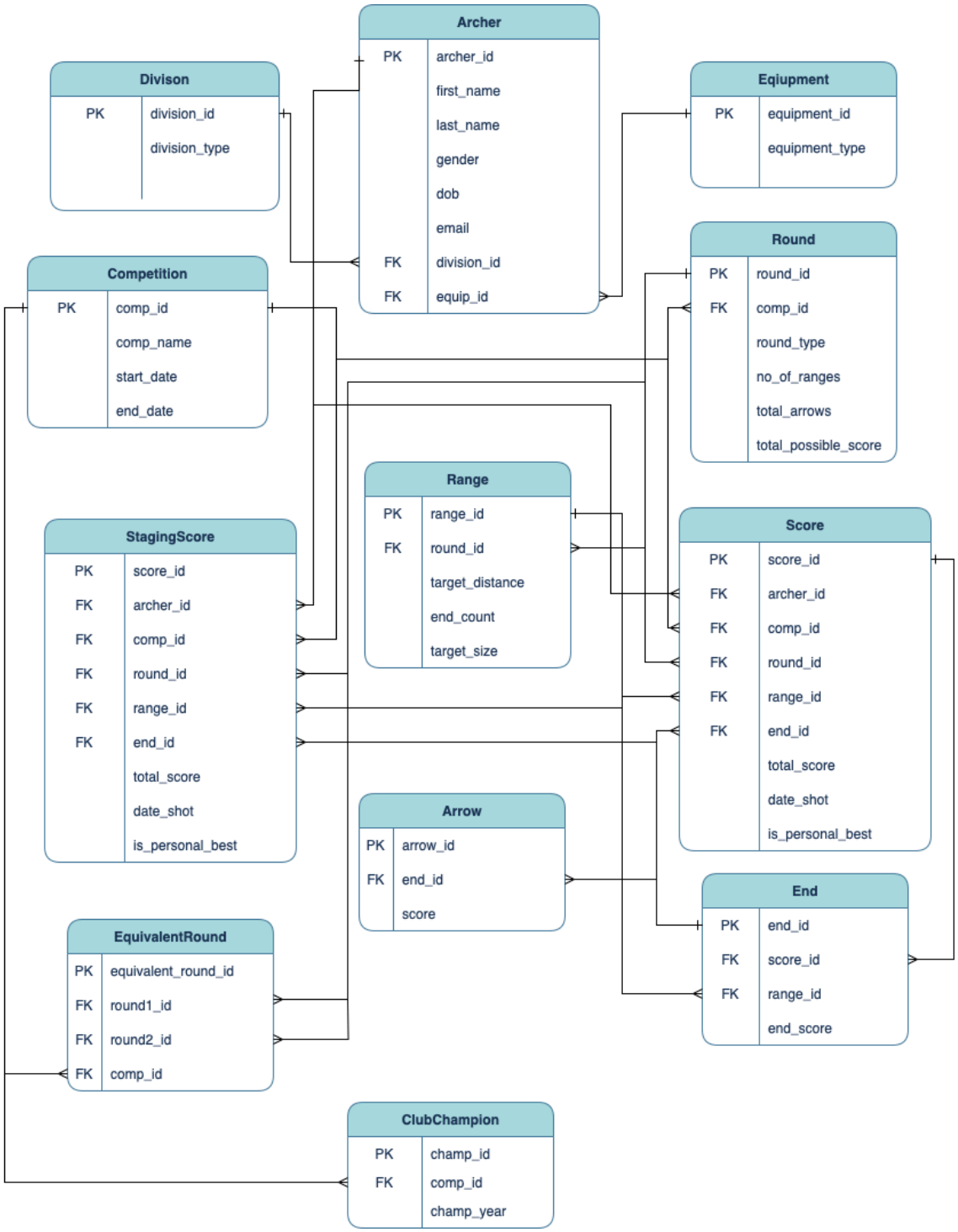
🎨 User interaction and design



⚠️ Out of Scope

- Online registration or payment systems for competitions.
- Integration with external scoring systems or applications.
- Mobile application development (assuming a web-based solution).
- Advanced reporting or analytics features beyond basic score tracking and competition results.

Entity Relationship Diagram



📖 Group Meeting Notes

Meeting overview

We will receive an update on the project's major checkpoints during this conference. We've developed the database, created user personas for a deeper understanding, and set up Jira and Confluence for collaboration.

📝 Meeting minutes

Date	Attendees	Agenda	Notes, decisions and action items
Feb 28, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Team Building and Team Agreement	• First meeting of the group. • Introductions were made and contacts exchanged for team collaboration and follow-ups. • Confluence was set up and a Team Agreement was made with mutual consent.
Mar 6, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Linking Jira to Confluence	• Week 2 meeting. • Confluence was linked to Jira by @Pulkit • Issues were made with the future of the project kept in mind. • A thorough discussion of the project brief was done so that we all had a good understanding of the project requirements and could clarify any doubts with our tutor • Started working on other confluence pages. • The Project plan was made with a group discussion. • Roles and responsibilities were divided although everyone is expected to work together and help out a team member if they are having problems. • Risk Assessment Matrix was also made.

Mar 13, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Entity Modelling	• Week 3 meeting. • Confluence pages made in the previous meeting were completed and polished according to the feedback provided by our tutor. • The project brief was read again for a better understanding. • Had an introduction to the draw.io software. • Started on the Entity Relationship diagram for our database.
Mar 20, 2024	@MelvinGoxhaj @Upek Isuranga @Nikhil mohite @ARSH KHANNA @Pulkit	Personas, Empathy Mapping and Product Requirements	• Week 4 meeting. • Finished up on the ER diagram and got feedback on it from our tutor. • Discussed as a group about the various personas suitable for our product. • Personas was created in relation to the entity diagram and our project situation. • Empathy Maps was created for the previously created personas. • Product Requirements were verified.
Mar 27, 2024	@MelvinGoxhaj @Upek Isuranga @Pulkit @ARSH KHANNA	Progress Check and Resolving Issues	• Week 5 meeting • Checking many-to-many relationship • Investigation for database's nominal form • Updating Jira
Apr 10, 2024	@MelvinGoxhaj @Nikhil mohite @Upek Isuranga @Pulkit	Create Database And Tables	• Week 6 meeting • Creation of Database by @MelvinGoxhaj • Creating tables in the database
Apr 17, 2024	@ARSH KHANNA @Pulkit @Nikhil mohite @Upek Isuranga @MelvinGoxhaj	Define nullable values and Enter Dummy data	• week 7 meeting • Defining nullable values in the database • Creation of dummy data • Decide on the way to upload the data. • Uploading the data to the table.
May 6, 2024	@Pulkit @MelvinGoxhaj @ARSH KHANNA	Populate the table with data for the unit tests to successfully run. Finalize group account and personal account tables.	• Tables are complete, some group members need to adjust minor tweaks such as PK and FK combinations • Unit test command have been created & run against tables, some are not currently passing due to minor schematic differences.

May 15, 2024	@Pulkit @ARSH KHANNA	Explore topics related to majors and data management	• Tasks for different majors were explored in simple forms, focusing on the relevance to data management. • Games and Software Development tasks focused on implementing the main use case of the project.
May 22, 2024	@Pulkit @Upek Isuranga @ARSH KHANNA @MelvinGoxhaj @Nikhil mohite	Finalize the product deliverables and finish the report.	• Front-end is working and enables user input for archer's database. • Demonstration video has been recorded outlining the project. • Confluence report has been updated and finalized for submitting. • Self and peer assessments have been completed and submitted.

👷 Health Inspection Report

ℹ️ 9Use the project team health monitor template to keep track of your team's health. Keep this template in your team space and if there are any areas that you're not confident are green, dive into the plays to get back on track. For detailed facilitation instructions go to health monitor for project teams.

Team name	**High Distinction Only**
Sponsor	Swinburne University of Technology
Health monitor cadence	Weekly

👨‍👨‍👦Team health assessment

With your team, read the definition of each attribute of healthy, high-performing teams out loud. On the count of three have each person rate how they feel the team is doing compared to each definition (thumbs-up/green, thumbs-sideways/yellow, thumbs-down/red). Record the results of each attribute rating in the table. Highlight each cell using this color code: HEALTHY = "We're strong here", BIT SICK = "We're ok... but a little shaky", SICK = "We're not healthy".

Area	Apr 16, 2024	May 13, 2024	May 26, 2024
🚩 Full-time owner There is **one lead who is accountable** for the result of this project. This needs to be someone whose time is at least 80% dedicated to it, and who can champion the mission inside and outside of the team.	Pulkit is our team leader who keeps up to date on the status of our project. He looks after the team member's tasks and directs them to be completed. The other members are also suitable and come from diverse knowledge backgrounds. Overall, this area is quite strong, and we have a good direction from our team lead. • Pulkit - 9 • Melvin - 9 • Arsh - 9 • Nikhil - 10 • Upek - 8	Pulkit continues to excel as our team leader, consistently staying updated on the project's status. He effectively manages our team members' tasks, ensuring they are completed on time. The team is still strong, with members contributing to the project. Pulkit's leadership continues to provide us with clear direction and efficient progress. • Pulkit - • Melvin - • Arsh - • Nikhil - • Upek - 10	Pulkit is still our team leader, staying on track with the project's status. He looks after our team members' tasks, and makes sure they are completed ahead of time. Because of this, we completed our tasks on time with high-quality data. • Pulkit - 10 • Melvin - 10 • Arsh - 9 • Nikhil - 9 • Upek - 10
⚖️ Balanced team **Roles and responsibilities are clear** and agreed upon. The	We know what tasks are to be done, but when something personal comes up, we fall a little behind. Our team	We are doing better now as we know the tasks that must be done. After a team meeting, we discussed and	After our last meeting, we have solved our major problems and have stayed on track to complete our work on time.

project has people with the right blend of skill set. Acknowledge that team members can change by stage.	members can change up the pace and adapt to the changes well at times. • Pulkit - 6 • Melvin - 8 • Arsh - 8 • Nikhil - 7 • Upek - 6	sorted the problems out so our team members become more flexible and adapt to changes when needed. This helped us stay on track and maintain progress. • Pulkit - • Melvin - • Arsh - • Nikhil - • Upek - 7	• Pulkit - 10 • Melvin - 9 • Arsh - 9 • Nikhil - 9 • Upek - 10
💞 **Shared understanding** The team has a **common understanding of why they're here**, the problem/need, are convinced about the idea, confident they have what they need, and trust each other.	Everyone on our team communicates well, knows our project's goal, and trusts each other. We also understand the project's goals and are on the right track to complete them. • Pulkit - 9 • Melvin - 8 • Arsh - 8 • Nikhil - 9 • Upek - 9	Everyone on our team communicates well and understands the project's target. We continue to be on the right track to complete our objectives. Here are the present performance ratings for our team members: • Pulkit: 9 • Melvin: 8 • Arsh: 8 • Nikhil: 9 • Upek: 9	Everyone continues to have good chemistry with one another and understands what we need to do in the project. We have good communication and discuss daily on discord. • Pulkit - 10 • Melvin - 9 • Arsh - 10 • Nikhil - 10 • Upek - 10
💍 **Value and metrics** It's clear **what success means** from a business and user's perspective, and there is a unique value proposition in place for the target users and to the business. Success is defined, with a goal, and how it will be measured.	We sometimes discuss the KPIs and keep track of some of our success, but more work is required in this area. • Pulkit - 6 • Melvin - 7 • Arsh - 7 • Nikhil - 6 • Upek - 6	We have improved a little on this section and kept track of some of our successes, but more work is still required in this area. • Pulkit - 6 • Melvin - 7 • Arsh - 7 • Nikhil - 6 • Upek - 7	We have improved heaps on this section and we fully understand the target users and the goals. • Pulkit - 10 • Melvin - 9 • Arsh - 10 • Nikhil - 10 • Upek - 10
🥽 **Proof of concept** Some sort of demonstration has been created and tested, that demonstrates why this problem needs to be solved, and demonstrates its value.	We have demonstrated why the problem needs to be solved, but more testing could be done. • Pulkit - 5 • Melvin - 8 • Arsh - 8 • Nikhil - 7 • Upek - 6	We have demonstrated why the problem must be solved and have conducted multiple tests on the project. • Pulkit - 5 • Melvin - 8 • Arsh - 8 • Nikhil - 7 • Upek - 7	We polished up our tests and improved our descriptions on the project. • Pulkit - 8 • Melvin - 9 • Arsh - 8 • Nikhil - 9 • Upek - 9
📄 **One-pager** The **project is summarized** in a	We are yet to write a one-pager • Pulkit - 3 • Melvin - 7	We are yet to write a one-pager • Pulkit - 3	We have written a one-pager which outlines the project description and its goals.

one-pager and shared with anyone so that they understand the purpose of the project, and its value.	• Arsh - 7 • Nikhil - 5 • Upek - 5	• Melvin - 7 • Arsh - 7 • Nikhil - 5 • Upek - 5	• Pulkit - 10 • Melvin - 10 • Arsh - 10 • Nikhil - 10 • Upek - 10
🤹‍♀️ **Managed dependencies** Clear understanding of complexity, infrastructure involved, risks, resources, effort, and timeline. Clear understanding of **who we depend on, and who depends on us.**	In general, we know each person's job in the group. We can rely on them from time to time to get work done. • Pulkit - 6 • Melvin - 9 • Arsh - 9 • Nikhil - 9 • Upek - 7	We understand each person's role in the group and can rely on them to get work done as needed. • Pulkit - 8 • Melvin - 9 • Arsh - 9 • Nikhil - 9 • Upek - 9	We understand the member's roles and they continue to deliver reliable work on the project. • Pulkit - 10 • Melvin - 10 • Arsh - 10 • Nikhil - 10 • Upek - 9
🏇 **Velocity** The team is making **incremental progress** by shipping concrete iterations to stakeholders (and, even better, to production), learning along the way, and **implementing lessons learned**, resulting in greater success.	We are making solid progress in getting the work done on time. • Pulkit - 7 • Melvin - 9 • Arsh - 9 • Nikhil - 9 • Upek - 7	We are on a continuous track to get the work done on time. • Pulkit - 9 • Melvin - 9 • Arsh - 9 • Nikhil - 9 • Upek - 9	We have worked hard on the project and got most of our work done on time and we are happy with the end result. • Pulkit - 9 • Melvin - 10 • Arsh - 10 • Nikhil - 10 • Upek - 8

🎯 Focus areas

Ask your team to collectively come up with one attribute you want to focus on. Then, call out ways to move the SICK or BIT SICK toward HEALTHY. Make sure they are actionable, specific, and measurable.

Date	Focus areas and action items
16/04/2024	We need to focus on our communication and need to assign more team meet ups in person in comparison to virtual communications. Moreover, we need to take advantage to our tutor and unit convenor for better understanding of the different concepts and get feedback on our progress ☑ @Pulkit @MelvinGoxhaj @ARSH KHANNA @Upek Isuranga @Nikhil mohite regularly provide updates regarding their individual progress and keep a list of questions to ask the tutor and get feedback ☑ @Pulkit @MelvinGoxhaj @ARSH KHANNA @Upek Isuranga @Nikhil mohite although there are specific roles for everyone for this project it would be great if everyone helped each other with their works at these early stages of the project
13/05/2024	We need to focus on identifying the key aspects of the database functionality as mentioned in the project requirements and try to focus on getting those implementations correct. Moreover, we need to write the one page summary for our project that explains everything in brief.

	- [x] @Pulkit @MelvinGoxhaj @ARSH KHANNA @Upek Isuranga @Nikhil mohite identify key assets of the projects and the main requirements of the database to implement - [x] @Pulkit @MelvinGoxhaj @ARSH KHANNA @Upek Isuranga @Nikhil mohite create a one page description of our project

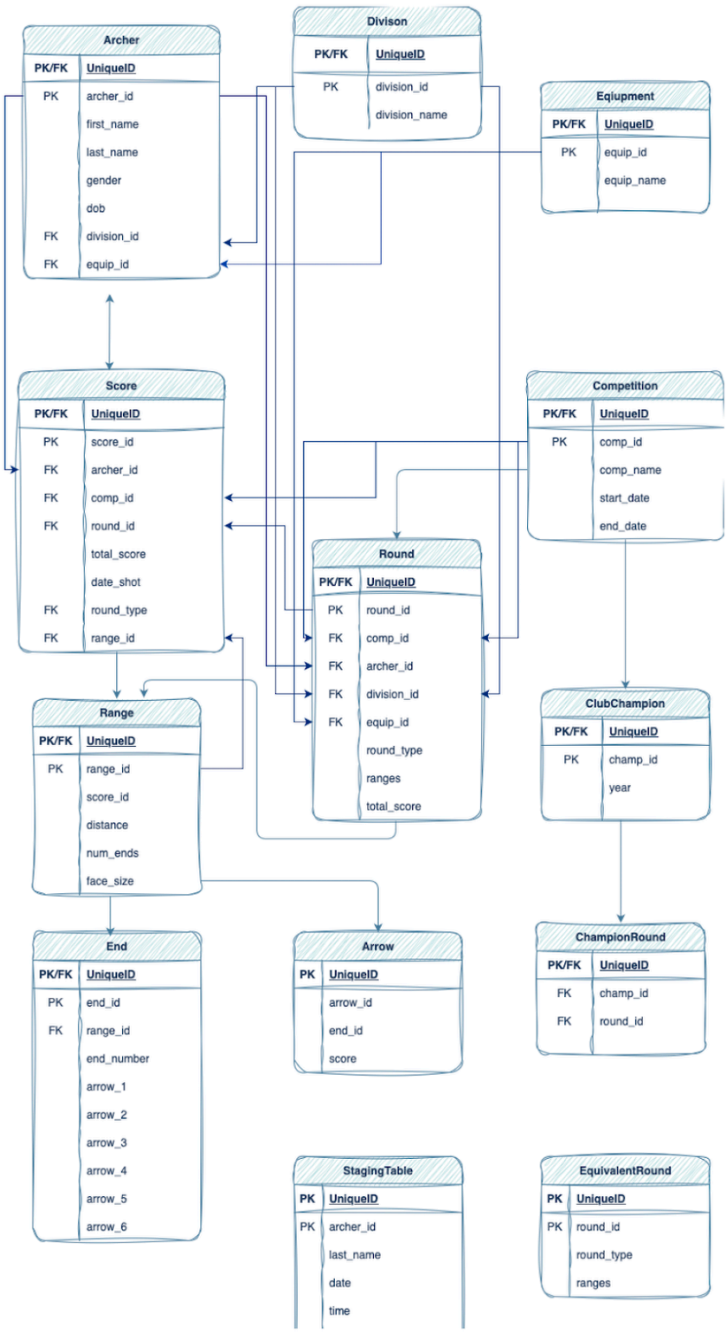
🌹 Next steps

Consider running the plays in suggested in Step 2 of the facilitation instructions for improving BIT SICK and SICK attribute areas. Don't treat these as prescriptions! You know your team better than anyone, so check them out, explore other plays, and do what you think is best.

Review of the ER diagram and Normalisation

Initial Version of the ER

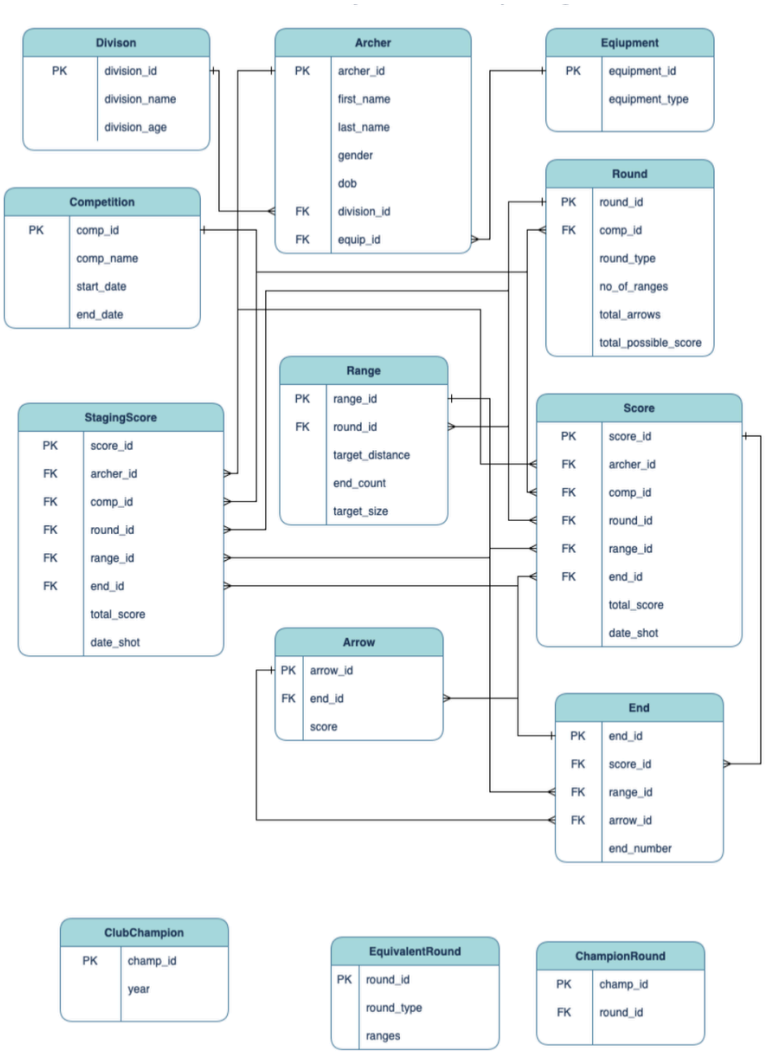
Our initial version of the ER diagram had a basic outline of all, if not most, of the tables we needed in our database. But it wasn't fully functional.



As you can see, the initial ER diagram looks very unprofessional and is badly formatted in the way that the relation between two tables is not clearly represented. Moreover, there are some tables which are not connected to any other table such as Staging Table and Equivalent Round. These tables needed logical connections with the other tables by using correct foreign key constraints so that the functionality of our archery club relational database is perfect. So starting with these reasons we decided to improve on our ER diagram and create the next iteration of it.

First Iteration

For the first iteration of our ER diagram we started off by fixing the formatting of our diagram by arranging all the tables in such a way so that it's easier for anyone to understand the hierarchy of the tables and the relation between each table with respect to the foreign key constraints.

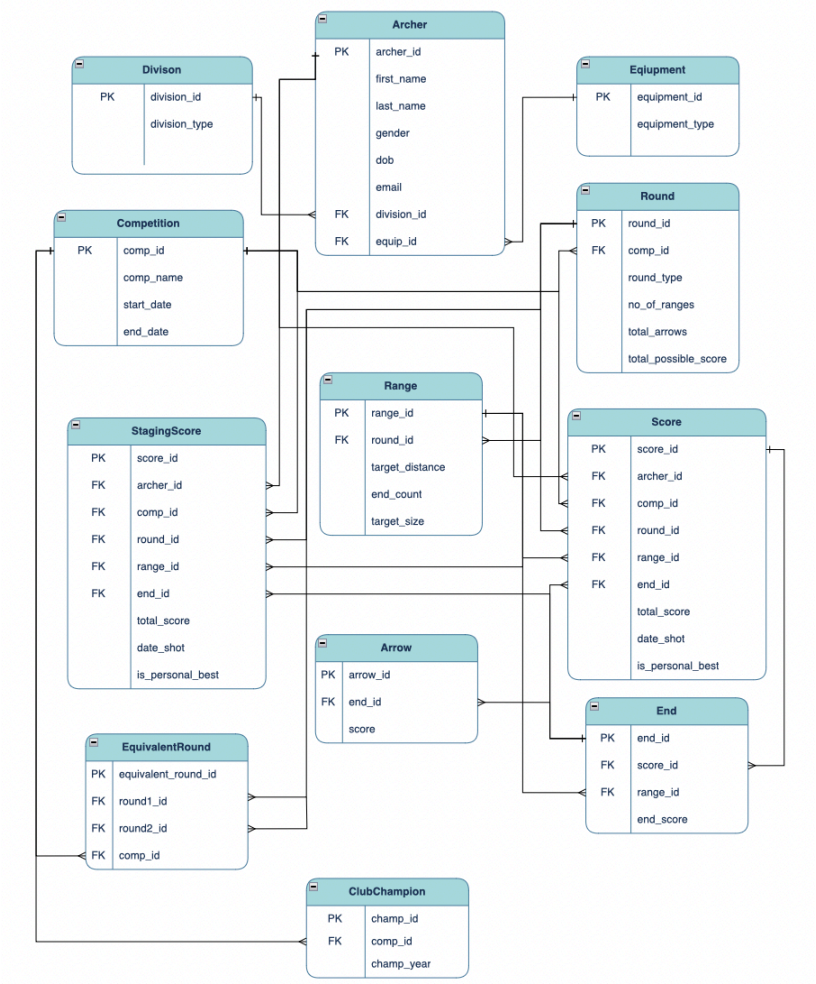


Another thing we added is that we connected the Staging Score table with the rest of the tables in our database by correctly configuring the foreign key constraints. We added a division age column in the Division table as per the necessity of our project requirements. The Round table was fixed by removing the unnecessary foreign key constraints to just the competition id from the Competition table and adding two columns named “total arrows” and “total possible score”. The Range table was updated to use the round id from the round table as its foreign key and got rid of the score id column. The Score table was first organized and then the round type column was removed as it proved to be unnecessary in the implementation of the database. In the previous diagram, the End and the Arrow table were conflicting with each other as the End table had individual columns to store the arrow data and the arrow table would also store individual arrow data. So, we just removed the individual arrow references in the End table and added the score id from the Score table and the arrow id from the

Arrow table as the foreign keys in the End Table. Moreover, some of the tables in the previous diagram were connected incorrectly with rest of the tables so we removed them from the rest so that we could later figure out how to deal with them

Second Iteration

The second iteration improves on the previous version in some subtle details but mostly it was just connecting all the tables with each other so that all te foreign key constraints are satisfied.



So the Equivalent Round and the Club Champion round were connected to the rest of the tables satisfying their foreign keys. In the Equivalent Round table, we added round1_id and round2_id columns as the foreign key round id from the Round table and the competition id as the foreign key from the Competition table. We also removed the division age column from the division table as we realized during testing that it was obsolete. Moreover, we added a “is_personal_best” column to both the Score and Staging Score tables so that we can note and change accordingly if an archer has scored a personal best in a round.

Normalisation

Finally, due to good initial planning while making our ER diagram, all of our tables were already normalised and in the third normal form.

Physical Database (Create Table Statements)

We began by creating the `Division` table to store information about the various categories of archers, such as age groups and skill levels.

```
CREATE TABLE Division (
  division_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  division_name VARCHAR(25) NOT NULL
);
```

The `division_id` column serves as the primary key, ensuring uniqueness and efficient data retrieval. The `division_name` column captures the descriptive details of each division, with the `NOT NULL` constraint enforcing the presence of this value.

Next, we implemented the `Equipment` table to catalog the different types of equipment used by the archers.

```
CREATE TABLE Equipment (
  equipment_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  equipment_type VARCHAR(30) NOT NULL
);
```

The `equipment_id` column is the primary key, and the `equipment_type` column stores the specific type of equipment, with the `NOT NULL` constraint ensuring that this information is provided.

The `Competition` table captures the essential details of each archery event.

```
CREATE TABLE Competition (
  comp_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  comp_name VARCHAR(50) NOT NULL,
  start_date DATE NOT NULL,
  end_date DATE NOT NULL
);
```

The `comp_id` column is the primary key, while the `comp_name`, `start_date`, and `end_date` columns store the critical details of each competition, with the `NOT NULL` constraint enforcing the presence of these values.

The `Round` table stores information about the various rounds within a competition.

```
CREATE TABLE Round (
  round_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  comp_id INT,
  INDEX compID (comp_id),
  FOREIGN KEY (comp_id) REFERENCES Competition(comp_id) ON DELETE SET NULL ON UPDATE CASCADE,
  round_type VARCHAR(25) NOT NULL,
  no_of_ranges INT NOT NULL,
  total_arrows INT NOT NULL,
  total_possible_score INT NOT NULL
);
```

The `round_id` column is the primary key. The `round_type`, `no_of_ranges`, `total_arrows`, and `total_possible_score` columns capture the essential details of each round, with the `NOT NULL` constraint enforcing their presence. The `comp_id` column is a foreign key referencing the `Competition` table's `comp_id` column, maintaining data integrity and allowing for the deletion or update of related records in the `Round` table when the parent record in the `Competition` table is deleted or updated, respectively.

The `Range` table holds details about each range within a round.

```sql
CREATE TABLE `Range` (
  range_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  round_id INT,
  INDEX roundID (round_id),
  FOREIGN KEY (round_id) REFERENCES Round(round_id) ON DELETE SET NULL ON UPDATE CASCADE,
  target_distance INT NOT NULL,
  end_count INT NOT NULL,
  target_size INT NOT NULL
);
```

The `range_id` column is the primary key. The `target_distance`, `end_count`, and `target_size` columns store the specific details of each range, with the `NOT NULL` constraint enforcing their presence. The `round_id` column is a foreign key referencing the `Round` table's `round_id` column, maintaining data integrity and allowing for the deletion or update of related records in the `Range` table when the parent record in the `Round` table is deleted or updated, respectively. Moreover, the `Range` table uses composite keys consisting of `range_id` and `round_id`.

The `Archer` table encapsulates the essential details of each archer.

```sql
CREATE TABLE Archer (
  archer_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  first_name VARCHAR(25) NOT NULL,
  last_name VARCHAR(25) NOT NULL,
  gender VARCHAR(15) NOT NULL,
  dob DATE NOT NULL,
  email VARCHAR(50) NOT NULL,
  division_id INT,
  equipment_id INT,
  INDEX divID (division_id),
  FOREIGN KEY (division_id) REFERENCES Division(division_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX equipID (equipment_id),
  FOREIGN KEY (equipment_id) REFERENCES Equipment(equipment_id) ON DELETE SET NULL ON UPDATE CASCADE
);
```

The `archer_id` column is the primary key. The `first_name`, `last_name`, `gender`, `dob`, and `email` columns store the personal details of each archer, with the `NOT NULL` constraint enforcing their presence. The `division_id` and `equipment_id` columns are foreign keys referencing the `Division` and `Equipment` tables, respectively, maintaining data integrity and allowing for the deletion or update of related records in the `Archer` table when the parent records in the `Division` or `Equipment` tables are deleted or updated, respectively.

The `Score` table houses the scores achieved by each archer in the various competitions, rounds, and ranges.

```sql
CREATE TABLE Score (
  score_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  archer_id INT,
  comp_id INT,
  round_id INT,
  range_id INT,
  INDEX archID (archer_id),
  FOREIGN KEY (archer_id) REFERENCES Archer(archer_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX compID (comp_id),
  FOREIGN KEY (comp_id) REFERENCES Competition(comp_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX roundID (round_id),
  FOREIGN KEY (round_id) REFERENCES Round(round_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX rangeID (range_id),
  FOREIGN KEY (range_id) REFERENCES `Range`(range_id) ON DELETE SET NULL ON UPDATE CASCADE,
  total_score INT NOT NULL,
  date_shot DATE NOT NULL,
  is_personal_best BOOLEAN NOT NULL
);
```

The `score_id` column is the primary key. The `archer_id`, `comp_id`, `round_id`, and `range_id` columns are foreign keys referencing the `Archer`, `Competition`, `Round`, and `Range` tables, respectively, maintaining data integrity and allowing for the deletion or update of related records in the `Score` table when the parent records in the respective tables are deleted or updated. The `total_score`, `date_shot`, and `is_personal_best` columns store the score details, with the `NOT NULL` constraint enforcing their presence.

The `StagingScore` table has an identical structure to the `Score` table, likely used for staging or temporary storage purposes.

```
CREATE TABLE StagingScore (
  score_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  archer_id INT,
  comp_id INT,
  round_id INT,
  range_id INT,
  INDEX archID (archer_id),
  FOREIGN KEY (archer_id) REFERENCES Archer(archer_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX compID (comp_id),
  FOREIGN KEY (comp_id) REFERENCES Competition(comp_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX roundID (round_id),
  FOREIGN KEY (round_id) REFERENCES Round(round_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX rangeID (range_id),
  FOREIGN KEY (range_id) REFERENCES `Range`(range_id) ON DELETE SET NULL ON UPDATE CASCADE,
  total_score INT NOT NULL,
  date_shot DATE NOT NULL,
  is_personal_best BOOLEAN NOT NULL
);
```

The `RangeEnd` table stores the scores for each end (target segment) within a range.

```
CREATE TABLE RangeEnd (
  end_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  score_id INT,
  range_id INT,
  INDEX scoreID (score_id),
  FOREIGN KEY (score_id) REFERENCES Score(score_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX rangeID (range_id),
  FOREIGN KEY (range_id) REFERENCES `Range`(range_id) ON DELETE SET NULL ON UPDATE CASCADE,
  end_score INT NOT NULL
);
```

The `end_id` column is the primary key. The `score_id` and `range_id` columns are foreign keys referencing the `Score` and `Range` tables, respectively, maintaining data integrity and allowing for the deletion or update of related records in the database. Moreover, `score_id` and `range_id` are the composite keys to this table, ensuring that an end can only exist within a specific score.

The `Arrow` table stores information about individual arrow scores within an end.

```
CREATE TABLE Arrow (
  arrow_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  end_id INT,
  INDEX endID (end_id),
  FOREIGN KEY (end_id) REFERENCES RangeEnd (end_id) ON DELETE SET NULL ON UPDATE CASCADE,
  score INT NOT NULL
);
```

Each arrow is assigned a unique `arrow_id` as the primary key. The table includes a foreign key `end_id` that references the RangeEnd table to indicate which end the arrow belongs to. The Arrow table also contains a column for the arrow score. Moreover, similarly like the `RangeEnd` table, it uses `arrow_id` and `end_id` as composite keys so that it ensures that an arrow can only exist within a specific end.

The `EquivalentRound` table represents the relationship between equivalent rounds in different competitions.

```sql
CREATE TABLE EquivalentRound (
  equivalent_round_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  round1_id INT,
  round2_id INT,
  comp_id INT,
  INDEX roundID (round1_id),
  FOREIGN KEY (round1_id) REFERENCES Round(round_id) ON DELETE SET NULL ON UPDATE CASCADE,
  INDEX compID (comp_id),
  FOREIGN KEY (comp_id) REFERENCES Competition(comp_id) ON DELETE SET NULL ON UPDATE CASCADE
);
```

It allows for the comparison and mapping of rounds across competitions. The table has a primary key `equivalent_round_id` to uniquely identify each equivalent round relationship. It includes foreign keys `round1_id`, `round2_id` and `comp_id` that reference the Round and Competition tables, respectively, to establish the relationship between equivalent rounds and their associated competitions.

The `ClubChampion` table stores information about the club champions for each year and competition

```sql
CREATE TABLE ClubChampion (
  champ_id INT NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
  comp_id INT,
  INDEX compID (comp_id),
  FOREIGN KEY (comp_id) REFERENCES Competition(comp_id) ON DELETE SET NULL ON UPDATE CASCADE,
  champ_year DATE NOT NULL
);
```

Each club champion entry is identified by a unique `champ_id` as the primary key. The table includes a foreign key `comp_id` that references the Competition table to indicate the competition in which the club champion was determined. The ClubChampion table also contains a column for the championship year.

Data Creation and Table Population

The process of data population entails utilising MySQL Workbench to methodically import CSV data into a MySQL database. The steps I took and the order I used are explained in detail below:

Details

Data Generation:

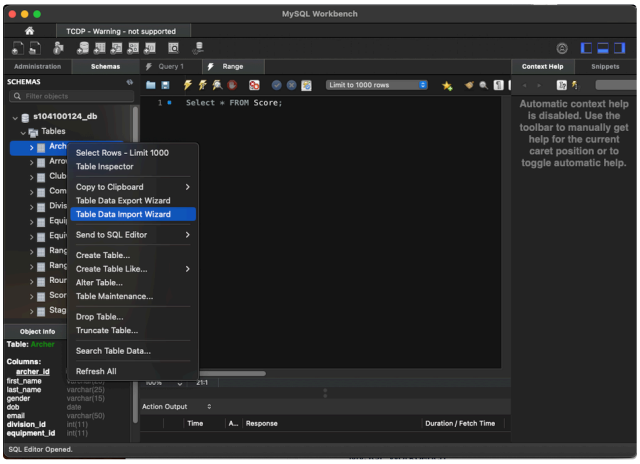
I used ChatGPT, an AI tool, to help with creating data by uploading the table-creation script. I instructed the AI to create files using GPT 4 after uploading the script to the tool.

Software Used to Populate Database:

MySQL Workbench

Steps to Add Data through MySQL Workbench:

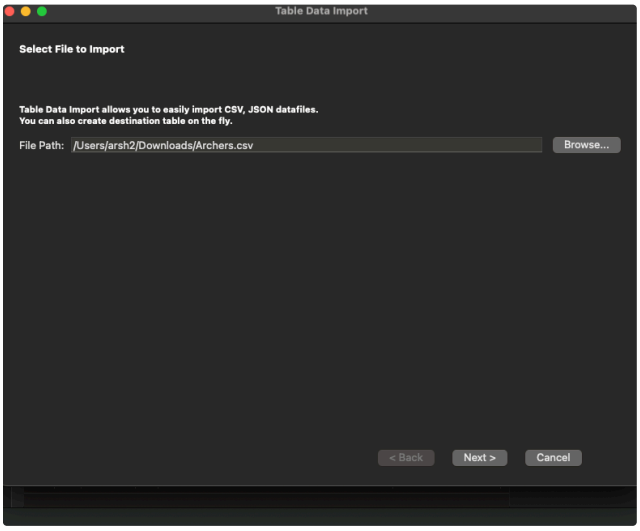
1. Connect to the Database
2. Right-click on the database and select Table Data Import Wizard.



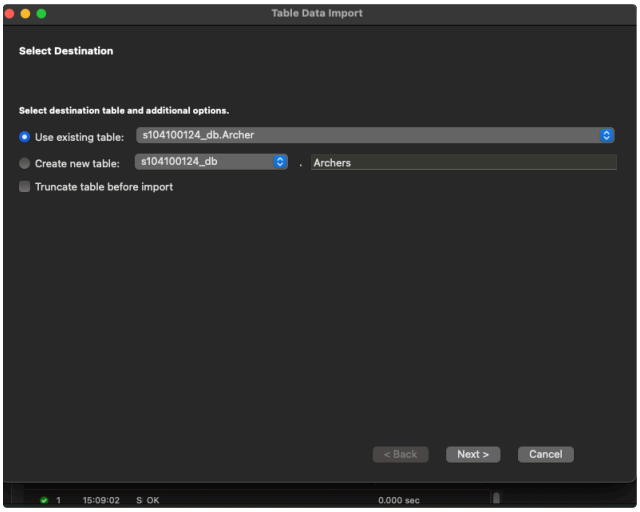
3. Select your CSV file.

Archers

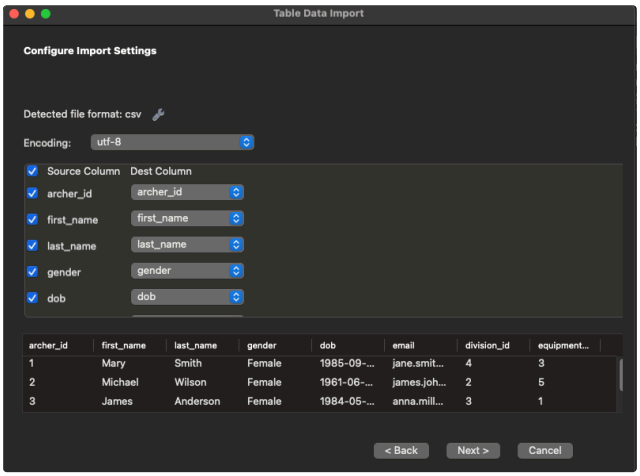
archer_id	first_name	last_name	gender	dob	email	division_id	equipment_id
1	Mary	Smith	Female	1985-09-01	jane.smith@example.com	4	3
2	Michael	Wilson	Female	1961-06-24	james.johnson@example.com	2	5
3	James	Anderson	Female	1984-05-28	anna.miller@example.com	3	1
4	Mary	Taylor	Female	1995-12-27	anna.smith@example.com	5	2
5	James	Johnson	Female	1957-02-20	robert.brown@example.com	1	2
6	David	Brown	Male	1963-04-16	linda.davis@example.com	16	5
7	David	Anderson	Male	1971-08-09	robert.johnson@example.com	14	5
8	Jane	Williams	Female	1966-01-18	james.johnson@example.com	6	3
9	Jane	Johnson	Male	1994-01-05	jane.brown@example.com	7	1
10	Robert	Davis	Female	1966-02-09	mary.anderson@example.com	15	3
11	Michael	Davis	Male	1999-04-08	jane.anderson@example.com	5	3



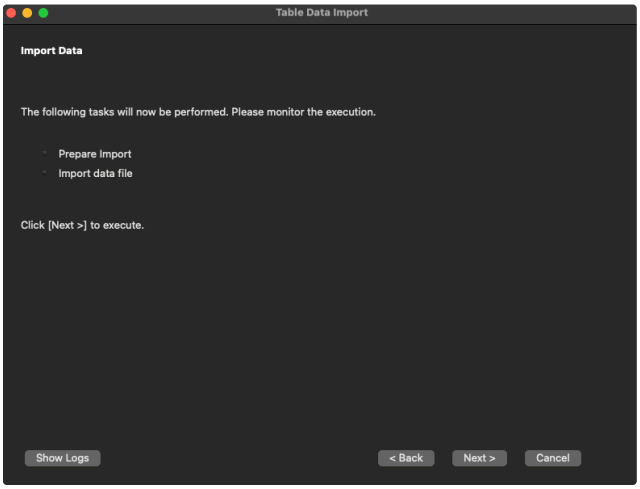
4. Select an existing table or enter a name for a new table.



5. Configure the import settings (data types, line separator, etc).



6. Finish the process.



Order of Data Population:

1. Division And Equipment: Since these tables don't rely on any other tables, I chose to import them first.
2. Archer: Because it depends on the Division and Equipment tables for foreign keys, I chose to import next.
3. Competition: Since this table doesn't rely on information related to archery, I chose to import it after archer.
4. Round: Since it depends on the competition table, I choose to import after it.
5. Range: As it depends on Round, I made the decision to import after round.
6. Score And Staging Score: Because they depend on Archer, Competition, Round, and Range, I choose to import this table next.
7. RangeEnd: Since it depends on Score and Range, I choose to import next.
8. Arrow: Given that this table depends on RangeEnd, I chose to import it next.
9. EquivalentRound: Since it depends on the round and the competition, I choose to import next.
10. ClubChampion: Because it depends on the competition, I chose to import this table last.

Challenges Faced:

1. Deciding on the best approach for data population.

2. To Set Priorities What Table should be populated first.

How I overcame all the challenges I encountered:

1. I did a little research on it to determine the best course of action, and I ultimately decided to use the Table Data Import Wizard in MySQL Workbench to populate the table.
2. To Prioritize Which Table ought to be filled up initially? I looked at the database's ER diagram and prioritised based on table dependencies.

Highlight important information in a panel like this one. To edit this panel's color or style, select one of the options in the menu.

📋 References

1. https://chatgpt.com
2. How to Import a CSV using MySQL Workbench. Database Star. https://www.databasestar.com/mysql-workbench-import-csv/

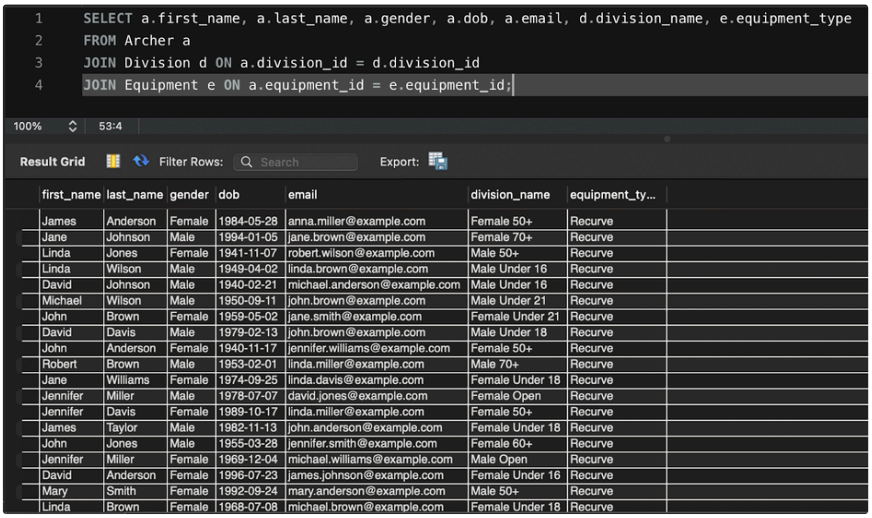
Data Creation and Table Population

Use Cases and SQL Statements

USE CASE 1

If the user wants to retrieve a list of all the archers with their personal details and default equipment they can run this query. Here, we join the division and equipment table to the archer table on division id and equipment id respectively:

```
SELECT a.first_name, a.last_name, a.gender, a.dob, a.email, d.division_name, e.equipment_type
FROM Archer a
JOIN Division d ON a.division_id = d.division_id
JOIN Equipment e ON a.equipment_id = e.equipment_id;
```



```
SELECT a.first_name, a.last_name, a.gender, a.dob, a.email, d.division_name, e.equipment_type
FROM Archer a
JOIN Division d ON a.division_id = d.division_id
JOIN Equipment e ON a.equipment_id = e.equipment_id;
```

100% 53:4

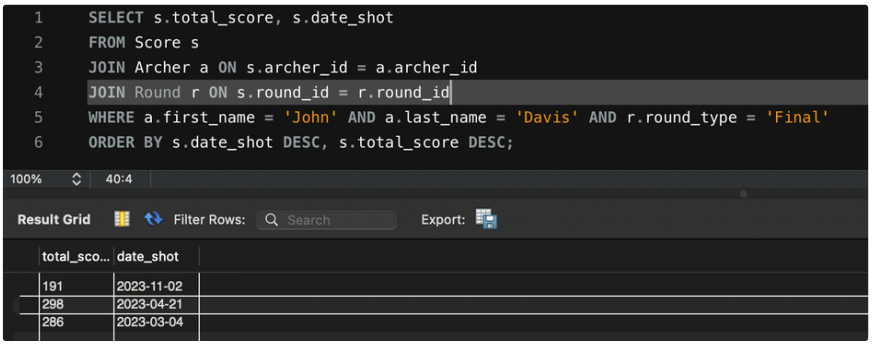
Result Grid Filter Rows: Search Export:

first_name	last_name	gender	dob	email	division_name	equipment_ty...
James	Anderson	Female	1984-05-28	anna.miller@example.com	Female 50+	Recurve
Jane	Johnson	Male	1994-01-05	jane.brown@example.com	Female 70+	Recurve
Linda	Jones	Female	1941-11-07	robert.wilson@example.com	Male 50+	Recurve
Linda	Wilson	Male	1949-04-02	linda.brown@example.com	Male Under 16	Recurve
David	Johnson	Male	1940-02-21	michael.anderson@example.com	Male Under 16	Recurve
Michael	Wilson	Male	1950-09-11	john.brown@example.com	Male Under 21	Recurve
John	Brown	Female	1959-05-02	jane.smith@example.com	Female Under 21	Recurve
David	Davis	Male	1979-02-13	john.brown@example.com	Male Under 18	Recurve
John	Anderson	Female	1940-11-17	jennifer.williams@example.com	Female 50+	Recurve
Robert	Brown	Male	1953-02-01	linda.miller@example.com	Male 70+	Recurve
Jane	Williams	Female	1974-09-25	linda.davis@example.com	Female Under 18	Recurve
Jennifer	Miller	Male	1978-07-07	david.jones@example.com	Female Open	Recurve
Jennifer	Davis	Female	1989-10-17	linda.miller@example.com	Female 50+	Recurve
James	Taylor	Male	1982-11-13	john.anderson@example.com	Female Under 18	Recurve
John	Jones	Male	1955-03-28	jennifer.smith@example.com	Female 60+	Recurve
Jennifer	Miller	Female	1969-12-04	michael.williams@example.com	Male Open	Recurve
David	Anderson	Female	1996-07-23	james.johnson@example.com	Female Under 16	Recurve
Mary	Smith	Female	1992-09-24	mary.anderson@example.com	Male 50+	Recurve
Linda	Brown	Female	1968-07-08	michael.brown@example.com	Female Under 18	Recurve

USE CASE 2

If the user wants to retrieve the scores of an archer for a specific given round type, they can use the archers first name, last name and the round type to get the scores for that round on each date shot. Here, we join the Archer and Round tables to the Score table on archer id and round id respectively:

```
SELECT s.total_score, s.date_shot
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
WHERE a.first_name = 'John' AND a.last_name = 'Davis' AND r.round_type = 'Final'
ORDER BY s.date_shot DESC, s.total_score DESC;
```



```
SELECT s.total_score, s.date_shot
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
WHERE a.first_name = 'John' AND a.last_name = 'Davis' AND r.round_type = 'Final'
ORDER BY s.date_shot DESC, s.total_score DESC;
```

100% 40:4

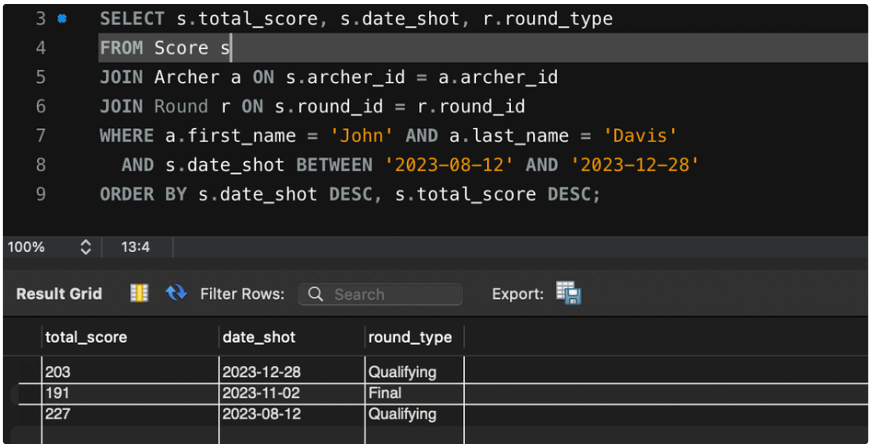
Result Grid Filter Rows: Search Export:

total_sco...	date_shot
191	2023-11-02
298	2023-04-21
286	2023-03-04

USE CASE 3

If the user wants to get the score of a particular archer within a specific date range, it can be done by this query. Here, we join the Archer and Round table to the Score table on archer i and round id respectively:

```
SELECT s.total_score, s.date_shot, r.round_type
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
WHERE a.first_name = 'John' AND a.last_name = 'Davis'
  AND s.date_shot BETWEEN '2023-08-12' AND '2023-12-28'
ORDER BY s.date_shot DESC, s.total_score DESC;
```

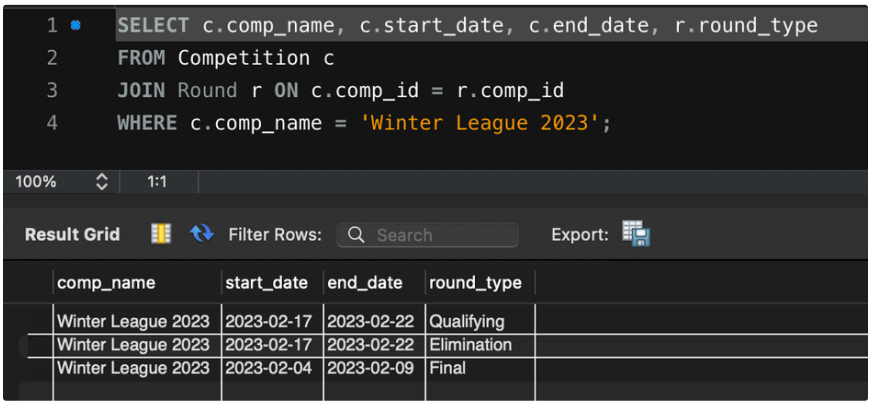


USE CASE 4

If the user wants the details for a specific competition they can get it by this query

Retrieve the details of a specific competition. Here we join the Round table to the Competition table on the competition id of both the tables:

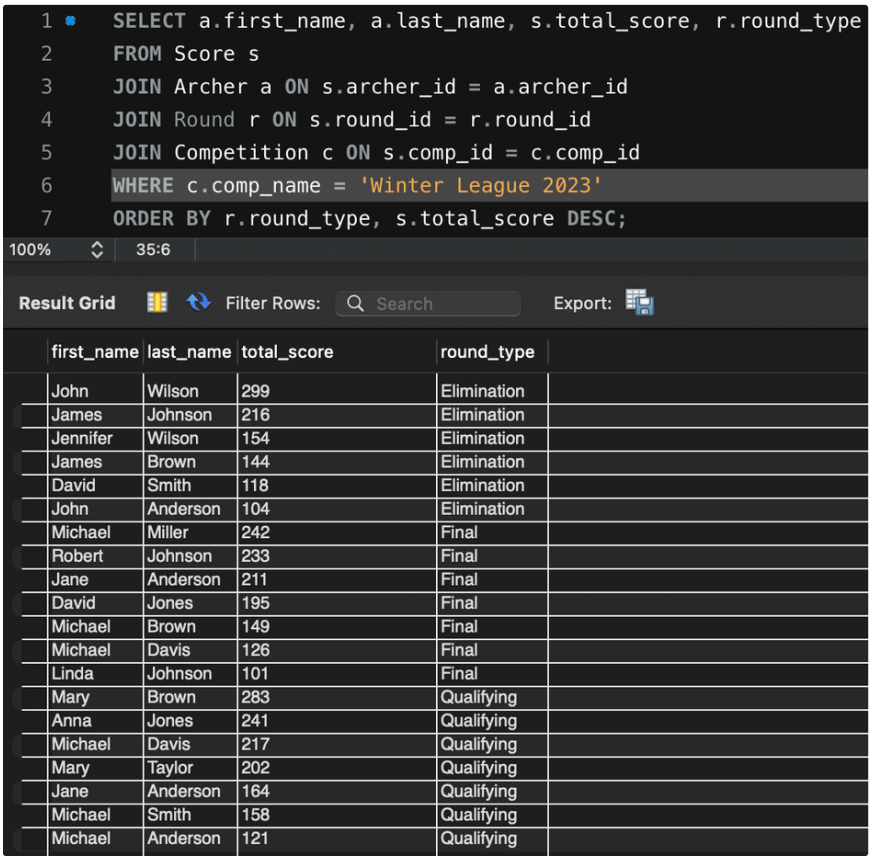
```
SELECT c.comp_name, c.start_date, c.end_date, r.round_type
FROM Competition c
JOIN Round r ON c.comp_id = r.comp_id
WHERE c.comp_name = 'Winter League 2023';
```



USE CASE 5

If the user wants to retrieve the scores of all archers for a specific competition, we use this query by specifying the competition name. Here, we join the Archer, Round and Competition tables to the Score table on archer id, round id and competition id respectively:

```
SELECT a.first_name, a.last_name, s.total_score, r.round_type
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
JOIN Competition c ON s.comp_id = c.comp_id
WHERE c.comp_name = 'Winter League 2023'
ORDER BY r.round_type, s.total_score DESC;
```



```
SELECT a.first_name, a.last_name, s.total_score, r.round_type
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
JOIN Competition c ON s.comp_id = c.comp_id
WHERE c.comp_name = 'Winter League 2023'
ORDER BY r.round_type, s.total_score DESC;
```

100% 35:6

Result Grid Filter Rows: Search Export:

first_name	last_name	total_score	round_type
John	Wilson	299	Elimination
James	Johnson	216	Elimination
Jennifer	Wilson	154	Elimination
James	Brown	144	Elimination
David	Smith	118	Elimination
John	Anderson	104	Elimination
Michael	Miller	242	Final
Robert	Johnson	233	Final
Jane	Anderson	211	Final
David	Jones	195	Final
Michael	Brown	149	Final
Michael	Davis	126	Final
Linda	Johnson	101	Final
Mary	Brown	283	Qualifying
Anna	Jones	241	Qualifying
Michael	Davis	217	Qualifying
Mary	Taylor	202	Qualifying
Jane	Anderson	164	Qualifying
Michael	Smith	158	Qualifying
Michael	Anderson	121	Qualifying

USE CASE 6

If the user wants to retrieve the details of rounds and their corresponding ranges they use this query where we join the Range table to the Round table on the round id in both of the tables:

```
SELECT r.round_type, r.no_of_ranges, r.total_arrows, r.total_possible_score,
       ra.target_distance, ra.end_count, ra.target_size
FROM Round r
JOIN `Range` ra ON r.round_id = ra.round_id;
```

```
SELECT r.round_type, r.no_of_ranges, r.total_arrows, r.total_possible_score,
       ra.target_distance, ra.end_count, ra.target_size
FROM Round r
JOIN `Range` ra ON r.round_id = ra.round_id;
```

100% 1:6

Result Grid Filter Rows: Search Export:

round_type	no_of_ranges	total_arro...	total_possible_sc...	target_distan...	end_count	target_size
Final	1	71	352	30	10	69
Final	1	71	352	50	9	64
Final	1	71	352	20	9	57
Final	1	71	352	20	7	94
Elimination	3	76	349	20	6	53
Elimination	3	76	349	40	10	84
Elimination	3	76	349	70	6	43
Elimination	3	76	349	20	10	91
Final	4	36	682	10	6	108
Final	4	36	682	60	10	89
Final	4	36	682	50	7	87
Elimination	2	60	787	30	9	94
Elimination	2	60	787	10	6	54
Elimination	2	60	787	40	9	58
Elimination	2	60	787	30	8	87
Final	3	36	682	60	8	88
Final	3	36	682	30	5	42
Final	3	36	682	40	5	113
Final	3	36	682	60	5	64
Final	2	38	476	40	10	49
Final	2	38	476	10	6	89
Final	2	38	476	30	10	91
Final	2	38	476	10	5	115
Final	2	38	476	40	5	101
Qualifying	1	58	676	40	10	86
Qualifying	1	58	676	90	10	98
Qualifying	1	58	676	40	7	108
Qualifying	1	58	676	60	9	86
Qualifying	1	72	649	10	8	115

USE CASE 7

If the user wants to retrieve the equivalent rounds for a specific competition, they can get the results by mentioning the competition round they want to get the equivalent rounds for. This is done by joining the Round and Competition tables to the Equivalent Round table on the round1 id of the equivalent round table with the round id of the Round table and competition id of both the Equivalent Round and Competition table:

```
SELECT r.round_type
FROM EquivalentRound er
JOIN Round r ON er.round1_id = r.round_id
JOIN Competition c ON er.comp_id = c.comp_id
WHERE c.comp_name = 'Autumn Open 2023';
```

```
SELECT r.round_type
FROM EquivalentRound er
JOIN Round r ON er.round1_id = r.round_id
JOIN Competition c ON er.comp_id = c.comp_id
WHERE c.comp_name = 'Autumn Open 2023';
```

100% 1:6

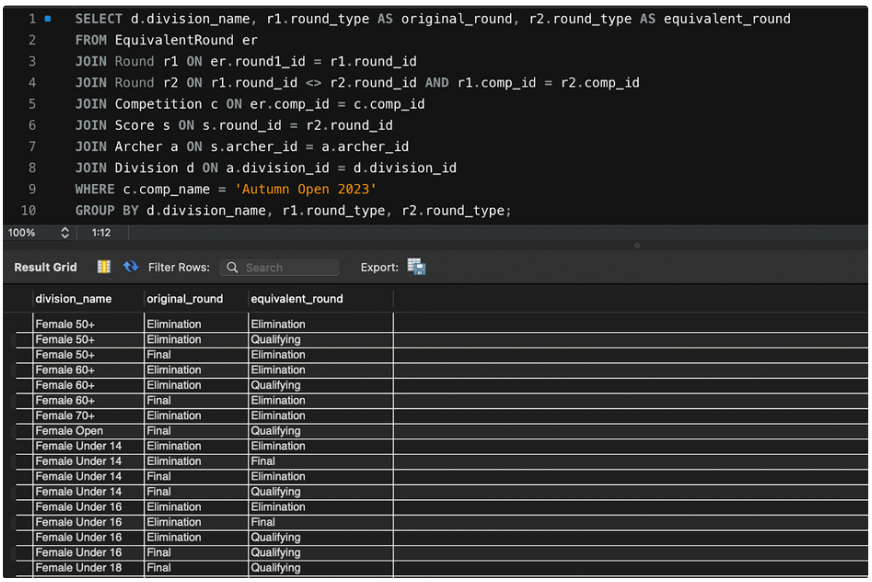
Result Grid Filter Rows: Search Export:

round_type
Final
Final
Elimination
Elimination
Final
Elimination

USE CASE 8

If the user wants to retrieving equivalent rounds for a specific competition with better description and detail, they can do the same as before but now we are also joining the Equivalent Round table to the Score, Archer and Division tables on the round id, archer id and division id respectively:

```
SELECT d.division_name, r1.round_type AS original_round, r2.round_type AS equivalent_round
FROM EquivalentRound er
JOIN Round r1 ON er.round1_id = r1.round_id
JOIN Round r2 ON r1.round_id <> r2.round_id AND r1.comp_id = r2.comp_id
JOIN Competition c ON er.comp_id = c.comp_id
JOIN Score s ON s.round_id = r2.round_id
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Division d ON a.division_id = d.division_id
WHERE c.comp_name = 'Autumn Open 2023'
GROUP BY d.division_name, r1.round_type, r2.round_type;
```



```
SELECT d.division_name, r1.round_type AS original_round, r2.round_type AS equivalent_round
FROM EquivalentRound er
JOIN Round r1 ON er.round1_id = r1.round_id
JOIN Round r2 ON r1.round_id <> r2.round_id AND r1.comp_id = r2.comp_id
JOIN Competition c ON er.comp_id = c.comp_id
JOIN Score s ON s.round_id = r2.round_id
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Division d ON a.division_id = d.division_id
WHERE c.comp_name = 'Autumn Open 2023'
GROUP BY d.division_name, r1.round_type, r2.round_type;
```

100% 1:12

Result Grid Filter Rows: Search Export:

division_name	original_round	equivalent_round
Female 50+	Elimination	Elimination
Female 50+	Elimination	Qualifying
Female 50+	Final	Elimination
Female 60+	Elimination	Elimination
Female 60+	Elimination	Qualifying
Female 60+	Final	Elimination
Female 70+	Elimination	Elimination
Female Open	Final	Qualifying
Female Under 14	Elimination	Elimination
Female Under 14	Elimination	Final
Female Under 14	Final	Elimination
Female Under 14	Final	Qualifying
Female Under 16	Elimination	Elimination
Female Under 16	Elimination	Final
Female Under 16	Elimination	Qualifying
Female Under 16	Final	Qualifying
Female Under 18	Final	Qualifying

USE CASE 9

If the user wants to retrieve the arrow scores for a specific end in a score they use this query by joining the Arrow table to the Range End and Score table on the end id and score id respectively:

```
SELECT a.score
FROM Arrow a
JOIN RangeEnd re ON a.end_id = re.end_id
JOIN Score s ON re.score_id = s.score_id
WHERE s.score_id = 1 AND re.end_id = 17;
```

```
SELECT a.score
FROM Arrow a
JOIN RangeEnd re ON a.end_id = re.end_id
JOIN Score s ON re.score_id = s.score_id
WHERE s.score_id = 1 AND re.end_id = 17;
```

100% 1:1

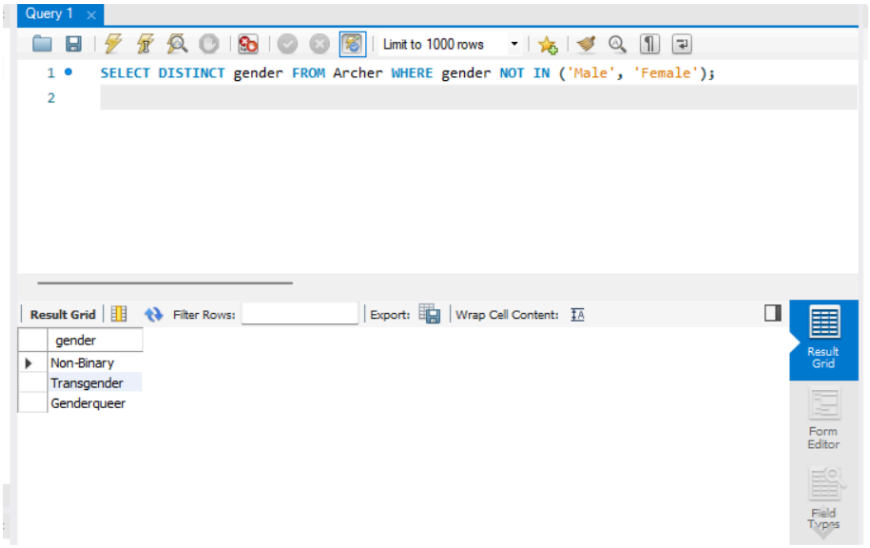
Result Grid Filter Rows: Search Export:

score
1
4
8

USE CASE 10

If the user wants to retrieve data to check if a valid gender is entered they use this query:

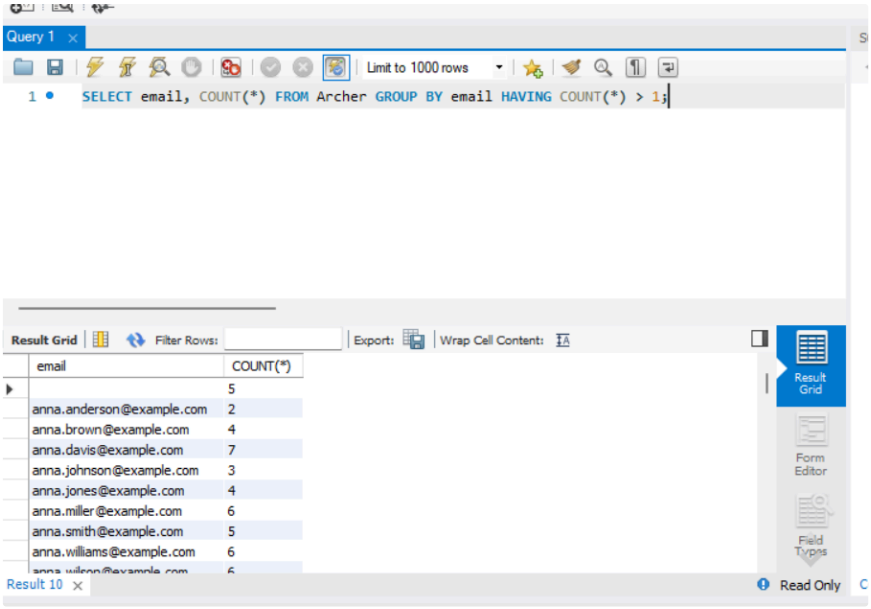
```
SELECT DISTINCT gender FROM Archer WHERE gender NOT IN ('Male', 'Female');
```



USE CASE 11

If the user wants to retrieve data for duplicate emails and shows count of emails, they use this query:

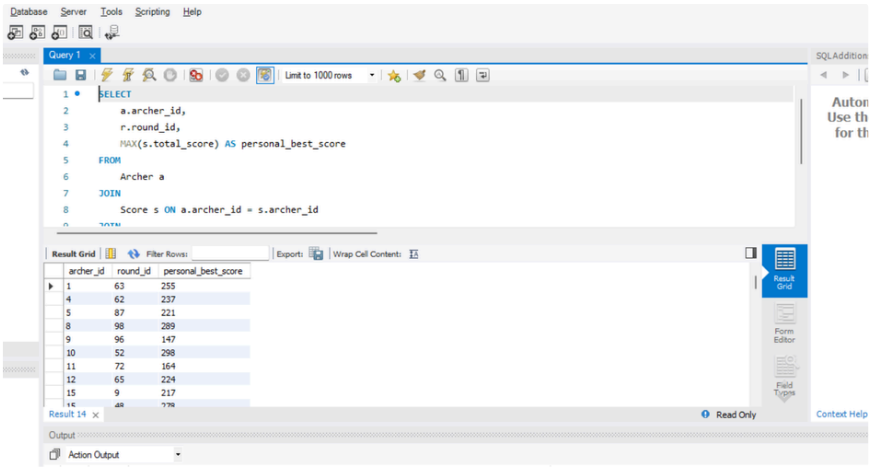
```
SELECT email, COUNT(*) FROM Archer GROUP BY email HAVING COUNT(*) > 1;
```



USE CASE 12

If the user wants to retrieve the maximum scores as personal best scores of all archers for a specific round type, they use this query where we join the Score and Round table to the Archer table on archer id and round id respectively:

```
SELECT
    a.archer_id,
    r.round_id,
    MAX(s.total_score) AS personal_best_score
FROM
    Archer a
JOIN
    Score s ON a.archer_id = s.archer_id
JOIN
    Round r ON s.round_id = r.round_id
GROUP BY
    a.archer_id,
    R.round_id;
```

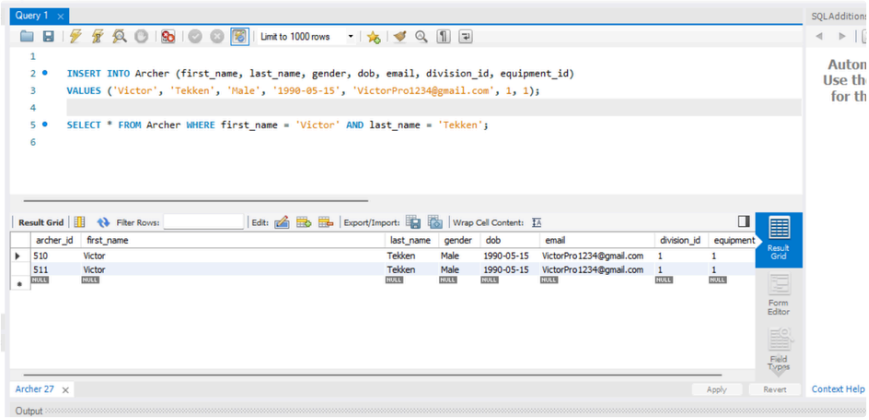


USE CASE 13

If the recorder wants to insert new archers and their details, they can simply do this by using a simple Insert query provided below:

```
INSERT INTO Archer (first_name, last_name, gender, dob, email, division_id, equipment_id)
VALUES ('Victor', 'Tekken', 'Male', '1990-05-15', 'VictorPro1234@gmail.com', 1, 1);
```

```
SELECT * FROM Archer WHERE first_name = 'Victor' AND last_name = 'Tekken';
```



USE CASE 14

If the user wants to retrieve the personal best scores for all the archers for a particular round, they can do this by specifying the round type the want to get the scores for and the query joins the Archer and Round table to the Score table on archer id and round id respectively:

```
SELECT a.first_name, a.last_name, s.total_score, s.date_shot
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
WHERE s.is_personal_best = TRUE AND r.round_type = 'Final'
ORDER BY s.total_score DESC;
```

```
SELECT a.first_name, a.last_name, s.total_score, s.date_shot
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
WHERE s.is_personal_best = TRUE AND r.round_type = 'Final'
ORDER BY s.total_score DESC;
```

100% 1:9

Result Grid Filter Rows: Search Export:

first_name	last_name	total_score	date_shot
James	Johnson	299	2023-02-09
Robert	Davis	298	2023-12-18
Jane	Anderson	296	2023-04-17
David	Miller	291	2023-07-20
Michael	Williams	291	2023-06-17
Michael	Taylor	290	2023-10-26
Jane	Jones	290	2023-11-12
Linda	Smith	289	2023-10-02
James	Williams	288	2023-06-17
Robert	Johnson	285	2023-04-14
James	Miller	285	2023-09-06
David	Miller	282	2023-11-27
Anna	Wilson	278	2023-09-10
Anna	Anderson	278	2023-02-05
John	Anderson	273	2023-01-01
Mary	Davis	270	2023-07-06
Anna	Miller	270	2023-03-18
Anna	Jones	264	2023-03-03
Jane	Taylor	263	2023-04-07
John	Jones	260	2023-04-04

USE CASE 14

If the user wants to retrieve the club champions for a specific year, they can do this by specifying the championship year and the query joins the Archer, Round and Competition tables to the Score table on archer id, round id and comp id respectively and joins the Club Champion table to the Competition table on comp id of both the tables:

```
SELECT a.first_name, a.last_name, s.total_score, r.round_type
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
JOIN Competition c ON s.comp_id = c.comp_id
JOIN ClubChampion cc ON c.comp_id = cc.comp_id
WHERE cc.champ_year = '2024-07-18'
ORDER BY s.total_score DESC;
```

```
3  SELECT a.first_name, a.last_name, s.total_score, r.round_type
4  FROM Score s
5  JOIN Archer a ON s.archer_id = a.archer_id
6  JOIN Round r ON s.round_id = r.round_id
7  JOIN Competition c ON s.comp_id = c.comp_id
8  JOIN ClubChampion cc ON c.comp_id = cc.comp_id
9  WHERE cc.champ_year = '2024-07-18'
10 ORDER BY s.total_score DESC;
```

100% 28:1

Result Grid Filter Rows: Search Export:

first_name	last_name	total_score	round_type
Mary	Jones	263	Final
Michael	Anderson	260	Elimination
Mary	Anderson	246	Elimination
Mary	Taylor	231	Final
Mary	Brown	220	Final
Michael	Smith	199	Qualifying
Mary	Taylor	196	Elimination
Jennifer	Brown	143	Qualifying
Jane	Davis	124	Qualifying
Anna	Anderson	121	Final
Mary	Miller	107	Qualifying

USE CASE 15

If the Archers want to retrieve their own scores by date, range and type of round, they call this query by specifying their archer id, date shot and the round type. Here in this query, we join the Archer and Round table to the Score table on archer id and round id respectively:

```
SELECT s.score_id, s.total_score, s.date_shot, r.round_type
FROM Score s
JOIN Archer a ON s.archer_id = a.archer_id
JOIN Round r ON s.round_id = r.round_id
WHERE a.archer_id = 336
  AND s.date_shot >= '2023-05-07'
  AND r.round_type = 'Final'
ORDER BY s.date_shot DESC, s.total_score DESC
LIMIT 10;
```

```
SELECT s.score_id, s.total_score, s.date_shot, r.round_type
4  FROM Score s
5  JOIN Archer a ON s.archer_id = a.archer_id
6  JOIN Round r ON s.round_id = r.round_id
7  WHERE a.archer_id = 336
8    AND s.date_shot >= '2023-05-07'
9    AND r.round_type = 'Final'
10 ORDER BY s.date_shot DESC, s.total_score DESC
11 LIMIT 10;
```

100% 40:1

Result Grid Filter Rows: Search Export:

score_id	total_score	date_shot	round_type
497	192	2023-10-07	Final

USE CASE 16

If the user wants to retrieve the competition results ordered by total score of an archer, they can do this by specifying the competition name and the query join the Archer, Competition and the Range End tables to the Score table on archer id, comp id and score id respectively:

```
SELECT
  c.comp_name AS Competition,
  CONCAT(a.first_name, ' ', a.last_name) AS Archer,
  SUM(re.end_score) AS TotalScore
FROM
  Score s
  JOIN Archer a ON s.archer_id = a.archer_id
  JOIN Competition c ON s.comp_id = c.comp_id
  JOIN RangeEnd re ON s.score_id = re.score_id
WHERE
  c.comp_name = 'Autumn Open 2020'
GROUP BY
  c.comp_id,
  c.comp_name,
  a.archer_id,
  a.first_name,
  a.last_name
ORDER BY
  TotalScore DESC;
```

```sql
SELECT
  c.comp_name AS Competition,
  CONCAT(a.first_name, ' ', a.last_name) AS Archer,
  SUM(re.end_score) AS TotalScore
FROM
  Score s
  JOIN Archer a ON s.archer_id = a.archer_id
  JOIN Competition c ON s.comp_id = c.comp_id
  JOIN RangeEnd re ON s.score_id = re.score_id
WHERE
  c.comp_name = 'Autumn Open 2020'
GROUP BY
  c.comp_id,
  c.comp_name,
  a.archer_id,
  a.first_name,
  a.last_name
ORDER BY
  TotalScore DESC;
```

100% 35:11

Result Grid Filter Rows: Search Export:

Competition	Archer	TotalScore
Autumn Open 2020	Anna Anderson	259
Autumn Open 2020	Linda Johnson	174
Autumn Open 2020	John Johnson	169
Autumn Open 2020	David Jones	158
Autumn Open 2020	Jane Jones	135
Autumn Open 2020	Anna Brown	124
Autumn Open 2020	James Johnson	109
Autumn Open 2020	Jennifer Davis	104
Autumn Open 2020	David Brown	99

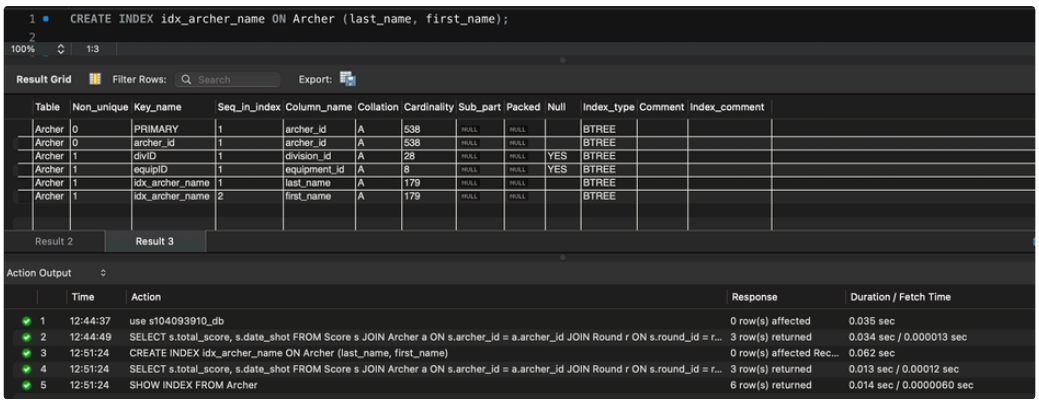
Indexing (Performance Improvement)

Indexing a column helps to speed up the look-up process from a database and helps in the overall performance of the application. For our database tables, we have decided to index 7 columns in total based on their frequency of use in our look-up test queries and the select statements in general.

last_name & first_name

An Archer can be looked up by his/her last and first name apart from their Archer ID. Since there will be a lot of look-ups regarding the Archers name, hence it is useful to index these columns.

```
CREATE INDEX idx_archer_name ON Archer (last_name, first_name);
```

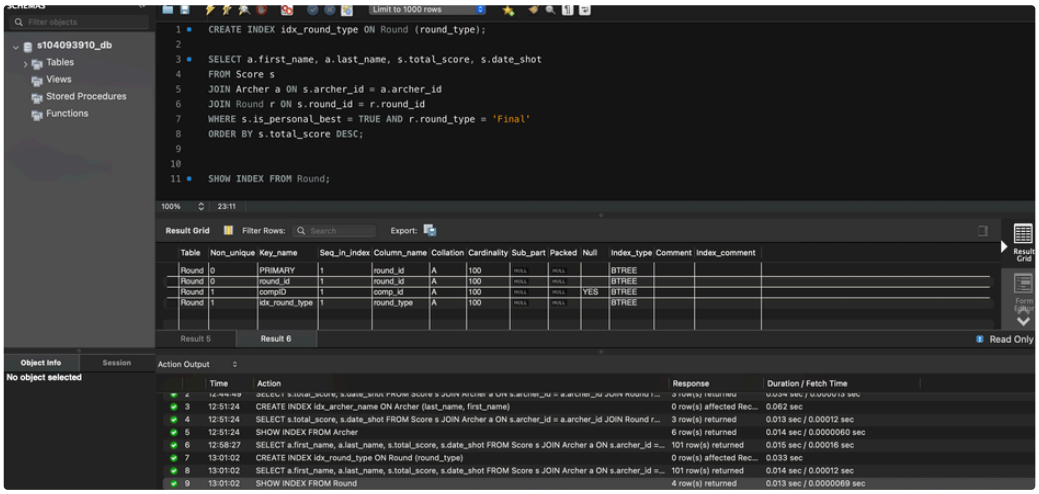


Notice the difference in the Duration/Fetch Time for running the query before and after Indexing.

round_type

Majority of the Archers or a member of the public would like to retrieve the details of a particular round in a competition. Hence, the select statements for getting these results would look-up a lot in the round_type column of the Round table. Therefore, we are indexing this column.

```
CREATE INDEX idx_round_type ON Round (round_type);
```

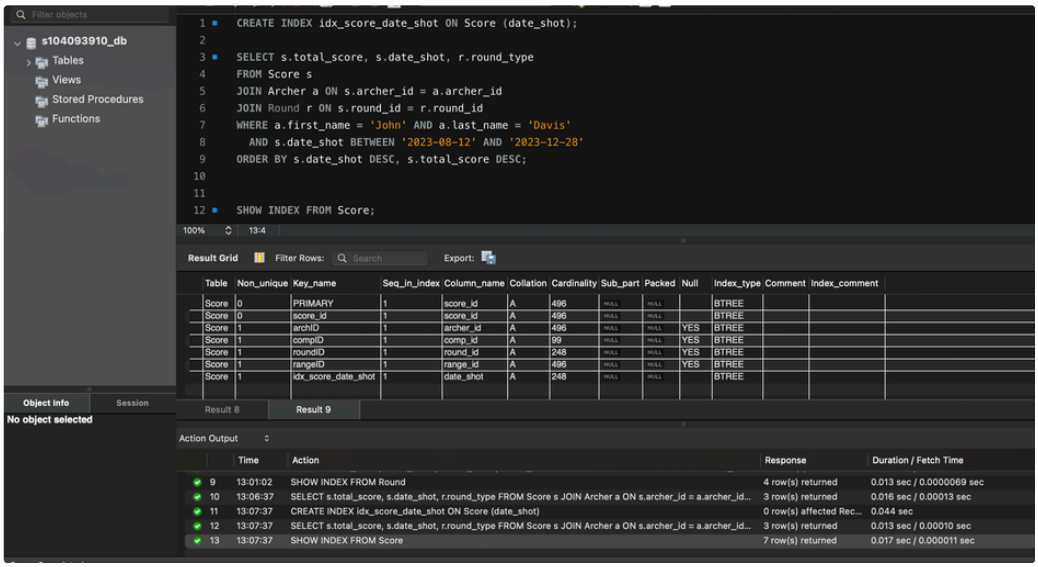


Notice the difference in the Duration/Fetch time (action output 6 and 8) for running the query before and after indexing.

date_shot

Most people filter results based on the date of the event, hence, indexing the date_shot column of the Score and the StagingScore table is beneficial for the overall performance of the database.

```
CREATE INDEX idx_score_date_shot ON Score (date_shot);
```

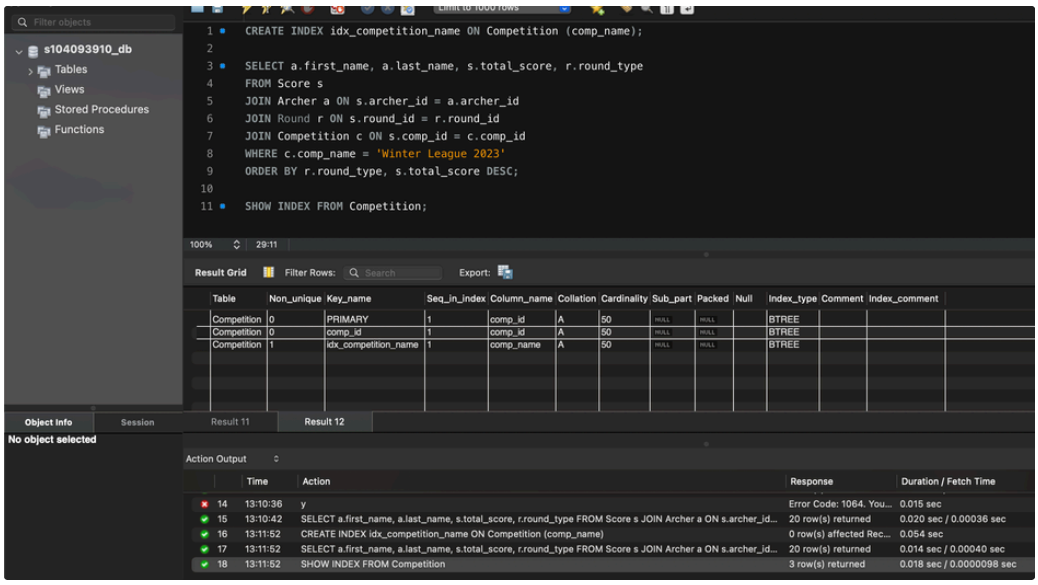


Notice the difference in the Duration/Fetch time (action output 10 and 12) for running the query before and after indexing.

comp_name

So practically, users would filter scores and results of a particular competition that they are watching and the comp_name column is looked into a lot for these queries. Hence, indexing this column is very helpful.

```
CREATE INDEX idx_competition_name ON Competition (comp_name);
```

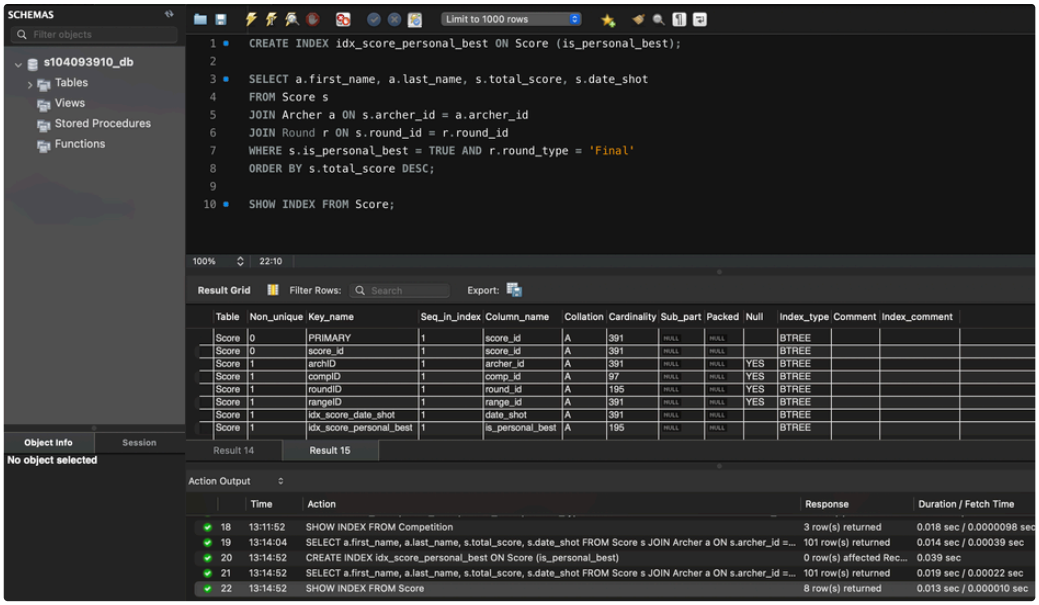


Notice the difference in the Duration/Fetch time (action output 15 and 17) for running the query before and after indexing.

is_personal_best

Ever so often an Archer might want to look up his own personal best Score or a general user is looking up personal best scores for archers in a particular competition, we use the is_personal_best column of the Score table to get the personal best score of the Archer, given that it exist.

```
CREATE INDEX idx_score_personal_best ON Score (is_personal_best);
```

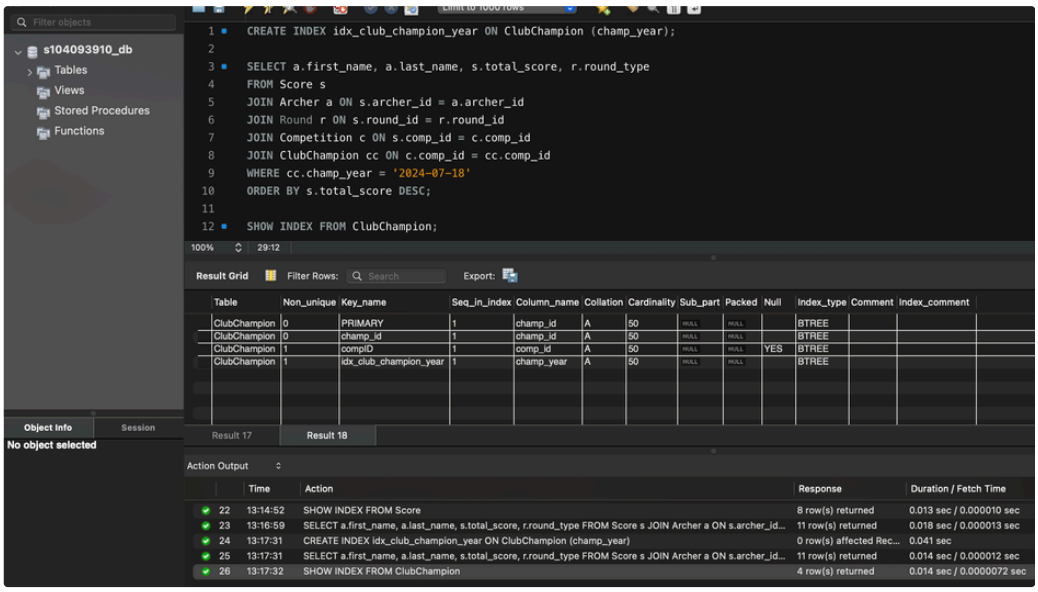


Notice the difference in the Fetch time (action output 19 and 21) for running the query before and after indexing.

champ_year

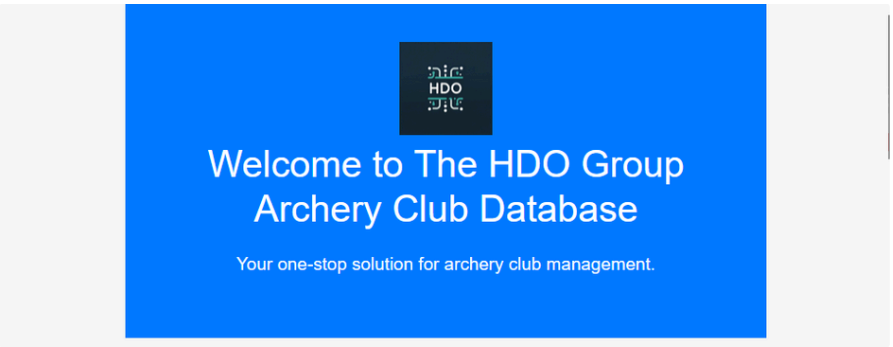
To get the results for a championship the user or the Archer might want to filter the championships by the year. So, we can index the champ_year column of the ClubChampion table.

```
CREATE INDEX idx_club_champion_year ON ClubChampion (champ_year);
```



Notice the difference in the Duration/Fetch time (action output 23 and 25) for running the query before and after indexing.

Major Specific Work (UX/UI)



For the major specific implementation of our project we built a comprehensive web application for managing the HDO Archery Club database using . This was led by @Upek Isuranga , as he is doing Games and Software Development as his major, and @Nikhil mohite who is well versed in web development as well. Another thing that we added from Cybersecurity perspective was a login system and a minor attempt at avoiding SQL Injection attacks, which was led by @ARSH KHANNA , @Pulkit and @MelvinGoxhaj .

We used Vue.js, Bootstrap and Node.js to develop our web app. It was our first time using such frameworks however throughout the process we learnt a lot. Here is an overview of the process and the implementation details for each feature.

Technology Stack:

- Vue.js: Used to replace majority of javascript code with its easier-to-read template design. Helped implement many reactive components of our website with ease such as buttons.
- Bootstrap: Implemented using a CDN link helped style the webpage quickly with a modern design and ensure features of the site were responsive.
- Node.js: Used to install packages and create a server-side for handling API ‘fetch’ requests when the user interacts with the database.
- MySQL Workbench & PHPmyAdmin: Database management system used to create relevant table and insert/fetch data from.

Login & User Authentication:

Login fields for both Archer and ScoreRecorder

- Currently, our login will sign in for both Archers & scoreRecorders using a predefined username and password fetched from the Users Table created in the backend.
  - Archer - Username ('archer') Password ('password')
  - Score Recorder - Username ('scoreRecorder') Password ('password1')

Respective welcome messages for both the Archer and Score Recorder

- The login function sends a POST request to the server which will validate the credentials.
- The check ‘checkSession’ method will check if the user is already logged in if they happen to refresh the page and also display the welcome message with their respective ‘role’ which is retrieved from the ‘role’ column in the user table.

```php
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $username = $_POST['username'];
    $password = $_POST['password'];

    $sql = "SELECT role, archer_id, score_recorder_id FROM users WHERE username = ? AND password = ?";
```

Example: Using POST request in login.php

```php
<?php
session_start();
header('Content-Type: application/json');
$response = [

    'username' => isset($_SESSION['username']) ? $_SESSION['username'] : '',
    'role' => isset($_SESSION['role']) ? $_SESSION['role'] : '',

    'archer_id' => isset($_SESSION['archer_id']) ? $_SESSION['archer_id'] : ''
];
echo json_encode($response);
?>
```

Check_session.php

Preventing SQL Injections

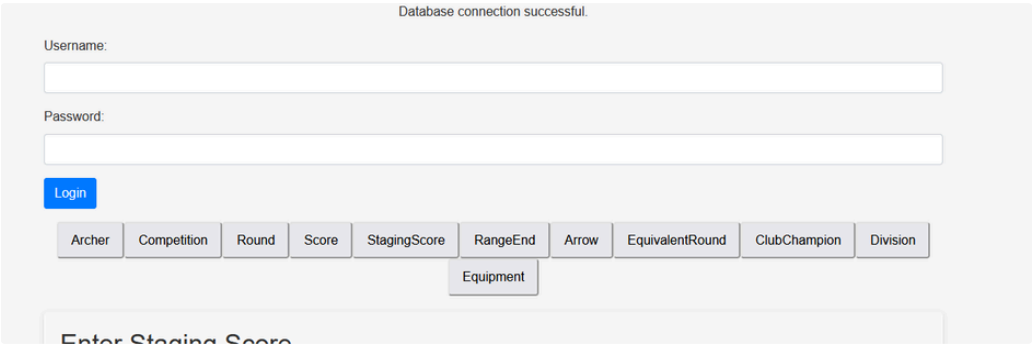
We created a basic PHP function which sanitises all the user inputs from any special characters they could use to access the database.

```php
function sanitise_input($data){//sanitises user input
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data;
}
```

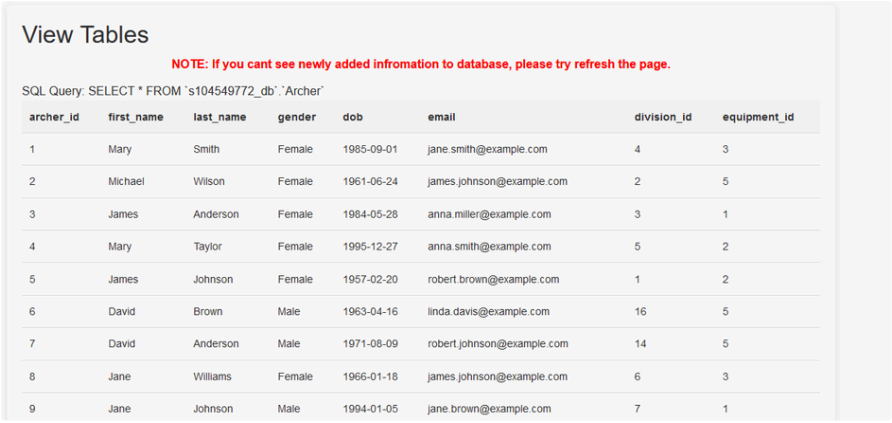
Sanitise input function

For every data the user enters is passed through this function and then to the database.

View Database Tables



View table buttons



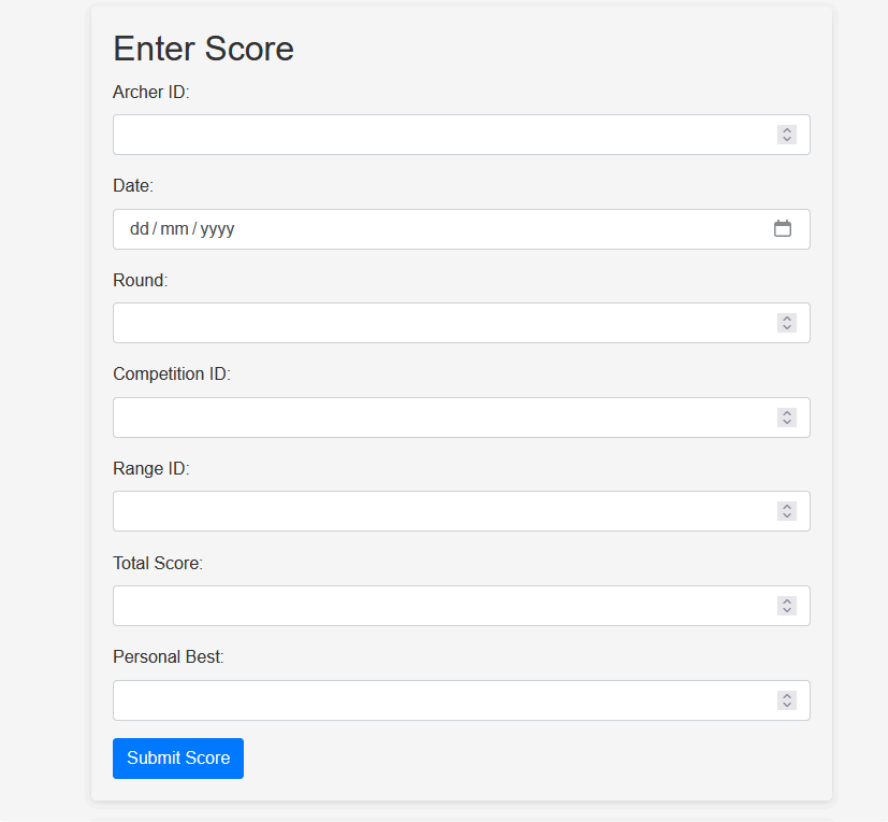
Tables retrieved from the database are displayed at the bottom of the page example: Archer table.

- Users can view all our tables simply by selecting a table name. The ‘showTable’ method will then effectively fetch and display the contents of the table at the bottom of the webpage.
- Vuejs made it extremely easy to insert messages that will warn if there is an issue in retrieving the database information displaying error messages.

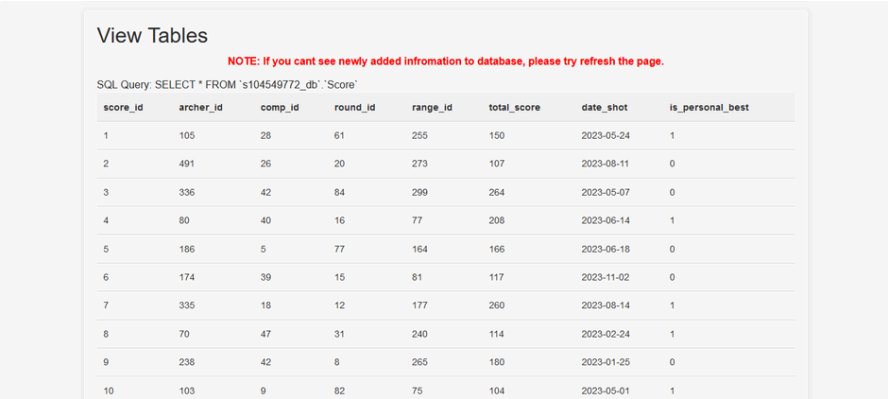
```php
// sure the table name is valid and exists in the database
$sql = "SHOW TABLES LIKE '$tableName'";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) == 0) {
    echo "Table '$tableName' does not exist."; //erorr if tables arent present
    exit;
}
Ctrl+L to chat, Ctrl+K to generate
$sql = "SELECT * FROM `$sql_db`.`$tableName`";
echo "SQL Query: $sql<br>"; //

$result = mysqli_query($conn, $sql);
if (!$result) {
    echo "Error fetching table data: " . mysqli_error($conn); //error message
    exit;
}
```

code snipped view_table.php

Enter Scores



Enter Scores forum



View Tables

NOTE: If you cant see newly added infromation to database, please try refresh the page.

SQL Query: SELECT * FROM `s104549772_db`.`Score`

score_id	archer_id	comp_id	round_id	range_id	total_score	date_shot	is_personal_best
1	105	28	61	255	150	2023-05-24	1
2	491	26	20	273	107	2023-08-11	0
3	336	42	84	299	264	2023-05-07	0
4	80	40	16	77	208	2023-06-14	1
5	186	5	77	164	166	2023-06-18	0
6	174	39	15	81	117	2023-11-02	0
7	335	18	12	177	260	2023-08-14	1
8	70	47	31	240	114	2023-02-24	1
9	238	42	8	265	180	2023-01-25	0
10	103	9	82	75	104	2023-05-01	1

Scores Table

- Enter Score enables scoreRecorder to enter scores directly into the ‘score’ table using a form. The ‘enterScore’ method will then append the form data.

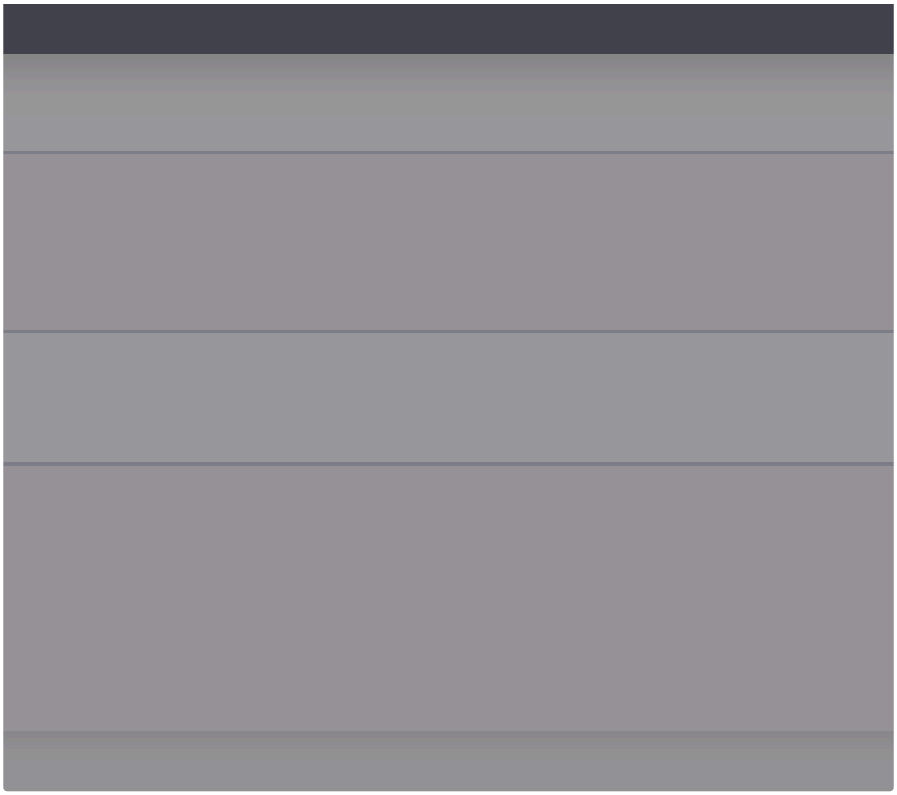
```php
$archer_id = $_POST['archer_id'];
$date = $_POST['date'];
$round = $_POST['round'];
$comp_id = $_POST['comp_id'];
$range_id = $_POST['range_id'];
$total_score = $_POST['total_score'];
$is_personal_best = $_POST['is_personal_best'];

$sql = "INSERT INTO Score (archer_id, date_shot, round_id, comp_id, range_id, total_score, is_personal_best)
        VALUES ('$archer_id', '$date', '$round', '$comp_id', '$range_id', '$total_score', '$is_personal_best')";
```

Enter scores code snippet from enter_score.php

```javascript
    enterScore() {
        const formData = new FormData();
        formData.append('archer_id', this.newScore.archer_id);
        formData.append('date', this.newScore.date);
        formData.append('round', this.newScore.round);
        formData.append('comp_id', this.newScore.comp_id);
        formData.append('range_id', this.newScore.range_id);
        formData.append('total_score', this.newScore.total_score);
        formData.append('is_personal_best', this.newScore.is_personal_best);

        fetch('enter_score.php', {
            method: 'POST',
            body: formData
        })
        .then(response => response.text())
        .then(alert);
    },
```

Fetching and appending respective fields located in app.js

Staging Score



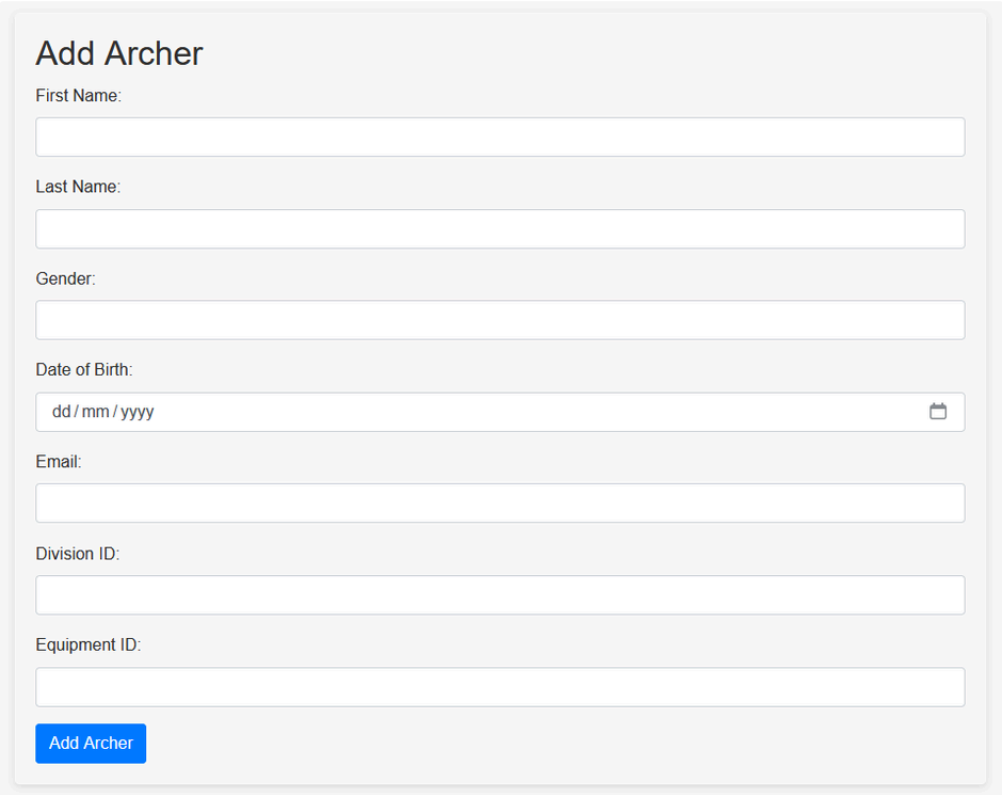
Staging Score Fourm

- The staging score forum feature allows archer users to enter their score in a temporary table called the Staging Score Table, this temporary storage allows scores to be reviewed and validated before they are transferred to the main ‘Score’ table by the Score Recorder.
- The ‘enterStagingScore' method in the Vue.js application collects the form data when the user submits the form to enter a new staging score. It then uses the Fetch APi to send this data to ‘enter_staging_score.php’ via a POST request.
- Once received the server will respond if successful displaying an alert message similar to all the other forums.

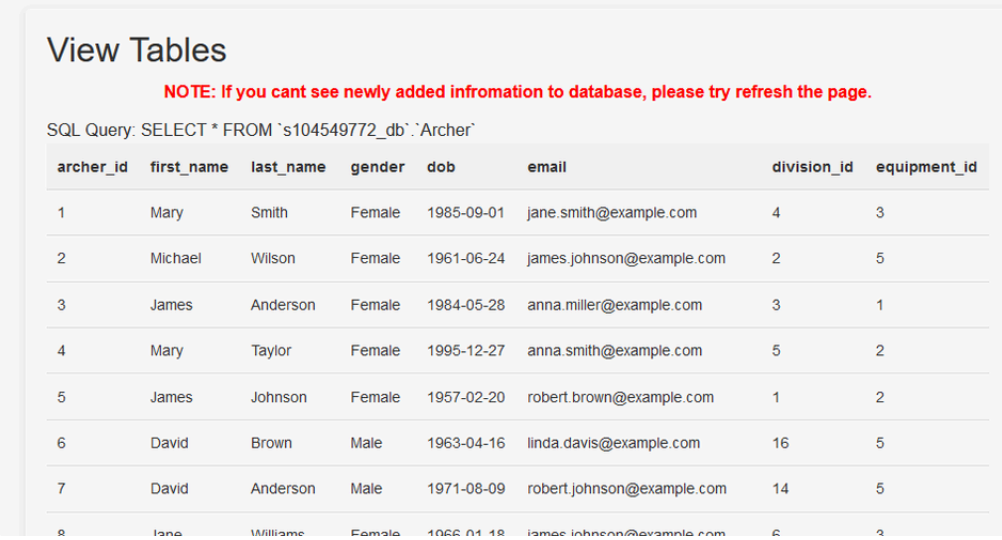
```php
<?php
session_start();
require("settings.php");
$conn = @mysqli_connect($host, $user, $pwd, $sql_db);

if (
    !isset($_POST['archer_id']) ||
    !isset($_POST['date']) ||
    !isset($_POST['round']) ||
    !isset($_POST['comp_id']) ||
    !isset($_POST['range_id']) ||
    !isset($_POST['total_score']) ||
    !isset($_POST['is_personal_best'])
) {
    echo "All fields are required.";
    exit;
}

$archer_id = $_POST['archer_id'];
$date = $_POST['date'];
$round = $_POST['round'];
$comp_id = $_POST['comp_id'];
$range_id = $_POST['range_id'];
$total_score = $_POST['total_score'];
$is_personal_best = $_POST['is_personal_best'];
```

code snippet located in enter_staging_score.php

Add Archer



Add archer forum



archer_id	first_name	last_name	gender	dob	email	division_id	equipment_id
1	Mary	Smith	Female	1985-09-01	jane.smith@example.com	4	3
2	Michael	Wilson	Female	1961-06-24	james.johnson@example.com	2	5
3	James	Anderson	Female	1984-05-28	anna.miller@example.com	3	1
4	Mary	Taylor	Female	1995-12-27	anna.smith@example.com	5	2
5	James	Johnson	Female	1957-02-20	robert.brown@example.com	1	2
6	David	Brown	Male	1963-04-16	linda.davis@example.com	16	5
7	David	Anderson	Male	1971-08-09	robert.johnson@example.com	14	5

Archer Table

- Users can add new archers to the ‘Archer’ table. The insertArcher function $POSTs relevant fields and inserts them into the database.
- The ‘ssssssssi’ part located in the code snippet is used to format strings used in the ‘bind_param’ method to prepare the statement in MYSQLI. It indicates the types of parameters being bound for instance.
  - ‘s' stands for ‘string’ (total of 6 's’ indicating first 6 parameters are strings)
  - ‘i' stands for ‘integer’ (one ‘i’ for integer)
- The ‘addArcher' method collects the form data to be sent to the server as seen in the vue.js code snippet below.
- Respective success and error messages were implemented.

```php
// to insert a new archer
function insertArcher($conn) {
    $firstName = $_POST['firstName'];
    $lastName = $_POST['lastName'];
    $gender = $_POST['gender'];
    $dob = $_POST['dob'];
    $email = $_POST['email'];
    $division_id = $_POST['division_id'];
    $equipment_id = $_POST['equipment_id'];

    $sql = "INSERT INTO Archer (first_name, last_name, gender, dob, email, division_id, equipment_id)
            VALUES (?, ?, ?, ?, ?, ?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("sssssss", $firstName, $lastName, $gender, $dob, $email, $division_id, $equipment_id);
    $stmt->execute();

    if ($stmt->affected_rows > 0) {
        echo "New archer added successfully.";
    } else {
        echo "Error adding archer: " . $conn->error;
    }

    $stmt->close();
}
```

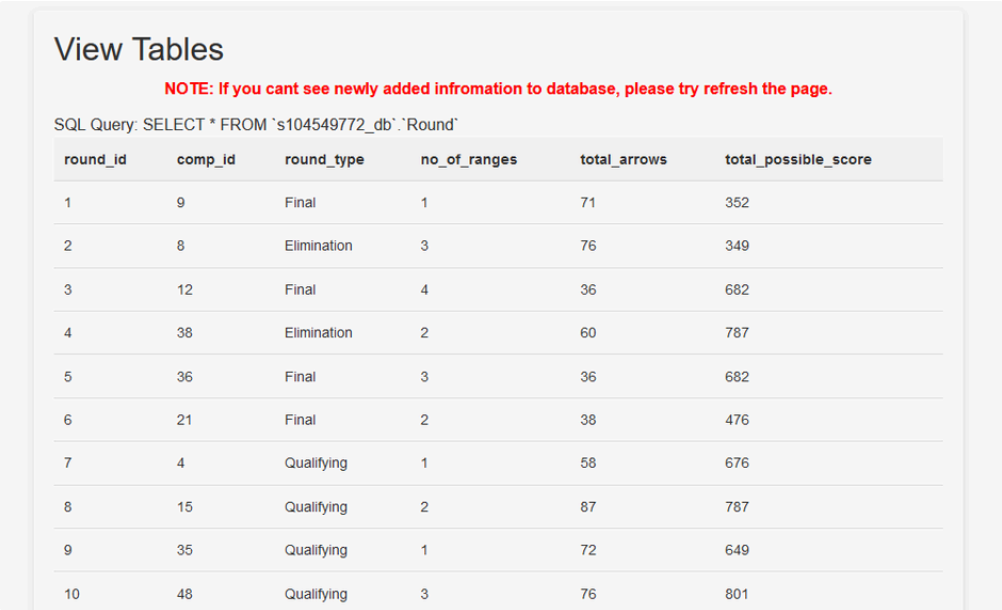
Code snippet adding archer located in insert_data.php

```javascript
addArcher() {
    const formData = new FormData(document.getElementById('addArcherForm'));
    formData.append('form_type', 'archer');

    fetch('insert_data.php', {
        method: 'POST',
        body: formData
    })
    .then(response => response.text())
    .then(alert)
    .catch(error => console.error('Error:', error));
},
```

Add Archer Vue.js located in App.js

Add Round

Add Round forum



View Tables

NOTE: If you cant see newly added infromation to database, please try refresh the page.

SQL Query: SELECT * FROM `s104549772_db`.`Round`

round_id	comp_id	round_type	no_of_ranges	total_arrows	total_possible_score
1	9	Final	1	71	352
2	8	Elimination	3	76	349
3	12	Final	4	36	682
4	38	Elimination	2	60	787
5	36	Final	3	36	682
6	21	Final	2	38	476
7	4	Qualifying	1	58	676
8	15	Qualifying	2	87	787
9	35	Qualifying	1	72	649
10	48	Qualifying	3	76	801

Round database table

- Users can add new rounds to the ‘Round' database table. The ‘addRound’ collects the data and send it to the server to be inserted into the database.
- ‘isiii’ is used to prepare the statements for insertion into the database.
- ‘new round added successfully’ & error messages were added.

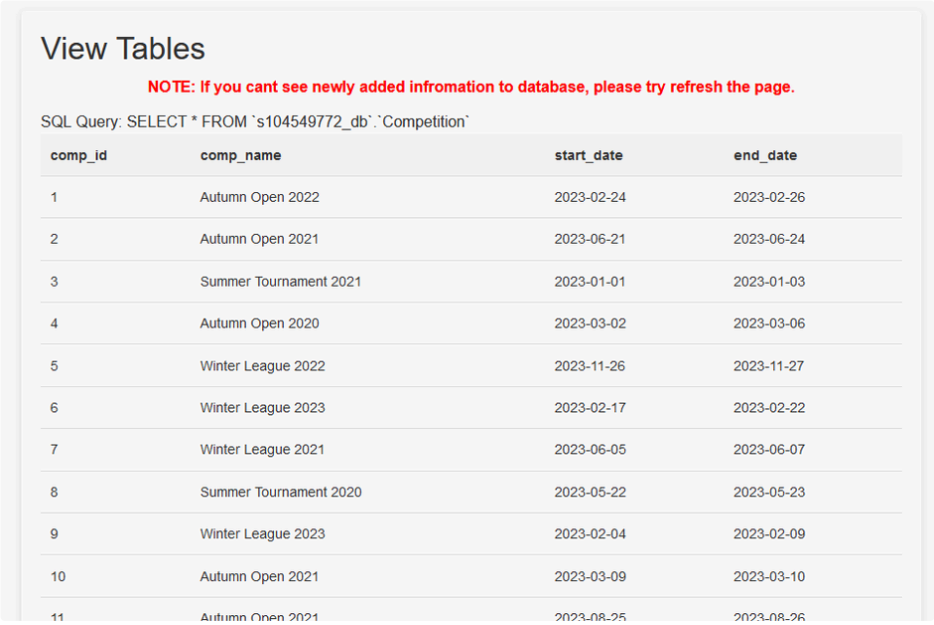
```php
// function put in a newnew round
function insertRound($conn) {
    $comp_id = $_POST['comp_id'];
    $round_type = $_POST['round_type'];
    $no_of_ranges = $_POST['no_of_ranges'];
    $total_arrows = $_POST['total_arrows'];
    $total_possible_score = $_POST['total_possible_score'];

    $sql = "INSERT INTO Round (comp_id, round_type, no_of_ranges, total_arrows, total_possible_score)
            VALUES (?, ?, ?, ?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("isiii", $comp_id, $round_type, $no_of_ranges, $total_arrows, $total_possible_score);
    $stmt->execute();

    if ($stmt->affected_rows > 0) {
        echo "New round added successfully.";
    } else {
        echo "Error adding round: " . $conn->error;
    }

    $stmt->close();
}
```

code snippet of insert round located in insert_data.php

```javascript
addRound() {
    const formData = new FormData(document.getElementById('addRoundForm'));
    formData.append('form_type', 'round');

    fetch('insert_data.php', {
        method: 'POST',
        body: formData
    })
    .then(response => response.text())
    .then(alert)
    .catch(error => console.error('Error:', error));
},
```

Vue.js script located in app.js

Add Competition



View Tables

NOTE: If you cant see newly added infromation to database, please try refresh the page.

SQL Query: SELECT * FROM `s104549772_db`.`Competition`

comp_id	comp_name	start_date	end_date
1	Autumn Open 2022	2023-02-24	2023-02-26
2	Autumn Open 2021	2023-06-21	2023-06-24
3	Summer Tournament 2021	2023-01-01	2023-01-03
4	Autumn Open 2020	2023-03-02	2023-03-06
5	Winter League 2022	2023-11-26	2023-11-27
6	Winter League 2023	2023-02-17	2023-02-22
7	Winter League 2021	2023-06-05	2023-06-07
8	Summer Tournament 2020	2023-05-22	2023-05-23
9	Winter League 2023	2023-02-04	2023-02-09
10	Autumn Open 2021	2023-03-09	2023-03-10
11	Autumn Open 2021	2023-08-25	2023-08-26

- Add competitions effectively allows score recorders to add new competitions to the ‘competition’ table.
- The ‘addCompetition’ method helps collects the data and inserts it into the database.
- Fields are then $POST as seen in the code snippet.
- error messages were added aswell

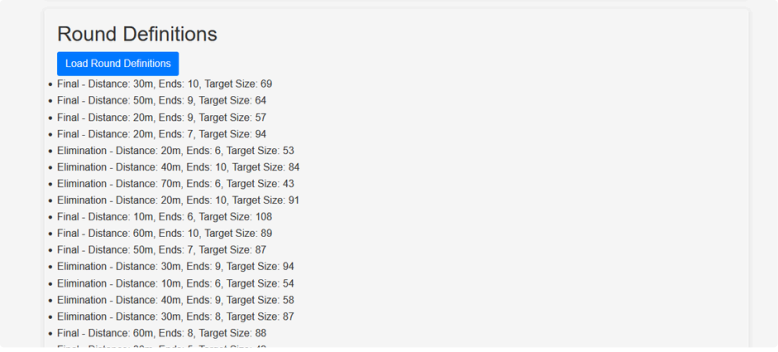
```php
// to insert a new competition
function insertCompetition($conn) {
    $comp_name = $_POST['comp_name'];
    $start_date = $_POST['start_date'];
    $end_date = $_POST['end_date'];

    $sql = "INSERT INTO Competition (comp_name, start_date, end_date)
            VALUES (?, ?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("sss", $comp_name, $start_date, $end_date);
    $stmt->execute();

    if ($stmt->affected_rows > 0) {
        echo "New competition added successfully.";
    } else {
        echo "Error adding competition: " . $conn->error;
    }

    $stmt->close();
}
```

Code snippet for insert competition located in insert_data.php

```
addCompetition() {
    const formData = new FormData(document.getElementById('addCompetitionForm'));
    formData.append('form_type', 'competition');

    fetch('insert_data.php', {
        method: 'POST',
        body: formData
    })
    .then(response => response.text())
    .then(alert)
    .catch(error => console.error('Error:', error));
},
```

Vue.js code snippet is located in app.js

Round Definitions



- Users can view definitions of different rounds, including details about target distance, end count and target size.
- The ‘getRoundDefinitions’ method in app.js sends a request to the server to fetch the round definitions, and the response is displayed on the webpage
- The button used to display the round definitions is also fixed to hide the list of round definitions.
- Unit test 6 can be seen being used in round_definitions.php to fetch the definitions, verifies that round details, along with ensuring their corresponding ranges are retrieved.

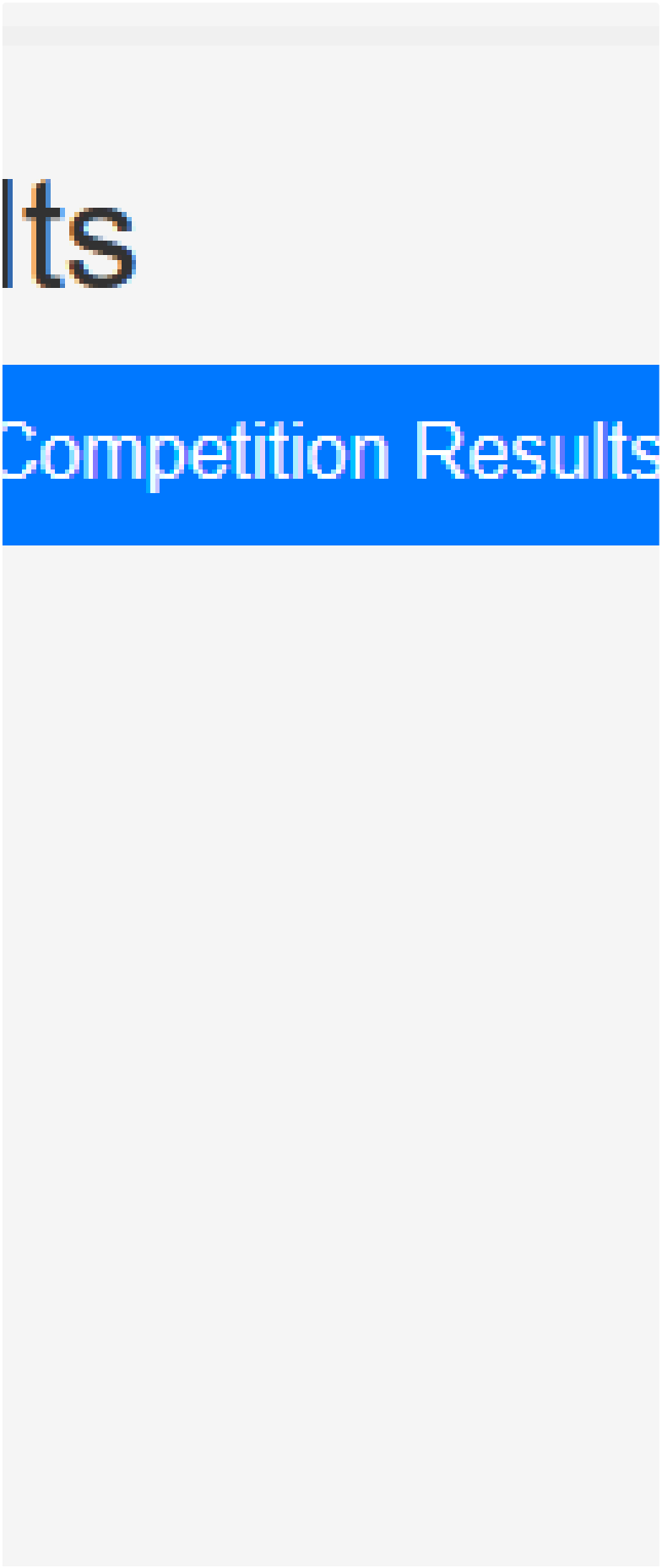
```
<?php
require("settings.php");
$conn = @mysqli_connect($host, $user, $pwd, $sql_db);

$sql = "SELECT r.round_type, ra.target_distance, ra.end_count, ra.target_size
        FROM Round r
        JOIN `Range` ra ON r.round_id = ra.round_id";

$result = mysqli_query($conn, $sql);
$rounds = [];

while ($row = mysqli_fetch_assoc($result)) {
    $rounds[] = $row;
}

echo json_encode($rounds);
?>
```

Code snipped located in round_definitions.php

```
getRoundDefinitions() {
    if (!this.showRoundDefinitions) {
        fetch('round_definitions.php')
            .then(response => response.json())
            .then(data => {
                this.roundDefinitions = data;
                this.showRoundDefinitions = true;
            });
    } else {
        this.showRoundDefinitions = false;
    }
},
```

getRoundDefinitions method located in app.js

Competition Results



Loading Competition Results

- Users can retrieve and view competition results by entering the competition name.
- As seen in competition_results.php unit test 5 has been used to retrieve and view respective competition results.

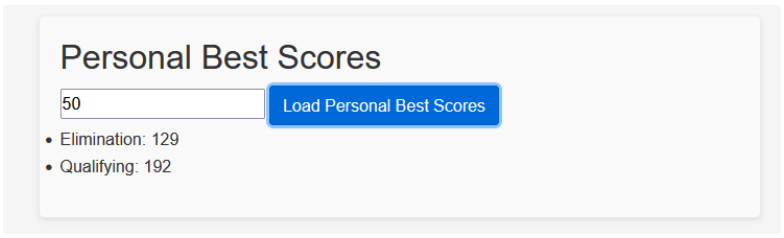
```php
<?php
require("settings.php");
$conn = @mysqli_connect($host, $user, $pwd, $sql_db);

$comp_name = $_GET['comp_name'];

$sql = "SELECT a.first_name, a.last_name, s.total_score, r.round_type
        FROM Score s
        JOIN Archer a ON s.archer_id = a.archer_id
        JOIN Round r ON s.round_id = r.round_id
        JOIN Competition c ON s.comp_id = c.comp_id
        WHERE c.comp_name = '$comp_name'
        ORDER BY s.total_score DESC";

$result = mysqli_query($conn, $sql);
$competitionResults = [];

while ($row = mysqli_fetch_assoc($result)) {
    $competitionResults[] = $row;
}

echo json_encode($competitionResults);
```

code snippet located in competition_results.php

```javascript
getCompetitionResults(compName) {
    fetch(`competition_results.php?comp_name=${compName}`)
        .then(response => response.json())
        .then(data => {
            this.competitionResults = data;
        });
},
```

fetching competition results located in app.js

Personal Best Scores



Example archer ID (50) loading personal best scores

- The ‘getPersonalBestScores’ method in Vue.js sends a request to ‘erson_best.php’ with the archer ID.
- PHP script will then retrieve the personal best score from the database and return them as JSON responses to be displayed on the webpage.

- Best scores for respective round types will then be listed below.

```php
<?php
require("settings.php");
$conn = @mysqli_connect($host, $user, $pwd, $sql_db);

$archer_id = $_GET['archer_id'];

$sql = "SELECT r.round_type, MAX(s.total_score) AS personal_best_score
        FROM Score s
        JOIN Round r ON s.round_id = r.round_id

        WHERE s.archer_id = '$archer_id'
        GROUP BY r.round_type";

$result = mysqli_query($conn, $sql);

$personalBestScores = [];

while ($row = mysqli_fetch_assoc($result)) {
    $personalBestScores[] = $row;
}

echo json_encode($personalBestScores);
?>
```

Code snippet is located in personal_best.php

```javascript
            getPersonalBestScores(archerId) {
                fetch(`personal_best.php?archer_id=${archerId}`)
                    .then(response => response.json())
                    .then(data => {
                        this.personalBestScores = data;
                    });
            }
        }
    });
```

Code snippet is located in app.js

Enhancements

- Couple of enhancements were made using both CSS and Bootstrap to improve the user experience. The most notable enhancements were ones we had used in past projects, including:
  - **Hover effects:** Certain elements of the page such as the HDO logo have hover effects added using @Keyframes.
  - **Fade In effects:** For our front page title, upon opening the page the opacity will be adjusted such that our welcome message will fade in. This is also implemented using @Keyframes
  - **Box shadows:** In order to improve the look of each forum section a small border has been added around it to give a shadow effect. This enhancement is minimal however impactful as the page looks more modern as each section looks like it is 3-D, popping out of the page.

```css
@keyframes hoverUpDown {
  0%, 100% { transform: translateY(0); }
  50% { transform: translateY(-10px); }
}
```

Hover effect using @Keyframes

```css
@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}
```

Fade In effect for our title using @Keyframes

```css
.section {
  background-color: #f9f9f9;
  padding: 20px;
  border-radius: 5px;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
  margin-bottom: 20px;
}
```

Shadow effect in CSS to give the forums the appearance of depth in our webpage

Team reflection

📋 Overview

Reflect back on what you and your team learned and what motivates the group to succeed by following the instructions for the 4Ls Retrospective Play.

Team	High Distinction Only
Team members	@ARSH KHANNA @Upek Isuranga @MelvinGoxhaj @Pulkit @Nikhil mohite
Date	25/05/2024
Retrospective period	March 26 - May 26

💭 4Ls retrospective

Milestones	Loved	Longed for	Loathed	Learned
Completed the project phases on time	• Collaboration with the team was great and everyone contributed to the project. • Team meetings were essential to our progress and helped to share our ideas with good chemistry among peers.	• More time for an in-depth analysis of our work	• When we had difficulties with some team members reaching deadlines	• We improved our time management skills • We got to understand our team members better and what our weaknesses were.
Got a front-end website up and running	• Team members equally contributed to working on the website and made it practical.	• A more visually appealing website for the Archers.	• Figuring out which SQL commands will work and run on the website was tricky. • Intensive coding was required to get the database running and so a lot of time was allocated to making this happen.	• Learned about how back-end development worked on a website and how SQL works alongside it.
Finished up project report on Confluence	• Completing the project report gave a strong sense of accomplishment and highlighted the team's hard work • The team worked together to update the Confluence pages and created a coherent report	• Additional time to reflect on the project's successes and challenges in greater detail would have been good for the team.	• The extensive amount of documentation required was overwhelming.	• We understood project management and how important it is to document our journey. • we understood each team members strengths and weaknesses which helped to assign a task to a member.

⚡Action plan

Action	Owner	Due date	Action items
Improve Time Management	All Members	Ongoing	Make us of project management tools and techniques
Enhance Communication	All Members	Ongoing	Have regular check-ins and clear assignment of tasks
Reduce Meeting Frequency	@Pulkit	Ongoing	Create meetings and focus only on key discussions to save time

📄Project Overview: Archery Club Database Management System

Purpose: Our Archery Club Database Management System aims to streamline the administrative processes of a local archery club by providing a centralized platform for storing, managing, and accessing multiple archer data. By using databases like MySQL, the system hosts efficient data management and enhances communication among club members.

Key Features:

- **Efficient Data Storage:** Using MySQL as the database system ensures structured storage of data, enabling efficient retrieval and editing of information.
- **Data Integrity and Security:** Robust mechanisms are implemented to maintain data integrity and ensure the security of sensitive club data, including user authentication and authorization protocols.
- **User-friendly Interface:** A website provides an intuitive front end to the club data, allowing members to track and view their scores.
- **Collaborative Tools Integration:** Integration with collaborative platforms such as Confluence and Jira enhances team collaboration, facilitates project planning, and enables seamless workflow management.
- **Mobile Responsiveness:** The web interface is designed to be responsive and adaptive to mobile devices which enhances accessibility to a wide range of users.
- **Backup Recovery:** Automated backup mechanisms are implemented to safeguard against data loss, ensuring continuity of club operations even in the event of system failures.

The Archery Club Database Management System offers the following benefits:

- **Efficiency:** Streamlines administrative tasks, saving time and resources for club administrators.
- **Transparency:** Provides visibility into club activities, scores, and project progress, giving transparency and accountability.
- **Collaboration:** Facilitates collaboration among club members which enables effective communication and coordination of club events and initiatives.
- **Data-driven Decisions:** Empowers club leadership to make informed decisions based on accurate and up-to-date data insights.
- **Scalability:** Designed to scale with the club's growth and evolving needs, ensuring long-term viability and sustainability.

Conclusion: The Archery Club Database Management System represents a significant advancement in enhancing the efficiency, communication, and data collecting capabilities of our local archery club. By using modern database technologies and collaborative tools, the system empowers club members to achieve their goals and fulfill the club's mission of promoting the sport of archery in our community.

List

Give feedback

Search list

Share Filter Group More

Type	Key	Summary	Status	Sprint	Assignee	Due date	Labels	Created	
☑	ACDHO-1	Implement RDMS with Mercury back-end	DONE	ACDHO Sprin...	MelvinGoxhaj			Mar 6, 2024	Apr 10, 2
☑	ACDHO-4	Following realational principles and ACID, ensure database is...	DONE	ACDHO Sprin...	MelvinGoxhaj			Mar 6, 2024	Apr 10, 2
☑	ACDHO-5	User Access Levels and Authentication	DONE	ACDHO Sprin...	Pulkit			Mar 6, 2024	May 26, 2
☑	ACDHO-2	Implement Archery Club Frontend site	DONE	ACDHO Sprin...	Nikhil mohite			Mar 6, 2024	May 6, 20
☑	ACDHO-13	Create dynamic database php creation file, remove hardcode...	DONE	ACDHO Sprin...	MelvinGoxhaj			Apr 20, 2024	May 1, 20
☑	ACDHO-14	Fix the Entity Relationship Diagram	DONE	ACDHO Sprin...	Pulkit			Apr 20, 2024	May 1, 20
☑	ACDHO-15	Create Entity Relationship Diagram	DONE	ACDHO Sprin...	ARSH KHANNA			Apr 20, 2024	Apr 20, 2
☑	ACDHO-16	Create Empathy Map	DONE	ACDHO Sprin...	Nikhil mohite			Apr 20, 2024	Apr 20, 2
☑	ACDHO-17	Make a project plan	DONE	ACDHO Sprin...	Pulkit			Apr 21, 2024	Apr 21, 2
■	ACDHO-19	Fix the PHP file for the database	DONE	ACDHO Sprin...	Upek Isuranga			Apr 21, 2024	May 6, 20
●	ACDHO-18	Fix Primary Key and Foreign Keys in the database	DONE	ACDHO Sprin...	ARSH KHANNA			Apr 21, 2024	May 6, 20
☑	ACDHO-3	Unit test SQL queries and front-end code	DONE	ACDHO Sprin...	ARSH KHANNA			Mar 6, 2024	May 25, 2
■	ACDHO-20	populate tables with test data	DONE	ACDHO Sprin...	MelvinGoxhaj			May 6, 2024	May 26, 2

+ Create