

|  |
| --- |
| **CLIMBS OF the MOUNTAINEERING CLUB "Climbs in seconds"** |
|  |



Contents

[1 Business Description 3](#_Toc62212630)

[1.1 Business background 3](#_Toc62212631)

[1.2 Problems. Current Situation 3](#_Toc62212632)

[1.3 The benefits of implementing a database. Project Vision 3](#_Toc62212633)

[2 Model description 3](#_Toc62212634)

[2.1 Definitions & Acronyms 3](#_Toc62212635)

[2.2 Logical Scheme 3](#_Toc62212636)

[2.3 Objects 3](#_Toc62212637)

# 

# Business Description

## Business background

"Climbs in Seconds" is a mountain climbing club that organizes regular climbing expeditions. The club manages its own equipment, routes, sponsors, and climbers. The club’s mission is to provide a safe and structured environment for climbing enthusiasts, from beginners to experienced climbers.

Key Features:

Sponsors: The club receives financial support from sponsors. Sponsors can choose to donate funds directly to individual climbers or to the club itself.

Climbing Routes: A single climb may consist of one or more climbing routes, depending on the complexity and duration of the climb.

Climbers: The club hosts climbers with various levels of experience, all of whom rely on the club's equipment and routes to participate in organized climbs.

The club aims to grow its community and improve its operations by implementing better data management and organizational practices, ensuring smooth coordination of climbs and effective use of resources.

## Problems. Current Situation

Currently, the club "Climbs in Seconds" faces several challenges in managing its operations effectively:

Manual record-keeping: Information about climbers, routes, equipment, and sponsors is maintained manually, leading to errors and inefficiencies.

Difficulty in tracking sponsorships: It’s hard to track which sponsors are supporting individual climbers versus the club as a whole.

Lack of integration: Data related to climbs, routes, and equipment is not centralized, making it difficult to plan expeditions and maintain equipment usage records.

Limited reporting capabilities: Without a proper system, the club struggles to generate reports for decision-making, such as tracking finances or analyzing popular routes.

## the Benefits of implementing a database. Project Vision

The implementation of a database system will significantly improve the club's efficiency and data management capabilities. Key benefits include:

Centralized data management: All information about climbers, sponsors, equipment, routes, and climbs will be stored in one place, reducing redundancy and errors.

Enhanced sponsorship tracking: The database will allow for easy tracking of sponsorships, showing exactly how funds are allocated to climbers and the club.

Improved planning and resource management: With integrated data, the club can better plan climbs, manage routes, and track the usage and maintenance of equipment.

Better reporting and insights: The club will be able to generate detailed reports on various aspects, from financials to climber performance, helping with decision-making and future planning.

Scalability and growth: As the club grows, the database can easily scale to handle more data, ensuring long-term sustainability.

# Model description

## Definitions & Acronyms

Club: Refers to "Climbs in Seconds," the mountain climbing club responsible for organizing expeditions, managing climbers, routes, equipment, and sponsors.

Climb: A single event or expedition organized by the club, which may involve one or more climbing routes.

Route: A specific path or trail that is part of a climb. A climb can consist of multiple routes.

Climber: A person who participates in the club’s climbing activities.

Sponsor: An individual or organization that provides financial support to the club or individual climbers.

Equipment: The climbing gear owned and managed by the club, used by climbers during expeditions.

Database (DB): The system used to store, manage, and retrieve information about climbers, routes, climbs, sponsors, and equipment.

m:m

: Many-to-many relationship in the database, where multiple records from one table can be associated with multiple records from another table.

1:m (m:1)

: One-to-many (many-to-one) relationship in the database, where one record from a table is associated with many records from another table.

1:1

: One-to-one relationship in the database, where one record from a table is associated with one record from another table.

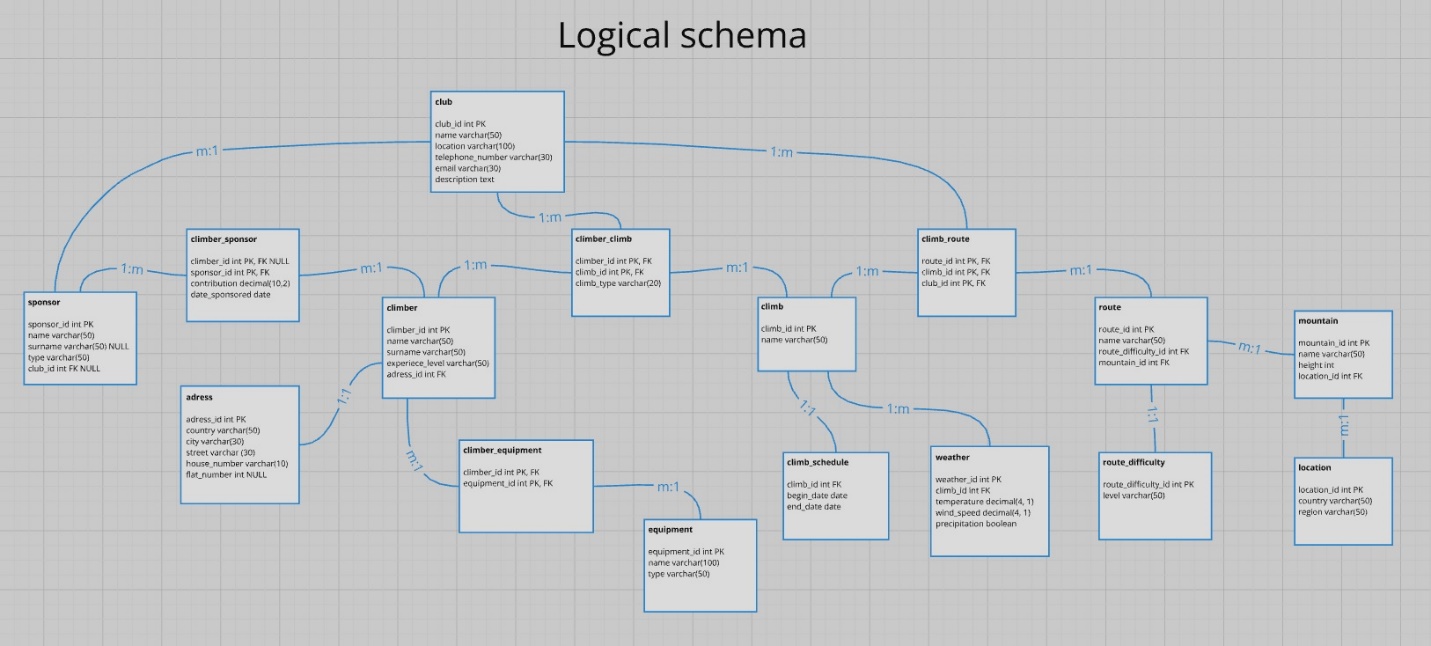
PK: Primary Key, a unique identifier for each record in a database table.

FK: Foreign Key, a reference to a Primary Key in another table to establish a relationship between two tables.

NULL: attribute in a table can either have a value or no value at all. Field is optional, and it’s not required to store a value for every record.

SQL: Structured Query Language, used for managing and querying data in the database.

## Logical Scheme



## Objects

1. **Sponsor table description**

This table stores information about the sponsors who donate to the club's, or climber’s activities.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| sponsor | sponsor\_id | unique identifier for each sponsor PK | int |
| name | sponsor first name | varchar (50) |
| surname | sponsor second name NULL (can be no value) | varchar (50) |
| type | indicates the category of the sponsor | varchar (50) |
| club\_id | foreign key referencing the club table FK | int |

Comments on table relationships

* sponsor ↔ climber: many-to-many (m:m) relationship, via climber\_sponsor;
* sponsor ↔ club: many-to-one (m:1) relationship, one club (because only one row in club table) can support multiple sponsors.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| sponsor\_id | name | surname | type |
| 1 | Microsoft | NULL | corporate |
| 2 | John | Canon | individual |
| 3 | National parks service | NULL | government |

1. **Climber\_sponsor table description**

This table is essential for tracking the financial support that climbers, or club receive from various sponsors.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climber\_sponsor | sponsor\_id | foreign key referencing the sponsor table FK PK (part of composite primary key) NULL (can be no value) | int |
| climber\_id | foreign key referencing the climber table FK PK (part of composite primary key) | int |
| club\_id | foreign key referencing the sponsor table the club table FK PK (part of composite primary key) NULL (can be no value) | int |
| contribution | the amount of money donated by the sponsor | decimal (10,2) |
| date\_sponsored | the date when the sponsorship was made | date |

Comments on table relationships

* climber\_sponsor connects climber and sponsor, indicating which sponsors support which climbers.
* this allows for a many-to-many relationship, as a climber can receive support from multiple sponsors, and a sponsor can support multiple climbers.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| sponsor\_id | climber\_id | club\_id | contribution | date\_sponsored |
| 1 | 1 | 1 | 500.00 | 2024-01-14 |
| 2 | 1 | 1 | 1000.00 | 2023-11-20 |
| 3 | 2 | 1 | 200.00 | 2024-03-13 |

1. **Climber table description**

This table stores information about the climbers who participate in the club's activities.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climber | climber\_id | unique identifier for each climber PK | int |
| name | climber first name | varchar (50) |
| surname | climber second name | varchar (50) |
| experience\_level | level of climbing experience | varchar (50) |
| address\_id | reference to the address table FK | int |

Comments on table relationships

* climber ↔ climb: many-to-many (m:m) relationship via climber\_climb table;
* climber ↔ club: many-to-many (m:m) relationship via climber\_climb table;
* climber ↔ sponsor: many-to-many (m:m) relationship via climber\_sponsor table;
* climber ↔ equipment: many-to-many (m:m) relationship via climber\_equipment table;
* climber ↔ adress: one-to-one (1:1) relationship, each address belongs to one climber.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| climber\_id | name | surname | experience\_level | address\_id |
| 1 | Jon | Jonovich | advanced | 3 |
| 2 | Laimis | Laimutis | intermediate | 2 |
| 3 | Oleg | Ars | beginner | 13 |

1. **Adress table description**

This table stores information about the climber adresses.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| adress | address\_id | unique identifier for each adress PK | int |
| country | climber country | varchar (50) |
| city | climber city | varchar (30) |
| street | climber street | varchar (30) |
| house\_number | climber house number | varchar (10) |
| flat\_number | climber flat number NULL (can be no values) | int |

Comments on table relationships

* adress ↔ climber: one-to-one (1:1) relationship, each address belongs to one climber.

Example with data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| adress\_id | country | city | street | house\_number | flat\_number |
| 1 | Lithuania | Vilnius | Tuskulenu | 41 | 4 |
| 2 | Poland | Krakow | Krakowiaka | 4 | NULL |
| 3 | Slovakia | Preshov | Liubocka | 1 | 10 |

1. **Climber\_equipment table description**

This table is essential for tracking the equipment that climbers use on each climb.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climber\_equipment | climber\_id | unique identifier for each climber FK, PK (part of composite primary key) | int |
| equipment\_id | unique identifier for each equipment FK, PK (part of composite primary key) | int |

Comments on table relationships

* climber\_equipment connects climber and equipment, indicating equipment used by climbers;
* this allows for a many-to-many relationship, as same equipment can be used by many climbers and climber can use separate equipment.

Example with data

|  |  |
| --- | --- |
| climber\_id | equipment\_id |
| 1 | 101 |
| 2 | 101 |
| 1 | 103 |

1. **Equipment table description**

This table stores information about the equipment.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| equipment | equipment\_id | unique identifier for each equipment PK | int |
| name | equipment name | varchar (100) |
| type | equipment type | varchar (50) |

Comments on table relationships

* climber ↔ equipment: many-to-many (m:m) relationship via climber\_equipment.

Example with data

|  |  |  |
| --- | --- | --- |
| equipment\_id | name | type |
| 101 | Petzl | rope |
| 102 | Black Diamond | helmet |
| 103 | Decathlon | shoes |

1. **Climber\_climb table description**

The climber\_climb table tracks the participation of climbers in specific climbs.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climber\_climb | climb\_id | foreign key referencing the climb table FK PK (part of composite primary key) | int |
| climber\_id | foreign key referencing the climber table FK PK (part of composite primary key) | int |
| climb\_type | type of climb | varchar (20) |

Comments on table relationships

* climber\_climb connects climber and climb, indicating the parcipation of climbers in specific climbs;
* this facilitates a many-to-many relationship, as a climber can participate in multiple climbs, and conversely, each climb can have several climbers involved.

Example with data

|  |  |  |
| --- | --- | --- |
| climb\_id | climber\_id | climb\_type |
| 1 | 1 | inside |
| 2 | 1 | outside |
| 1 | 2 | inside |

1. **Club table description**

This description highlights the club's mission to attract sponsorship not only for individual climbers but also for broader club initiatives. This structure provides all necessary information in one place.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| club | club\_id | unique identifier for club PK | int |
| name | name of the club (only 1 row in the table) | varchar (50) |
| location | location of the club | varchar (100) |
| telephone\_number | club telephone number | varchar (30) |
| email | club email | varchar (30) |
| description | description of mission and activities | text |

Comments on table relationships

* climber\_sponsor ↔ club: many-to-one (m:1) relationship, each climber\_sponsor belongs to one club (because club is only one);
* climb\_route ↔ club: many-to-one (m:1) relationship, each climb\_route belongs to one club.

Example with data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| club\_id | name | location | telephone\_number | email | description |
| 1 | Climbs in Seconds | Poland Krakow | +375174551789 | climbsinseconds@gmail.com | Mountain climbing club focused on promoting outdoor/indoor adventures and fostering a community of climbers. The club organizes various climbing events and training sessions while seeking sponsorship to support both individual climbers and overall club activities. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

1. **Climber\_climb table description**

This table records the participation of climbers in specific climbs, linking climbers to climbs

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climber\_climb | climber\_id | foreign key referencing the climber table FK PK (part of composite primary key) | int |
| climb\_id | foreign key referencing the climb table FK PK (part of composite primary key) | int |
| climb\_type | Indicates the type of climbing environment. Options include “inside” and "outside” | varchar (20) |

Comments on table relationships

* Climber\_climb ↔ club: many-to-one (m:1) relationship, each climber belongs to one club;
* This structure enables a many-to-many relationship, as a climber can participate in multiple climbs, and a climb can have multiple climbers.

Example with data

|  |  |  |
| --- | --- | --- |
| climber\_id | climb\_id | climb\_type |
| 1 | 1 | inside |
| 2 | 1 | inside |
| 1 | 2 | outside |

1. **Climb table description**

This table stores climb information.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climb | climb\_id | unique identifier for climbs PK | int |
| name | climb name | varchar (50) |

Comments on table relationships

* climb ↔ climber: many-to-many (m:m) relationship, each climber belongs to many climbs and climbs can have many climbers, via climber\_climb;
* climb ↔ route: many-to-many (m:m) relationship, each climb can have multiple routes, and each route can be part of multiple climbs. This relationship is managed via the c**limb\_route** table;
* climb ↔ weather: one-to-many (1:m) relationship, a single climb can span multiple days, and the weather conditions may vary throughout. Therefore, a climb can have multiple associated weather records.
* climb ↔ climb\_schedule: one-to-one (1:1) relationship, each climb has one corresponding schedule (begin date and end date)

Example with data

|  |  |
| --- | --- |
| climb\_id | name |
| 1 | Everest Expedition |
| 2 | Zakopane Morske Oko |
| 3 | Alpine Challenge |

1. **Climb\_schedule table description**

This table manages the scheduling of climbs.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climb | climb\_id | foreign key referencing the climb\_schedule FK | int |
| begin\_date | date of begin climb | date |
| end\_date | date of end climb | date |

Comments on table relationships

* climb ↔ climb\_schedule: one-to-one (1:1) relationship, each climb has one corresponding schedule (begin date and end date).

Example with data

|  |  |  |
| --- | --- | --- |
| climb\_id | begin\_date | end\_date |
| 1 | 2024-06-28 | 2024-07-13 |
| 2 | 2024-03-17 | 2024-03-17 |
| 3 | 2024-09-28 | 2024-09-30 |

1. **Weather table description**

This table stores weather information.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climb | weather\_id | unique identifier for weather PK | int |
| climb\_id | foreign key referencing the climb FK | int |
| temperature | information about temperature | decimal (4, 1) |
| wind\_speed | information about wind speed | decimal (4, 1) |
| precipitation | A boolean value indicating whether there was precipitation (rain, snow) during the climb. True means there was precipitation; False means there wasn't. | boolean |

Comments on table relationships

* climb ↔ weather: one-to-many (1:m) relationship, a single climb can span multiple days, and the weather conditions may vary throughout. Therefore, a climb can have multiple associated weather records.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| weather\_id | climb\_id | temperature | wind\_speed | precipitation |
| 1 | 1 | 18.5 | 5.2 | False |
| 2 | 1 | 7.0 | 6.3 | True |
| 3 | 3 | 17.2 | 10.2 | True |

1. **Route table description**

This table stores route information.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| route | route\_id | unique identifier for routes PK | int |
| name | climb name | varchar (50) |
| route\_difficulty\_id | foreign key referencing the route\_difficulty table FK | int |
| mountain\_id | foreign key to mountain table FK | int |

Comments on table relationships

* climb ↔ route: many-to-many (m:m) relationship, each climb can have multiple routes, and each route can be part of multiple climbs. This relationship is managed via the c**limb\_route** table;
* route ↔ route\_difficulty: one-to-one (1:1) relationship, each route has one corresponding difficulty;
* route ↔ mountain: many-to-one (m:1) relationship, many routes on one mountain.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| route\_id | name | difficulty\_level | mountain\_id |
| 1 | Everest Expedition | expert | 1 |
| 2 | Zakopane Morske Oko | easy | 2 |
| 3 | Alpine Challenge | medium | 3 |

1. **Climb\_route table description**

This table serves to associate specific climbs with their corresponding routes.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| climb\_route | climber\_id | foreign key referencing the climber table FK PK (part of composite primary key) | int |
| climb\_id | foreign key referencing the climb table FK PK (part of composite primary key) | int |
| club\_id | foreign key referencing the club table FK PK (part of composite primary key) | int |

Comments on table relationships

* Climb\_route ↔ club: many-to-one (m:1) relationship, each climb belongs to one club;
* This structure facilitates a many-to-many relationship, where a route can belong to multiple climbs, and a climb can comprise multiple routes.

Example with data

|  |  |  |
| --- | --- | --- |
| climber\_id | climb\_id | club\_id |
| 1 | 1 | 1 |
| 2 | 1 | 1 |
| 1 | 2 | 1 |

1. **Route\_difficulty table description**

This table is designed to store different difficulty levels for climbing routes.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| route | route\_difficulty\_id | unique identifier for the difficulty level PK | int |
| level | Indicates the type of difficulty level. Options include “easy”, “medium” and “expert” | varchar (50) |

Comments on table relationships

* route ↔ route\_difficulty: one-to-one (1:1) relationship, each route has one corresponding difficulty.

Example with data

|  |  |
| --- | --- |
| route\_difficulty\_id | level |
| 1 | expert |
| 2 | easy |
| 3 | medium |

1. **Mountain table description**

This table stores mountain information.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| mountain | mountain\_id | unique identifier for routes PK | int |
| name | climb name | varchar (50) |
| height | foreign key referencing the route\_difficulty table FK | int |
| location\_id | foreign key to mountain table FK | int |

Comments on table relationships

* mountain ↔ route: one-to-many (1:m) relationship, each mountain can have multiple routes;
* location ↔ mountain: many-to-one (m:1) relationship, many location to one mountain.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| mountain\_id | name | height | location\_id |
| 1 | Everest Expedition | 8848 | 1 |
| 2 | Zakopane Morske Oko | 3000 | 2 |
| 1 | Everest Expedition | 8848 | 3 |

1. **Location table description**

This table stores mountain location information.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| location | locaton\_id | unique identifier for mountain location PK | int |
| country | country name | varchar (50) |
| region | region name | varchar (50) |

Comments on table relationships

* location ↔ mountain: many-to-one (many:1) relationship, many location to one mountain.

Example with data

|  |  |  |
| --- | --- | --- |
| location\_id | country | region |
| 1 | Nepal | x |
| 3 | Himalai | y |
| 2 | Poland | z |

And so on so on so on 😊