# CA1: Database Design and Development

Contents

[CA1: Database Design and Development 1](#_Toc197771387)

[1. Project Overview/Scope 2](#_Toc197771388)

[2.Entity Relationship Diagram 4](#_Toc197771389)

[3.Assumptions Made 5](#_Toc197771390)

[4.Data Dictionary 5](#_Toc197771391)

[5.Technology Used 7](#_Toc197771392)

[6.Test Plan 7](#_Toc197771393)

# Project Overview/Scope

This project provides a technical design and provision of a new database server and the development of a new database for LetsGetFit gym that have multiple members, programs and trainers. Company have few requirements for their database:

1. In terms of members sign-in to gym, member is given a choice of payment options, they can choose just one but may change later.
2. Each member is issued with Membership card, which contains photo and unique membership number, when Membership card is used, login time and date is recorded.
3. Must be flexible to allow for changes such as members and trainers details.
4. It must lend itself to be used for marketing propose.
5. Key entities: Members, Trainers, Payments, Programs
6. In terms of GDPR compliance need to record details of:
7. If member is giving permission or not to contact with him\her by SMS, email
8. If member wish to retain their information’s after they leave or not and keep contact with them
9. In terms of data update:
10. Ability to create new member and their first program via a parameterised stored procedure
11. Ability to add new Trainer via a parameterised stored procedure and store all their information’s
12. Ability to delete a member and their foreign key records associated with this member via a parameterised stored procedure if they have made final payment and requested their details be deleted , as per GDPR compliance within 30 days of membership end date, database should retain business-critical information’s generated by this member
13. In terms of member data, if member has indicated that LetsGetFit can keep their information’s once they left, system should remain information’s but should show not active for applications
14. In terms of management information:
15. SQL view to see all active members their program and trainer details
16. In terms of deleted member data view:
17. SQL view to see not active members that ticked GDPR Data box

Ref.1 In Member table PaymentOptions column was created to store this information and it is changeable.

Ref.2 For this two tables was created, MembershipCard to store card information’s and Activity to store activity details, member photo is stored in Member table, unique membership number is generated once member is created in Member table in column MemberID.

Ref.3 Database allow to change member, trainer details.

Ref.4 Database allow to retrieve not active members who ticked GDPR data box and show their details if company would like to contact them.

Ref.5 Database contain key entities and three more PersonalizedProgram, Activity, MembershipCard.

Ref.6a This information is stored in Member table in LetsGetFitText column holding value 1 if allow for contact or 0 if not wishing.

Ref.6b This information is stored in Member table in LetsGetFitData column holding value 1 to keep data or 0 delete after end of membership.

Ref.7a Parameterised stored procedure spCreateMemberAndProgram was created for this propose.

Ref.7b Parameterised stored procedure spCreateTrainer was created for this propose.

Ref.7c Parameterised stored procedure spHardDelete was created for this propose.

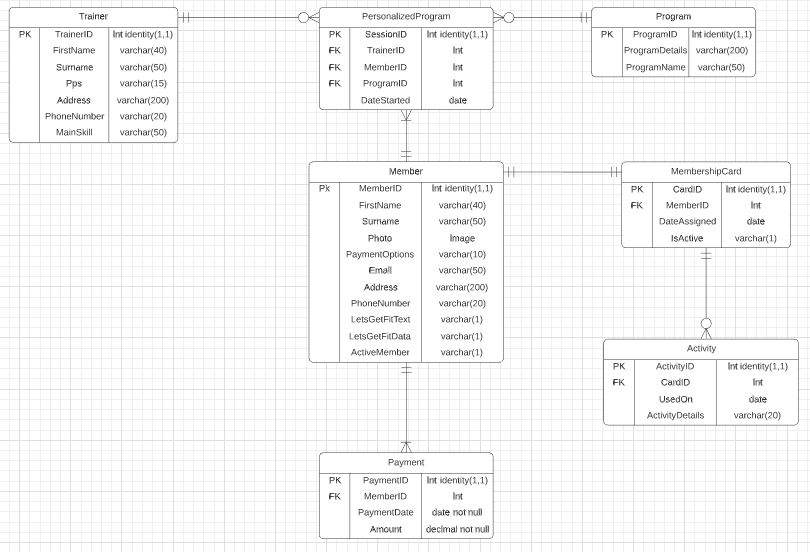
Ref.8 For this propose column ActiveMember in Member table was created, holding value 1 if active or 0 if not active.

Ref.9a SQL view was created viewActiveMembers.

Ref.10a SQL view was created viewNotActiveMembersGdprTicked.

All company requirements have been met.

# 2.Entity Relationship Diagram



# 3.Assumptions Made

Company listed 4 key entities and 3 additional was created:

1. MembershipCard table was created as each member have membership card assigned.
2. Activity was created as member activity need to be recorded.
3. PersonalizedProgram was created as each member when start gets his personalized program assigned

For Member table 11 columns was created to store member information’s. Column ActiveMember was created so applications know if the member is active or not.

When creating tables Payment, MembershipCard and PersonalizedProgram MemberID column was set to null on deletion of Member to retain other information’s critical for company.

When creating stored procedure spHardDelete was done few checks so is impossible to delete active member or member who ticked GDPR box. Also, when member left gym and ticked GDPR box is set to not active member for external applications.

# 4.Data Dictionary

All Data Dictionary based on ERD above.

Member

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Required** | **Description** |
| MemberID | Int identity (1,1) | Yes | Unique number for each member generated automatically |
| FirstName | Varchar (40) | No | Member first name |
| Surname | Varchar (50) | No | Member surname |
| Photo | Image | No | Member photo |
| PaymentOptions | Varchar (10) | No | Payment type made by member |
| Email | Varchar (50) | No | Member email |
| Address | Varchar (200) | No | Member address |
| PhoneNumber | Varchar (20) | No | Member phone number |
| LetsGetFitText | Varchar (1) | No | For GDPR if member tick box, agree to send emails and text |
| LetsGetFitData | Varchar (1) | No | For GDPR if member tick box to retain their information after they leave and keep contact with them |
| ActiveMember | Varchar (1) | No | For soft delete so applications know if member is active or not |

Trainer

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Required** | **Description** |
| TrainerID | Int identity (1,1) | Yes | Unique number for each trainer generated automatically |
| FirstName | Varchar (40) | No | Trainer first name |
| Surname | Varchar (50) | No | Trainer surname |
| Pps | Varchar (15) | No | Trainer pps number |
| Address | Varchar (200) | No | Trainer address |
| PhoneNumber | Varchar (20) | No | Trainer phone number |
| MainSkill | Varchar (50) | No | Trainer main skill |

Program

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Required** | **Description** |
| ProgramID | Int identity (1,1) | Yes | Unique number for each Program generated automatically |
| ProgramDetails | Varchar (200) | No | Program details |
| ProgramName | Varchar (50) | No | Program name |

Payment

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Required** | **Description** |
| PaymentID | Int identity (1,1) | Yes | Unique payment number generated automatically |
| MemberID | Int | No | Member unique number |
| PaymentDate | Date | Yes | Payment date |
| Amount | Decimal | Yes | Payment amount |

MembershipCard

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Required** | **Description** |
| CardID | Int identity (1,1) | Yes | Unique card number generated automatically |
| MemberID | Int | No | Member unique number |
| DateAssigned | Date | No | When card was assigned to member |
| IsActive | Varchar (1) | No | Is card active |

Activity

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Required** | **Description** |
| ActivityID | Int identity (1,1) | Yes | Unique activity number generated automatically |
| CardID | Int | No | Card number used |
| UsedOn | Date | No | Date when card was used |
| ActivityDetails | Varchar (20) | No | Activity details like door number |

PersonalizedProgram

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Datatype** | **Required** | **Description** |
| SesionID | Int identity (1,1) | Yes | Unique sesion number generated automatically |
| TrainerID | Int | No | Unique TrainerID number |
| MemberID | Int | No | Unique MemberID number |
| ProgramID | Int | No | Unique ProgramID number |
| DateStarted | Date | No | Date when personalized program starts |

# 5.Technology Used

Microsoft SQL Server Management Studio 18

Microsoft SQL Server Express 2017

Lucidchart.com website for ERD

# 6.Test Plan

Table 1 – the test plan

|  |  |  |  |
| --- | --- | --- | --- |
| Item Tested | Test Run | Expected Result | Actual Result |
| View viewActiveMembers | Select \* from viewActiveMembers | See all active members, their program details and trainer information’s | This same as expected result |
| View viewNotActiveMembersGdprTicked | Select \* from viewNotActiveMembersGdprTicked | See all not active members, their information’s who ticked GDPR box | This same as expected result |
| Stored procedure spCreateTrainer | execute spCreateTrainer'Neptun','Uranus','876353421','256 Park road, Indoskin','+3675271821','Machines'; | Trainer created | Trainer created |
| Stored procedure spCreateMemberAndProgram | execute spCreateMemberAndProgram 'Gudin','Mostano',null,'Monthly',  'Gudin@email.com','85 Kand road, Mustarke, Nidelka','+432535463',1,1,1,3,7,  '2020/11/11'; | Member created and their Personalized Program | Member created and their Personalized Program |
| Stored procedure spHardDelete against member who left gym and didn’t tick GDPR box | execute spHardDelete 9; | Member is deleted and all foreign key, retaining critical data | As expected, |
| Stored procedure spHardDelete against active member | execute spHardDelete 1; | Member is not deleted, displaying info “Member is active” | As expected, |
| Stored procedure spHardDelete against not active member who ticked GDPR box | execute spHardDelete 3; | Member is not deleted, displaying info “Member ticked GDPR box and is already not active” | As expected, |
| Stored procedure spHardDelete against active member who ticked GDPR data box but left gym (soft delete) | execute spHardDelete 10; | Member is not deleted, column ActiveMember in Member table is set to 0(not active), displaying info “Member ticked GDPR data box, set member as not active” | As expected, |
| Stored procedure spHardDelete against not existing member | execute spHardDelete 20; | Displaying info” Member doesnt exist,  Member doesnt have Payment Options set,  MemberID doesnt exist” | As expected, |