|  |
| --- |
| CS 5332: Database Theory and Design |
| Online Survey System |
| Guidance and Support Dr. Anne Hee Hiong Ngu |

Sruthi Chakravarthi and Team

# Requirements Gathering

We began the first phase of the project by reading the overview of the project, the aim was to design a database for the university’s Online survey system. We created several surveys on various Online survey platforms to make ourselves familiar with the types of surveys conducted, types of question/answers a typical survey can have. During this course, we familiarized ourselves with the type of question/ responses a typical survey has and studied the types of surveys like polls, course evaluations, ad hoc surveys and documented all the must haves for good Online surveys. Some of the Online survey systems we referred are the following:

* [https://www.surveymonkey.com](https://www.surveymonkey.com/)
* [https://www.qualtrics.com](https://www.qualtrics.com/)
* <https://www.limesurvey.org/>

We started the design of our database in line with the requirements from specification. Since all of us had taken university surveys in the past, we decided to think not only from a participant perspective but also from a survey creator perspective. The discussions happened over multiple meetings and it took considerable number of meetings to settle on a common ground for designing our database. We collected all our ideas and documented every single doubt and maintained an error log throughout the process. Following are a few pros and cons we listed during our meetings:

## Survey Monkey

Pros:

* + On Survey Monkey, lots of different options are available for question types like drop down, file upload, star rating and Date-time. The survey can be sent to many people at once.
  + We found that Survey Monkey was the best in monitoring as survey creators can view the results in customized charts and maps.
  + With this tool, you can do unique IP tracking as well. Overall, we found that the tool has an easy UI to navigate and crisp directions to use.

Cons:

* + We found that some of the features are limited when tried to create a new survey.
  + We observed that if we left a survey at a point and tried to come back to it, it was not saved. The drafting feature could have been a great add on to it.
  + We noticed that one cannot schedule the surveys to run at particular times which was also a requirement in our design.

## Qualtrics

Pros:

* + User friendly, very easy to use. One can navigate this tool without any difficulties. We loved the fact that it allows respondents to change their responses.
  + The feature which we liked the most is, this tool prevents the participant from taking the survey again, which is an important aspect in Online survey systems.
  + Another decent feature is- the surveys generated in Qualtrics have an expiry time.

Cons:

* + Very few options are available for question types. We noticed that we were getting stuck when filling in the surveys. There are some glitches with the UI.
  + The UI was user friendly but sometimes you need to struggle to understand it. The ease we had with the Lime survey was missing here.
  + Some of the options such as file upload, date-time, check box, comment box are

not available here.

## Lime Survey

Pros:

* + This platform features a clean and elegant interface.
  + Users can easily schedule surveys to run at a particular time.
  + The impressive feature about this tool is that it has anonymous responding options.

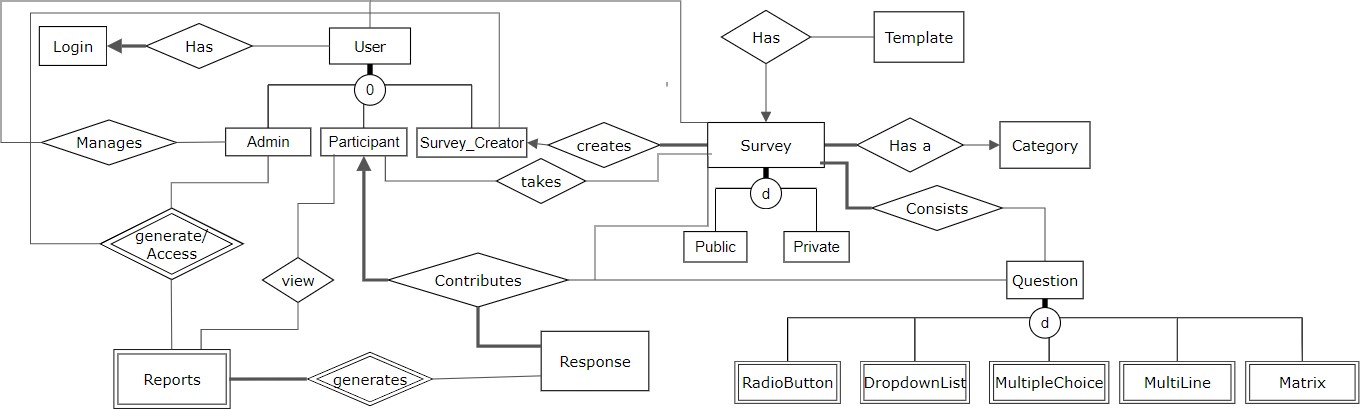
Cons:

* + The major drawback is fewer sharing options of this tool.
  + We noticed it was difficult to create surveys when compared to other platforms. The ease of use was not smooth.
  + This tool offers limited integrations.

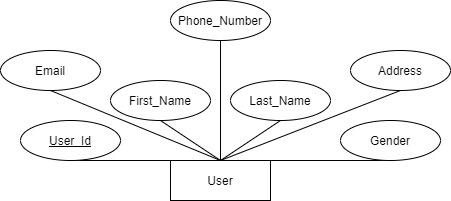
# Business Rules/ Constraints:

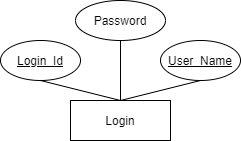
* User can be Admin, Participant, or survey creator.
* Each user has a unique Id and login credentials for all the registered users.
* Admin manages the users(create/delete/update).
* Admin will have access to all the reports.
* The registered users need to log in to respond to the survey
* Users can take survey anonymously for public surveys.
* Survey creators can create surveys.
* A survey creator can also participate in other surveys.
* Survey should no longer be available after designated date/time.
* Survey creators can access various types of reports from the generated responses.
* A participant can take multiple surveys (different surveys).
* A survey can have multiple participants.
* A participant cannot take the same survey more than once.
* Participants can view the report only after taking the survey.
* A respondent can view the surveys taken by him/her in the past.
* Each survey should belong to a survey category such as course evaluation, employee engagement, opinion poll etc.
* Each survey category can consist of multiple surveys.
* Every survey will have a set of questions with/without options.
* For each survey created, a unique URL/Link will be generated to send it to the respondents.
* Survey can use the template which has predefined set of questions.
* A question can be of different types for example- Radio button, Multiple choice, Matrix etc.
* Each question can be used in more than one survey.
* A question can have more than one choice.
* Each response stored in the table should be related to a participant and a survey.
* Points/Report should be generated after user submitted survey and stored
* Respondents cannot view responses of other participants in the survey.
* Admin/ Survey creator can view/generate the report
* No modifications are allowed after submission for participants.

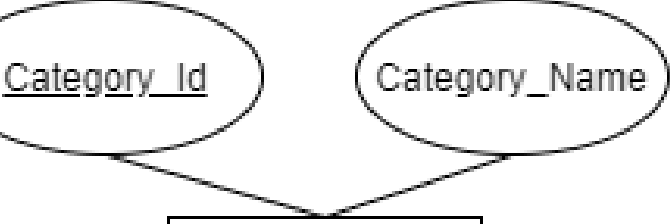
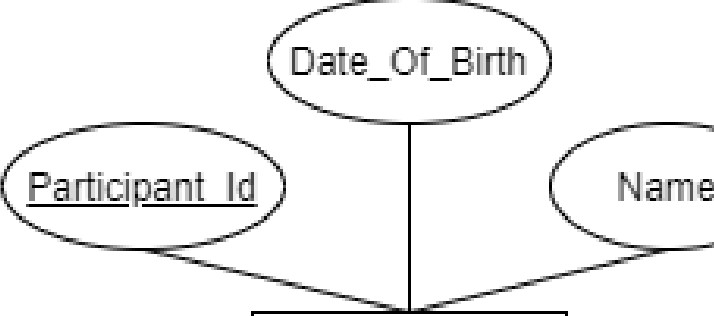
# ER Diagram:

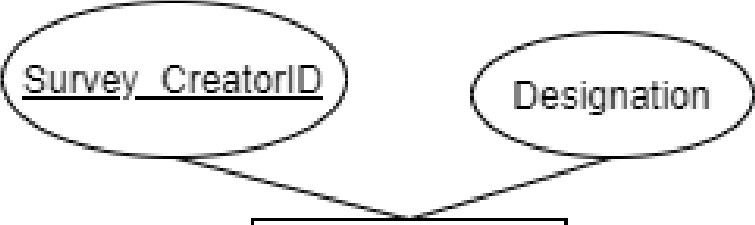


ER Design of each entity:

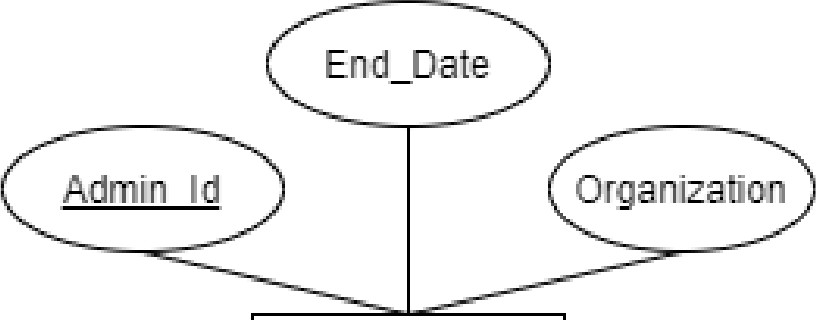




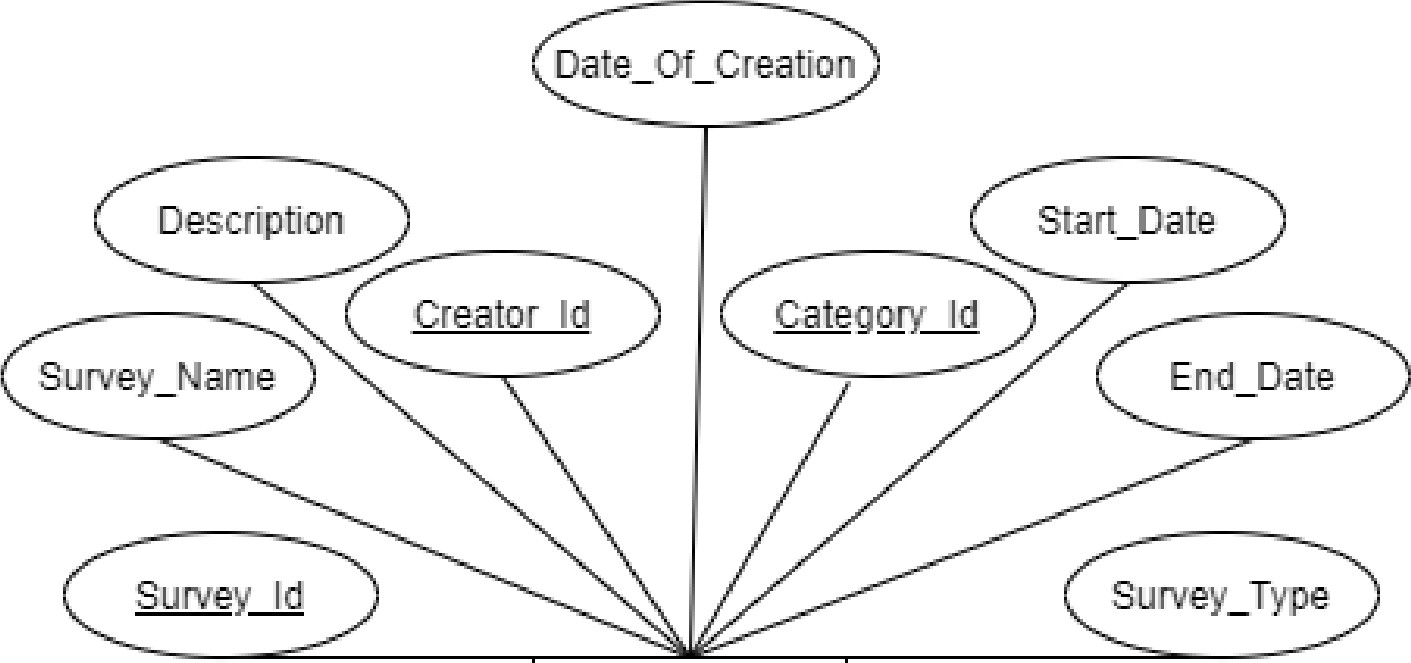




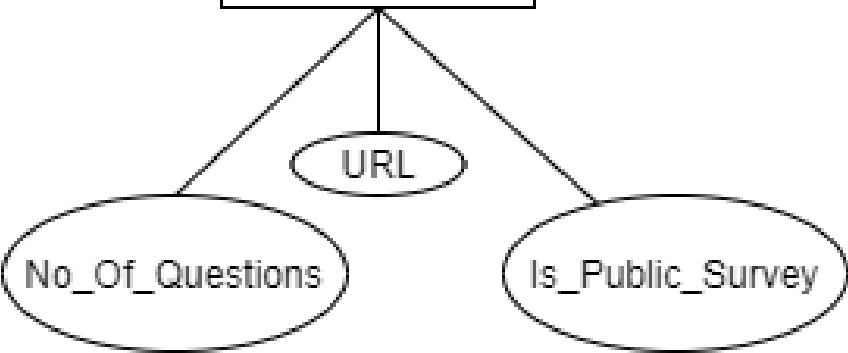
SurveyCm ator Partiicrpa11t

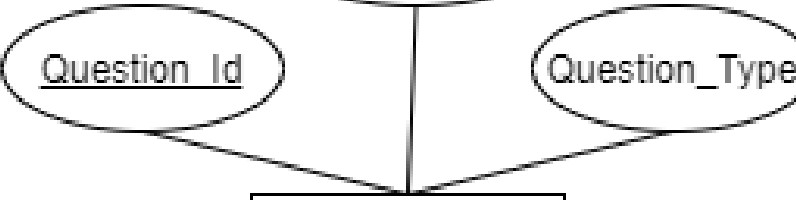
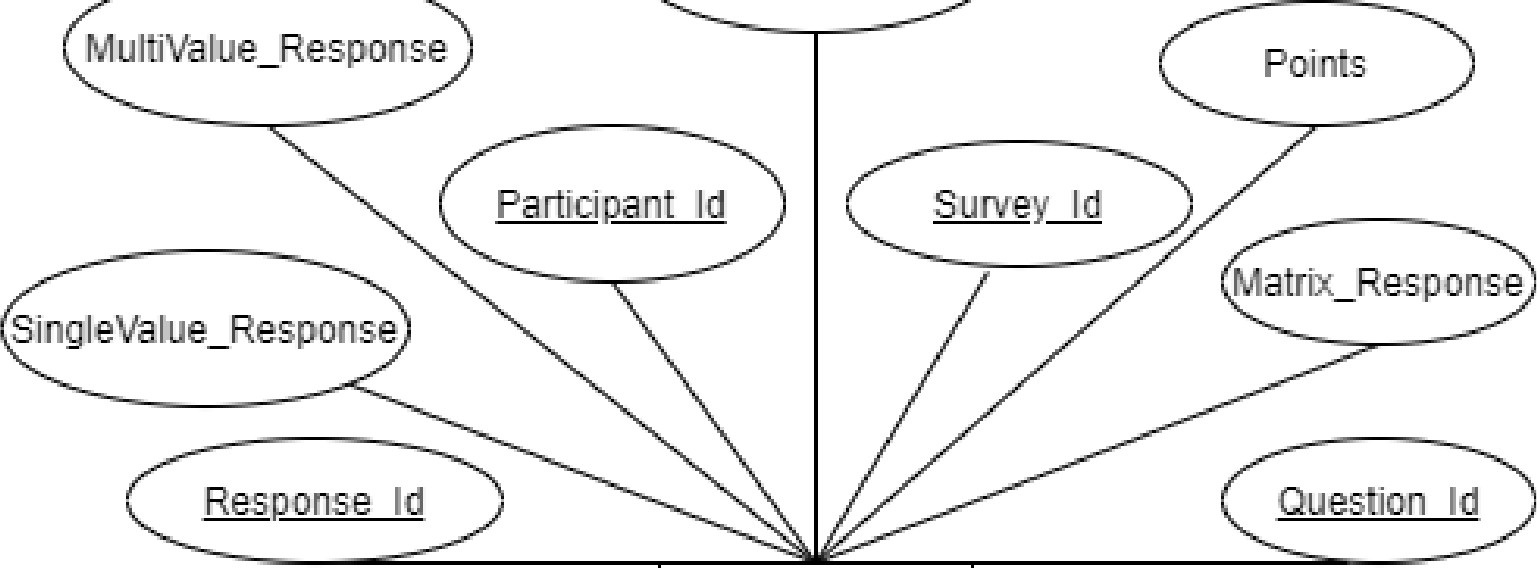


**Admill** Category



Survey



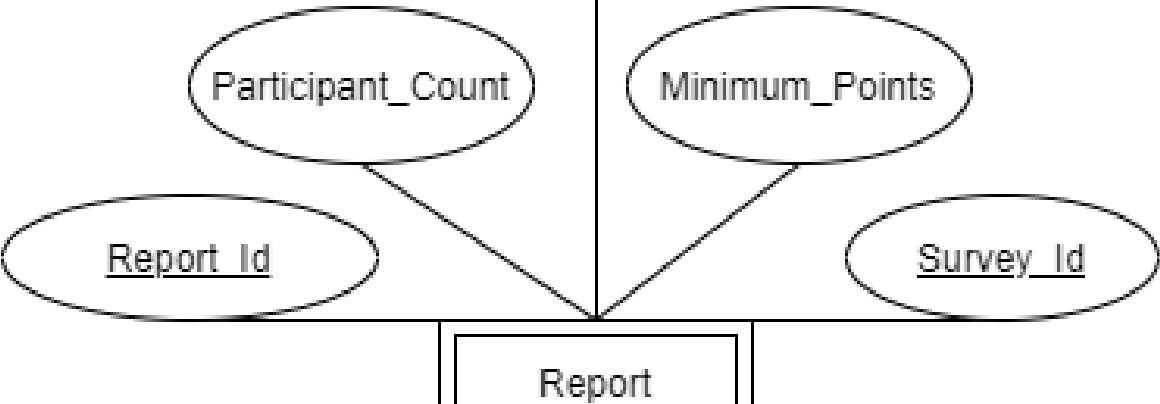




Multrli11e\_Respornse

Respon se

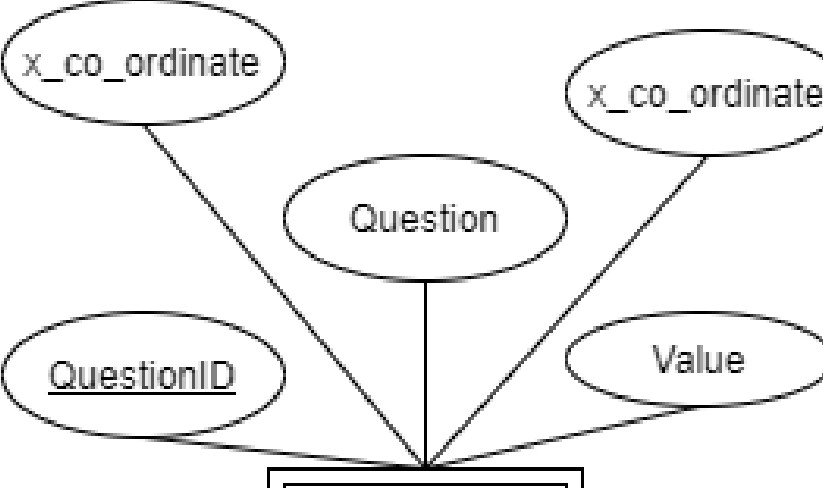
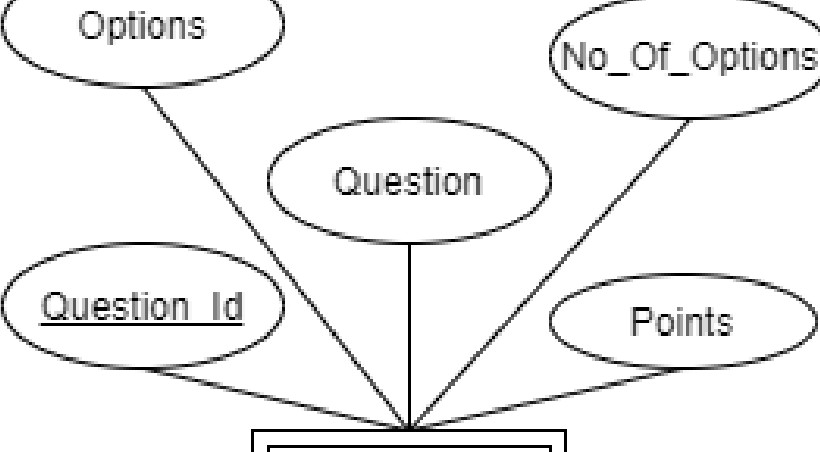


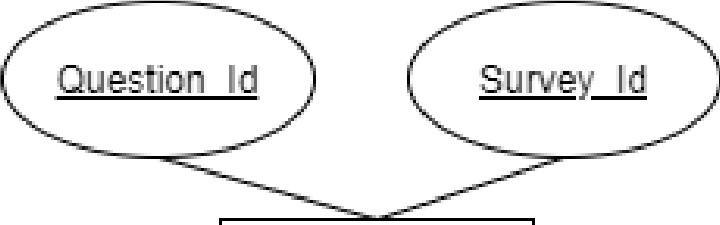




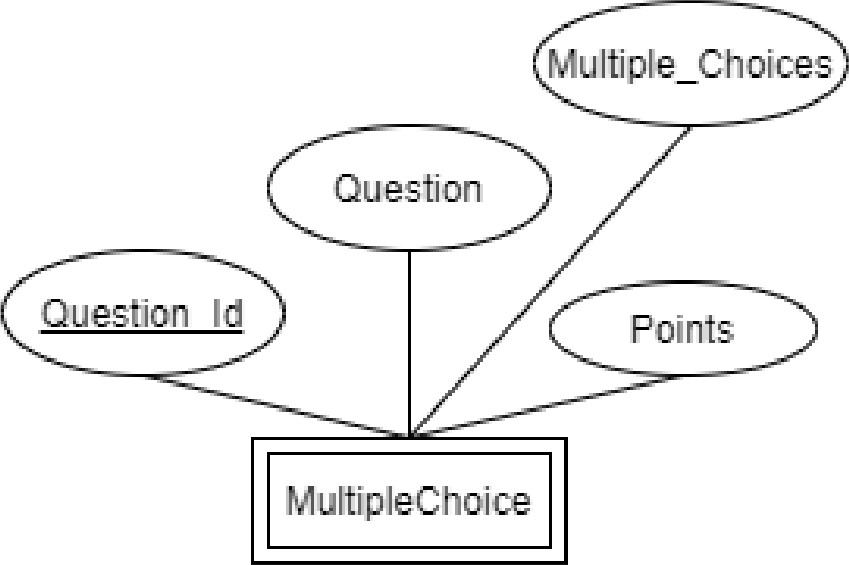
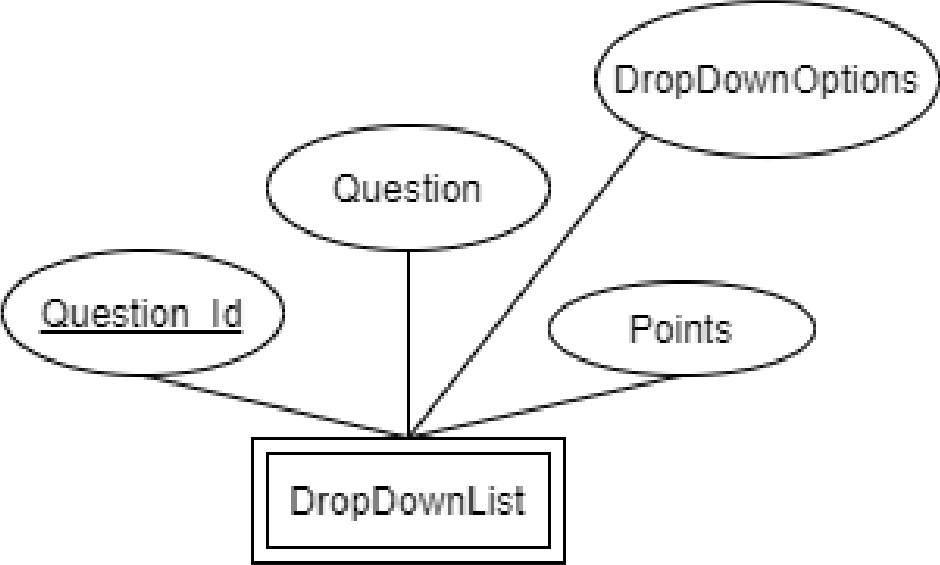
Ouestton\_Naime

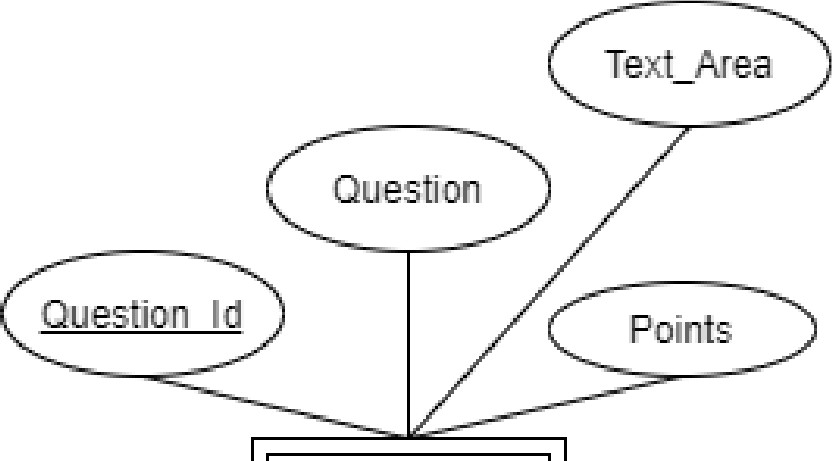
Ouesti-011s



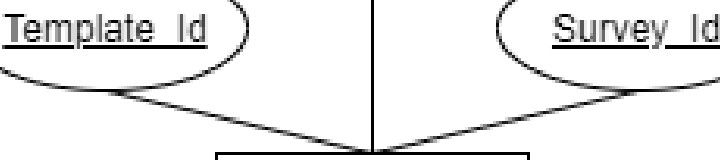


SurveyOLJeslio ri Radi0Butto11



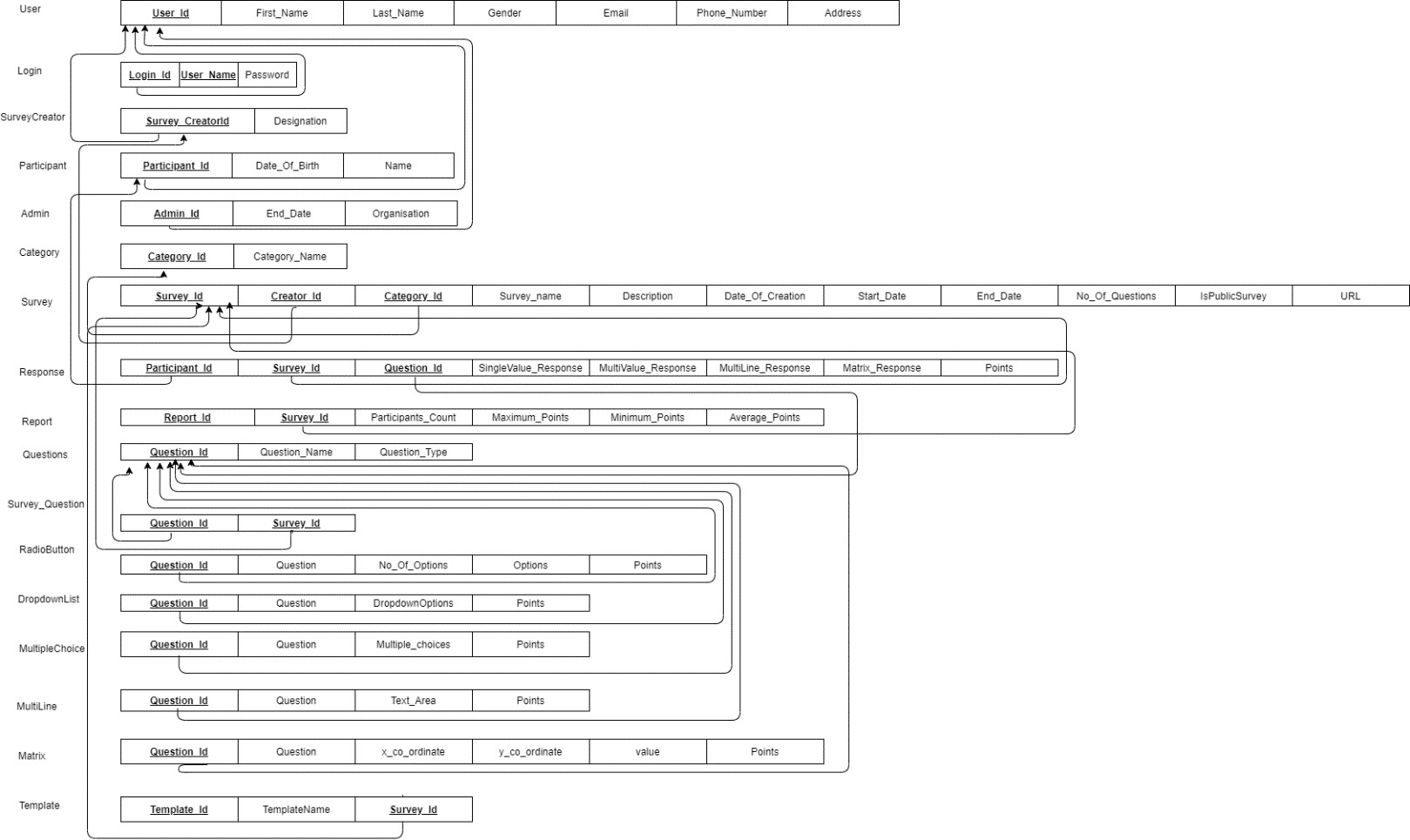
Ma lirix

M1.11til in e Points



Templae

# Relational Schema Diagram:



Relational Schema Definitions:

User (User\_Id, FirstName, LastName, Gender, Email, Phone\_Number, Address) Login (Login\_Id, Username, Password)

SurveyCreator (SurveyCreator\_Id, Designation) Participant (Participant\_Id, Date\_Of\_Birth, Name) Admin (Admin\_Id, End\_Date, Organization) Category (Category\_Id, Category\_Name)

Survey (Survey\_Id, Creator\_Id, Category\_Id, Survey\_Name, Description, DateOfCreation, Start\_Date, End\_Date, No\_Of\_Questions, IsPublicSurvey, URL)

Response (Participant\_Id, Survey\_Id, Question\_Id, SingleValue\_Response, MultiValue\_Response, Matrix\_Response, Points)

Report (Report\_Id, Survey\_Id, Participants\_Count, Maximum\_Points, Minimum\_Points, Average\_Points)

Questions (QuestionId, Question\_Type, Question\_Name) Survey\_Question (Question\_Id, Survey\_Id)

RadioButton (Question\_Id, Question, No\_Of\_Options, Options, Points) DropdownList (Question\_Id, Question, DropdownOptions, Points) MultipleChoice (Question\_Id, Question, Multiple\_Choices, Points) MultiLine (Question\_Id, Question, Text\_Area, Points)

Matrix (Question\_Id, Question, x\_co\_ordinate, y\_co\_ordinate, Value, Points) Template (Template\_Id, TemplateName, Survey\_Id)

# Design decision and Domain of all attributes:

|  |  |
| --- | --- |
| Entity: Login | |
| Primary key Attribute: User\_Name | |
| Foreign key Attribute: Login\_Id | |
| Attributes | Description |
| int Login\_Id | Login\_Id references the user\_Id from the User table |
| Varchar User\_Name [<50 chars] | To represent the User\_Name for the user |
| Varchar Password [<50 chars] | To represent the password for user |
| * Login Entity is used to store the username and password for all the users. * Each user is associated with a Login\_Id which references the User\_Id of Users table. | |

|  |  |
| --- | --- |
| Entity: User | |
| Primary key attribute: User\_Id | |
| Attributes | Description |
| Int User\_Id | An auto-generated id for each user |
| String First\_Name  [<50 chars] | To represent the User's FirstName |
| String Last\_Name  [<50 chars] | To represent the User’s Last Name |
| String Gender  [<50 chars] | To represent the Gender of User |
| Varchar Email  [String Value] | To represent the User's Email |
| Varchar Phone\_Number  [String Value] | To represent User's phone number |
| String Address  [<50 Value] | To represent the Address of user |
| * User is a superclass for the subclasses Admin, Participant and SurveyCreator. * Each user is associated with unique username and password. * There exists a 1:1 relationship between User and Login | |

|  |  |
| --- | --- |
| Entity: SurveyCreator | |
| Foreign key attribute: Survey\_CreatorId | |
| Attributes | Description |
| Int Survey\_Creator\_Id [Numeric Value] | To represent the Id for Survey Creator. |
| String Designation [<50 chars] | To represent the designation of survey creator. |
| * Survey Creator is a subclass from user and is inheriting the properties from user entity. * Survey Creator creates can create many surveys and each survey is associated with the survey creator. So, it is a 1: N relationship. * Since each survey should have a creator, there is total participation. And there can be few survey creators who did not create any survey, there will partial participation. | |

|  |  |
| --- | --- |
| Entity: Participant | |
| Foreign key attribute: Participant\_Id | |
| Attributes | Description |
| Int Participant\_Id [Numeric Value] | To represent the Identifier of Participant |
| Date Date\_Of\_Birth [Date String, 1990-01-25] | To represent the date of birth of Participant. |
| String Name [< 50 chars] | To represent the name of Participant |
| * Participant is a subclass from user and is inheriting the properties from user entity. * Participant can respond to many surveys. * Each survey response will have a single participant. So, 1: N relationship. | |

|  |  |
| --- | --- |
| Entity: Admin | |
| Foreign key attribute: Admin\_Id | |
| Attributes | Description |
| Int Admin\_Id [Numeric Value] | To represent the ID for Admin |
| Date End\_Date  [Date String, 1990-01-25] | To represent when the survey ends |
| String Organization [<50 chars] | To represent the organization to which Admin belongs. |
| * Admin is a subclass from user and is inheriting the properties from user entity. * Admin manages the Users. Admin also manages the surveys. * Admin can generate/view reports of the responses for each survey. | |

|  |  |
| --- | --- |
| Entity: Category | |
| Primary key attribute: Category\_Id | |
| Attributes | Description |
| Int Category\_Id [Numeric Value] | To represent the ID of the category. |
| String Category\_Name [<50 chars] | To represent the name of category. |
| * Each Survey belongs to category such as course evaluation, employee engagement, opinion poll etc | |

|  |  |
| --- | --- |
| Entity: Survey | |
| Primary Key Attribute: Survey\_Id | |
| Foreign Key Attribute: Creator\_Id, Category\_Id | |
| Attributes | Description |
| Int Survey\_Id [Numeric value] | To represent the Survey’s identifier |
| Int Creator\_Id [Numeric value] | To represent the SurveyCreator’s identifier from SurveyCreator entity |
| Int Category\_Id [Numeric value] | To represent the Category’s identifier from Category entity |
| String Survey\_Name [<100 chars] | To represent the Survey’s name |
| String Description [<300 chars] | To represent the Survey’s description |
| Date Date\_Of\_Creation [Date String, 1990-01-25] | To represent when the survey created |
| Date Start\_Date  [Date String, 1990-01-25] | To represent when the survey starts |
| Date End\_Date  [Date String, 1990-01-25] | To represent when the survey ends |
| Int No\_Of\_Questions [Numeric value] | To represent the total number of questions survey has |
| Boolean isPublicSurvey [Boolean value: True/False] | To represent whether Survey is public or private |
| String URL [<300 chars] | To represent the survey URL |
| * Survey can use templates. Each template can be used by multiple surveys, therefore 1: N relationship. * Survey belongs to category. A category can have multiple surveys, but each survey will belong to single category. So, M:1 relation. And, since survey must belong to a category, there is total participation. And a category may or may not have surveys, hence partial participation. * Survey consists of questions. A survey can contain many questions and a question can belong to many surveys, hence M: N relation. A survey should have questions so total participation, a question may or may not belong to survey, so there is partial participation. * Participant can respond to the same survey only once. Survey can have many participants. Hence 1: N relation. And participant must respond to survey so total participation. And a survey may or may not have participants, so partial participation. | |

|  |  |
| --- | --- |
| Entity: Response | |
| Composite Key Attribute: Participant\_Id, Survey\_Id, Question\_Id | |
| Attributes | Description |
| Int Participant\_Id [Numeric value] | To represent the Participant’s identifier |
| Int Survey\_Id [Numeric value] | To represent the Survey’s identifier |
| Int Question\_Id [Numeric value] | To represent the Question’s identifier from Category entity |

|  |  |
| --- | --- |
| String SingleValue\_Response [<50 chars] | To represent the Single value Response, like RadioButton |
| String MultiValue\_Response [<50 chars] | To represent the Multiple value Response, like Multiple Choice |
| String MultiLine\_Response [<50 chars] | To represent the Multiple Lines Response, like Text Area |
| String Matrix\_Response [<50 chars] | To represent the Matrix response, like Matrix |
| Int Points [Numeric value] | To represent the points |
| * Response generates the reports. Each report is associated with multiple responses and each response can be used to generate multiple reports. So, there is a M: N relationship. * For a report to exist there must be response, so there should be total participation. A response may or may not be used to create the report so there is a partial participation. * Each response is associated with a single participant, and a participant can have multiple responses for different surveys. So there exists a 1: N relation. | |

|  |  |
| --- | --- |
| Entity: Questions | |
| Primary Key Attribute: Question\_Id | |
| Attributes | Description |
| Int Question\_Id [Numeric Value] | To represent the Question’s identifier |
| String Question\_Name [<250 chars] | To represent the question’s name in text |
| String Question\_Type [<50 chars] | To represent the type of question |
| * Questions entity is further divided into subclasses RadioButton, DropdownList, MultipleChoice, MutliLine, Matrix to represent different type of survey questions. * A question may belong to multiple surveys. A survey can consist of multiple questions. So, M: N relationship. * A survey should contain questions, so total participation. A question may or may not belong to a survey hence partial participation. * A response entity can have responses to multiple questions. So, there is M: N relationship. * A response should be associated with question so there is total participation. A question may or may not belong to response so there is partial participation. | |

|  |  |
| --- | --- |
| Entity: Survey\_Question | |
| Composite Key Attribute: Question\_Id, Survey\_Id | |
| Attributes | Description |
| Int Question\_Id [Numeric Value] | To represent the Question’s identifier from Question entity |
| Int Survey\_Id [Numeric Value] | To represent the Survey’s identifier from Survey entity |
| * This entity maps the questions with the surveys. For each question associated with a survey, there will an entry in the Survey\_Questions table. | |

|  |  |
| --- | --- |
| Entity: RadioButton | |
| Foreign Key Attribute: Question\_Id | |
| Attributes | Description |
| Int Question\_Id [Numeric Value] | To represent the Question’s identifier from Question entity |
| String Question [<150 chars] | To represent the question’s name in text |
| int No\_Of\_Options [Numeric value] | To represent the number of choices to question |
| String Options [<150 chars] | To represent the choices to question |
| int Points [Numeric value] | To represent the points the question holds |
| * RadioButton entity is used represent the multiple-choice questions from with single choice responses. * RadioButton is a weak entity belonging to super class Questions. | |

|  |  |
| --- | --- |
| Entity: DropdownList | |
| Foreign Key Attribute: Question\_Id | |
| Attributes | Description |
| Int Question\_Id [Numeric Value] | To represent the Question’s identifier from Question entity |
| String Question [<150 chars] | To represent the question’s name in text |
| String DropdownOptions [<100 chars] | To represent the response choices to question |
| int Points [Numeric value] | To represent the points the question holds |
| * DropdownList entity is used to hold the Dropdown list type questions with single choice response. * DropdownList is weak entity belonging to super class Questions. | |

|  |  |
| --- | --- |
| Entity: MultipleChoice | |
| Foreign Key Attribute: Question\_Id | |
| Attributes | Description |
| Int Question\_Id [Numeric Value] | To represent the Question’s identifier from Question entity |
| String Question [<150 chars] | To represent the question’s name in text |
| String Multiple\_Choices [<100 chars] | To represent the response choices to question |
| int Points [Numeric value] | To represent the score the question holds |
| * MultipleChoice entity is used to hold the Multiple-Choice type questions where the response can be multiple options. * MultipleChoice is weak entity belonging to super class Questions | |

|  |  |
| --- | --- |
| Entity: MultiLine | |
| Foreign Key Attribute: Question\_Id | |
| Attributes | Description |
| Int Question\_Id [Numeric Value] | To represent the Question’s identifier from Question entity |
| String Question [<150 chars] | To represent the question’s name in text |
| String Text\_Area [<1000 chars] | To represent the response to question |
| int Points [Numeric value] | To represent the score the question holds |
| * MultiLine entity is used to hold the descriptive type questions from survey, response ranging up to 150 words * MultiLine is weak entity belonging to super class Questions | |

|  |  |
| --- | --- |
| Entity: Matrix | |
| Foreign Key Attribute: Question\_Id | |
| Attributes | Description |
| Int Question\_Id [Numeric Value] | To represent the Question’s identifier from Question entity |
| String Question [<150 chars] | To represent the question’s name in text |
| String x\_co\_ordinator [<50 chars] | To represent row’s value from Matrix type |
| String x\_co\_ordinator [<50 chars] | To represent column’s value from Matrix type |
| String Value [<150 chars] | To represent the response to question |
| int Points [Numeric value] | To represent the score the question holds |
| * Matrix entity is used to represent the matrix type questions. * Matrix is weak entity belonging to super class questions. | |

|  |  |
| --- | --- |
| Entity: Report | |
| Primary Key Attribute: Report\_Id | |
| Foreign Key Attribute: Survey\_Id | |
| Attributes | Description |
| Int Report\_Id [Numeric Value] | To represent the Report’s identifier |
| Int Survey\_Id [Numeric Value] | To represent the Survey’s identifier from Survey |
| int Participant\_Count [Numeric Value] | To represent the number of participants |
| int Maximum\_Points [Numeric Value] | To represent the maximum points achieved |

|  |  |
| --- | --- |
| int Minimum\_Points [Numeric Value] | To represent the minimum points achieved |
| int Average\_Points [Numeric Value] | To represent the average points achieved |
| * Reports is weak entity derived from Responses entity, without Reponses the Report cannot exist * Admin or SurveyCreator can generate/view report of the surveys. * Reports is related responses. Reports is associated with multiple responses and response may have multiple reports. So, M: N relationship. * For a report to exist there must be response, so there should be total participation. A response may or may not be used to create the report so there is a partial participation. | |

|  |  |
| --- | --- |
| Entity: Template | |
| Primary key attribute: Template\_Id | |
| Foreign Key Attribute: Survey\_Id | |
| Attributes | Description |
| Int Template\_Id [Numeric Value] | To represent the Template’s identifier |
| String TemplateName [<50 chars] | To represent the Template’s name |
| Int Survey\_Id [Numeric Value] | To represent the Survey’s identifier from Survey entity |
| * Template provides pre-defined set of questions that can be used to create survey. * Template can be associated with many surveys. Each survey is associated with single template\_Id. So there exists a 1: N relationship. | |

Description for mapping ER Diagram to Relational Schema Diagram

## N:M Relationships:

* There exists a "Has a" relation between Login entity and User Entity. This relation is handled by adding a foreign key user\_Id to the Login entity, also it satisfies the constraint that each user is associated with a unique username.
* Survey creator entity is connected by relation "creates" to the survey entity. This relation is a 1:M relation, as each survey will have one creator and a survey creator can create many surveys. We handled this by adding a foreign key constraint to the Survey entity.
* The relation between Survey entity and Category entity is handled by "belongs" relation. In this relation we use category\_id as the foreign key in Survey table, that way each category will belong to a particular category.
* The relation between Survey entity and Questions entity is handled by "consists" relation. This is handled by creating an entity Survey\_questions which has survey\_Id and question\_Id as foreign key, since each question\_Id is related to a survey\_Id

## Ternary Relationships:

* We have a ternary relation "Responds" which connects the entities Participant, Survey and Question in our ER diagram. We handled this relation by adding the composite key to the Response table which is combination of Survey\_Id, Question\_Id, and Participant\_Id. This composite key serves as the constraint that each participant will respond to a particular survey only once.

## Recursive Relationships:

* We do not have any recursive relationships in our ER diagram

## Multi-valued Attributes:

* The number of questions a survey consists can be multiple (For example survey\_id 2634 can have multiple question\_id's such as 34, 61, 23, 12). We handled this by adding a table Survey\_Questions which will have survey\_id and question\_id as foreign keys.

## Inheritance:

There are 3 instances of inheritance in our project.

* User is a superclass. Admin, Participant, and Survey\_Creator are the subclasses which inherit the User class. We implemented this inheritance by adding User\_Id as foreign constraint in Admin, Participant and Survey\_creator.
* Survey is a superclass. Public and Private surveys are the subclass of Survey class. Here we added the boolean parameter IsPublic to identify the public and private surveys.
* Radio, DropdownList, MultipleChoice, MultiLine, Matrix entities have a "is a" relationship with Questions entity. It is implemented by adding question\_Id as foreign key constraint in Radio, DropdownList, MultipleChoice, MultiLine, Matrix entities.

## Constraints that you can represent in ER diagram, but cannot be mapped to relational design:

* The admin manages the Users and the Survey. This is represented in ER diagram with a manages relationship, but we did not map it to relational design. During table creation, we can grant table-level permissions for the admin.

## Assumptions:

* We assume that, whenever a participant completes a survey, he/she will be awarded points by the web application.
* We assume that admin has access to remove/add the users
* We assume that admin has access to remove any inappropriate surveys or add any particular survey.
* We assume that for each anonymous survey, there will be an entry in the users table with null values except the user\_Id.
* Since the type of questions such as RadioButton, DropdownList, Matrix are all weak entities, they can be extended to support additional type of questions such as Audio, Video etc.

# Problems faced during the design of ER diagram and its Mapping:

* In the initial phase, the design decision whether to have a single users table or 4 tables (users, participants, survey\_creators, participants) was challenging. And, then we decided to implement the superclass- subclass approach, which makes our database design more scalable in future.
* Mapping the ternary relationship "Responds" in relational schema diagram was challenging. So, we came up with a composite key in the Response table which is combination of Survey\_Id, Question\_Id, and Participant\_Id. This key takes care of the constraint that a participant cannot take the same survey more than once, but he/she can take multiple surveys (different survey\_id).