Team 24

Yusuke Kawano  
Mohd Fadhli Ismail  
Izzatul Syazana Samsudin

Database project

Virtual Learning

Contents

[2 List of Queries 2](#_Toc414624758)

[2.1 Displaying Student’s Group No & The Team Members 2](#_Toc414624759)

[2.2 Displaying Student’s Ranking & Average Score 2](#_Toc414624760)

[2.3 Displaying Other Peers’ Average Score 2](#_Toc414624761)

[2.4 Upload XML File 2](#_Toc414624762)

[2.5 Show Comment & Score Received 3](#_Toc414624763)

[2.6 Show Ranking and Average Score Received by All Groups 3](#_Toc414624764)

[2.7 Number of Created Groups 3](#_Toc414624765)

[2.8 Show Students in Respective Allocated Group 3](#_Toc414624766)

[2.9 Show List of Unsorted Students 4](#_Toc414624767)

[2.10 Update Student’s Group 4](#_Toc414624768)

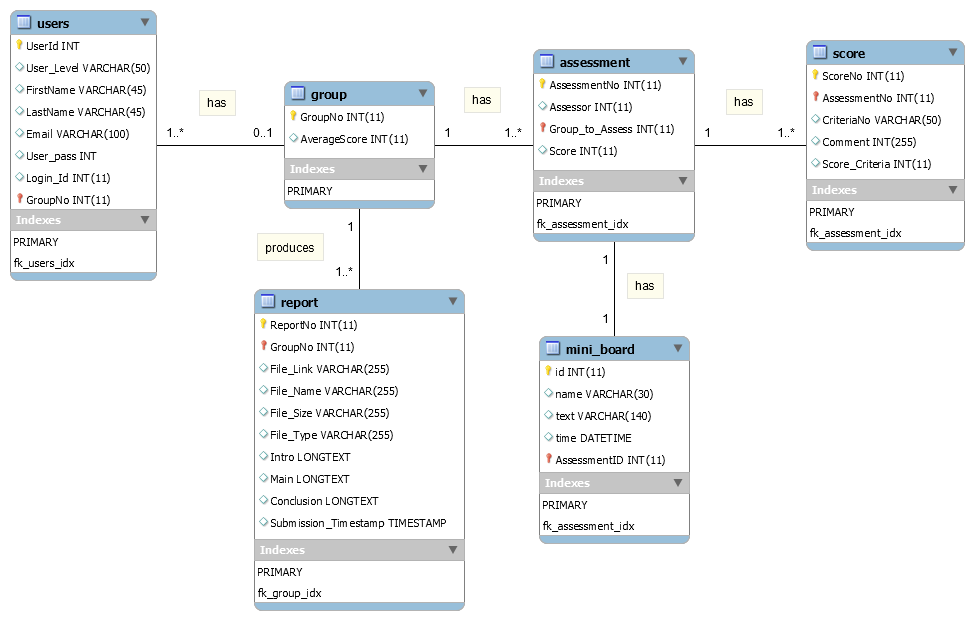
[2.11 Show List of Students 4](#_Toc414624769)

[2.12 Update Student’s Details 4](#_Toc414624770)

# Link to Youtube Video

<https://youtu.be/_payw6ILSxo>

# Entity Relationship Diagram (ERD)



Assumptions made:

# Database Schema

A listing of database schema with an explanation of how it translates the ER diagram

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| AssessmentNo | int(11) | NO | PRI | *NULL* | auto\_increment |
| Assessor | int(11) | NO |  | *NULL* |  |
| Group\_to\_Assess | int(11) | NO | MUL | *NULL* |  |
| Score | int(11) | NO |  | *NULL* |  |

Table 1: `assessment` table

asdsda"If Key is MUL, the column is the first column of a nonunique index in which multiple occurrences of a given value are permitted within the column."

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| GroupNo | int(11) | NO | PRI | *NULL* | auto\_increment |
| AverageScore | int(11) | YES |  | *NULL* |  |

Table 2: `group` table

Asdasds

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| id | int(11) unsigned | NO | PRI | *NULL* | auto\_increment |
| name | varchar(30) | NO |  | *NULL* |  |
| text | varchar(140) | NO |  | *NULL* |  |
| time | datetime | NO |  | *NULL* |  |
| AssessmentID | int(11) | NO |  | *NULL* |  |

Table 3: `mini\_board` table

Asdasdasd

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| ScoreNo | int(11) | NO | PRI | *NULL* | auto\_increment |
| AssessmentNo | int(11) | NO | MUL | *NULL* |  |
| CriteriaNo | varchar(50) | NO |  | *NULL* |  |
| Comment | varchar(255) | NO |  | *NULL* |  |
| Score\_Criteria | int(11) | NO |  | *NULL* |  |

Table 4: `score` table

Foreign key in `Assessment` table is `Group\_to\_Assess`.

# Third Normal Form

Analysis showing the database schema in 3rd normal form

# List of Queries

## Displaying Student’s Group No & The Team Members

SELECT FirstName, LastName, Email FROM `users` WHERE GroupNo = ?

This query is to display team member’s name in the group, which is shown on the home.php page. This is to allow user-student to easily contact their team members for the first meeting in case they do not know each other before.

## Displaying Student’s Ranking & Average Score

SELECT GroupNo, AverageScore, Rank FROM

(SELECT GroupNo, AverageScore,

@curr\_rank := IF(@prev\_score = AverageScore, @curr\_rank, @incr\_rank) AS Rank,

@incr\_rank := @incr\_rank + 1,

@prev\_score := AverageScore

FROM `group` AS g,

(SELECT @curr\_rank :=0, @prev\_score := NULL, @incr\_rank := 1) AS r

ORDER BY AverageScore DESC) AS s WHERE GroupNo = ?

This query is to show user-student’s group ranking as well as the average score they received from the peer assessments. In this query, if the average score of a group is the same as the average score in the previous rank, the rank of the group will remain the same as the current rank. Otherwise, the current rank is incremented by 1. Although the current rank remains the same, the rank will still be incremented in general with this function @incr\_rank := @incr\_rank + 1, hence there will be gap between the rank.

For example, if two groups received the same mark, they will receive the same rank, 8 for example. If a subsequent group has a different mark, this group rank will be 10 and not 9.

## Displaying Other Peers’ Average Score

SELECT `Assessor` FROM `assessment` WHERE `Group\_to\_Assess` = ?;

SELECT `AverageScore` FROM `group` WHERE `GroupNo` = ?

The objective of this query is to display the marks of other groups who assess another group in order to provide confidence and reliability of the assessments made by the peers. The first query is to find out the groups who are assessing the specific group. As the results are fetched from the first query in a while-loop, the second query is invoked to fetch the average score of each group of assessors. The results are displayed in the home.php page.

## Upload XML File

INSERT INTO report (GroupNo, File\_Link, File\_Name, File\_Size, File\_Type, Intro, Main, Conclusion) VALUES (?, ?, ?, ?, ?, ?, ?, ?)

ON DUPLICATE KEY UPDATE

File\_Link = ?, File\_Name = ?, File\_Size = ?, File\_Type = ?, Intro = ?, Main = ?, Conclusion = ?

This query runs when user upload an XML type file where the content of the XML file will be extracted to the database based on the element tag in the file. The presence of “ON DUPLICATE KEY UPDATE” in the query means that when the user re-uploads or resubmits their report, the value in the database will be updated if the `GroupNo` already exists. It will not insert a new row in the `report` table.

## Show Comment & Score Received

SELECT score.CriteriaNo, score.Comment, score.Score\_Criteria

FROM score

INNER JOIN assessment

ON assessment.AssessmentNo = score.AssessmentNo

WHERE assessment.AssessmentNo = ?

The above query is executed to display the comments and scores received for each criteria. Each group has been assigned to assess three other groups. To ensure anonymous assessment, `AssessmentNo` notation is used, thus the group will not know which groups they are assessing. The `score` and `assessment` tables are joined where the `AssessmentNo` is the foreign key. The results is fetched by looping through the `AssessmentNo` for that particular group.

SELECT score.AssessmentNo, SUM(score.Score\_Criteria) as OverallScore, ROUND(SUM((score.Score\_Criteria/3)\*2)) AS AverageScore

FROM score

INNER JOIN assessment

ON assessment.AssessmentNo = score.AssessmentNo

WHERE assessment.Group\_to\_Assess = ?

GROUP BY assessment.AssessmentNo

This query calculates the total score for each criteria for each assessment, where the result is used to find the average score for all three assessments.

## Show Ranking and Average Score Received by All Groups

SELECT GroupNo, AverageScore, Rank FROM

(SELECT GroupNo, AverageScore,

@curr\_rank := IF(@prev\_score = AverageScore, @curr\_rank, @incr\_rank) AS Rank,

@incr\_rank := @incr\_rank + 1,

@prev\_score := AverageScore

FROM `group` AS g,

(SELECT @curr\_rank :=0, @prev\_score := NULL, @incr\_rank := 1) AS r

ORDER BY AverageScore DESC) AS s

This query will rank each group based on their average score and list it on the student\_assessment.php page of the admin interface. The logic of this query is explained in details in section 2.2.

## Number of Created Groups

SELECT COUNT(DISTINCT GroupNo) AS `countgroup` FROM users WHERE `GroupNo` !=0

The `users` table consists of list of registered users on the website and they are distinguished by the user level: student or admin. If the `GroupNo` is zero, it means the student has not been assigned a group yet or if the user is an admin. Since the `GroupNo` is not unique for each row, the aim of this query is to count the number of existing groups by utilising the “DISTINCT” function and display it so that the admin knows the number of groups created.

## Show Students in Respective Allocated Group

SELECT `UserId`, `FirstName`, `LastName`, `GroupNo` FROM `users` WHERE `GroupNo` = ?

This query will display students in the group that they have been allocated on. In order to show this, the query is iterated equivalence to the number of groups that have been created, a value which is obtained from the query in section 2.7.

## Show List of Unsorted Students

SELECT `UserId`, `FirstName`, `LastName`, `GroupNo` FROM `users` WHERE `GroupNo` = 0 AND `User\_Level` = 'student' ORDER BY `FirstName`

For users labelled as “student” with `GroupNo` = 0 in the `users` table, this means the student has not been sorted to a group yet. This will allow admin to allocate the group to the student.

## Update Student’s Group

UPDATE `users` SET `GroupNo` = ? WHERE UserId = ?

The group allocation for student in the admin interface is done by dragging the name of the student into a box/container labelled with `GroupNo`. AJAX will processed the parameters, which are `GroupNo` and `UserId` and send to server, where it will update the `GroupNo` according to the specified `UserId`.

## Show List of Students

SELECT `UserId`, `FirstName`, `LastName`, `Email`, `Login\_Id`, `GroupNo` FROM `users` WHERE `User\_Level` = 'student' ORDER BY `FirstName`

This query will list down all registered students in a table and arranged them in order of their first name. Admin will be able to filter `FirstName`, `LastName` and `Login\_Id` to search for a particular student.

## Update Student’s Details

UPDATE `users` SET " .$name. " = ? WHERE UserId = ?

The variable $name represents the column in the `users` table. There are four components of the student details that the admin can amend: `FirstName`, `LastName`, `Email` & `GroupNo`. These are the values for $name. Therefore, this query is to update either of the aforementioned components for a student with the particular UserId.