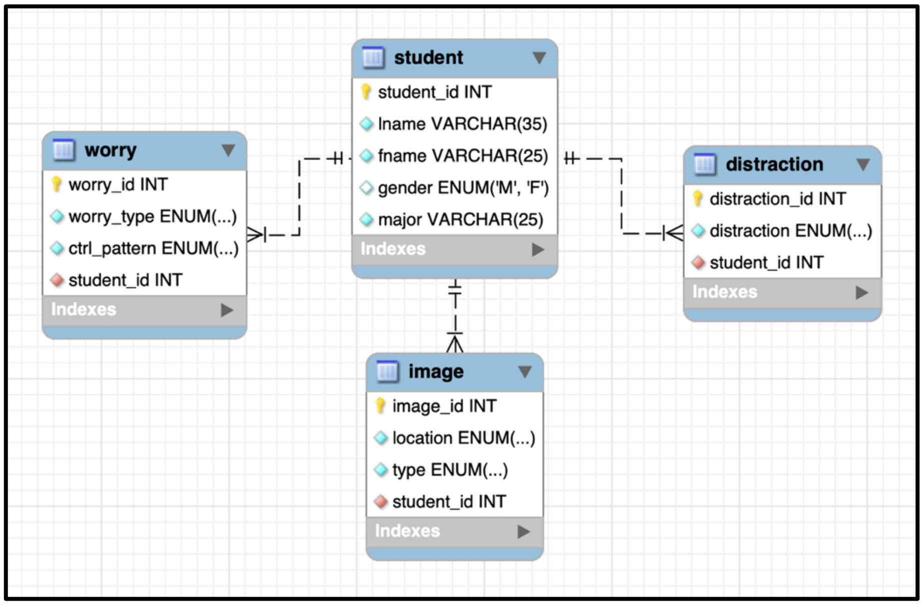
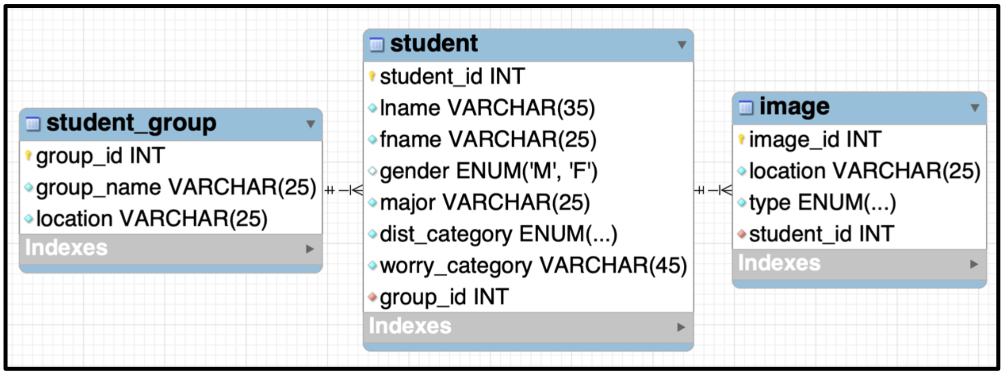
W04 – Relational Databases

CASE STUDY – Looking at the same Entity Relationship Diagrams from last week but now looking at the relationships between the entities.



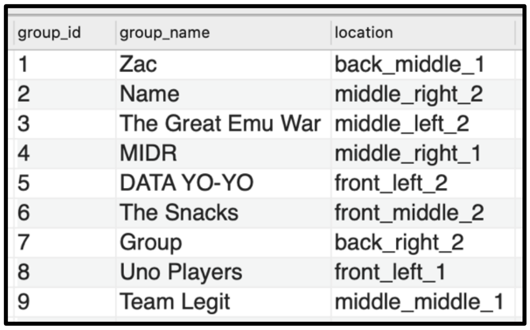
Here is the first ERD we looked at last week when we were discussing datatypes. Now let's talk about the dotted lines connecting each entity. Each student had worries, images and distractions that they kept track of. The relationships between the student and worry, for example, is a one-to-many relationship. Each student had one or more worries and each worry belongs to one and only one student. The image and distraction entities are related the same way with student. Each student had many images and each image was taken by one student. Each student had one or more distractions and each distraction belonged to one student.

Here's the other ERD we saw last week.



There are only 3 entities with this ERD. They are also showing a one-to-many relationship between each entity. Each group in the class has many different students in it and each student belongs to one and only one group. Each student had many different images and each image was taken by one and only one student. The worry and distraction was boiled down to just one value of that student's top distraction or top worry and included as an attribute inside the student table instead of as a separate entity in this ERD.

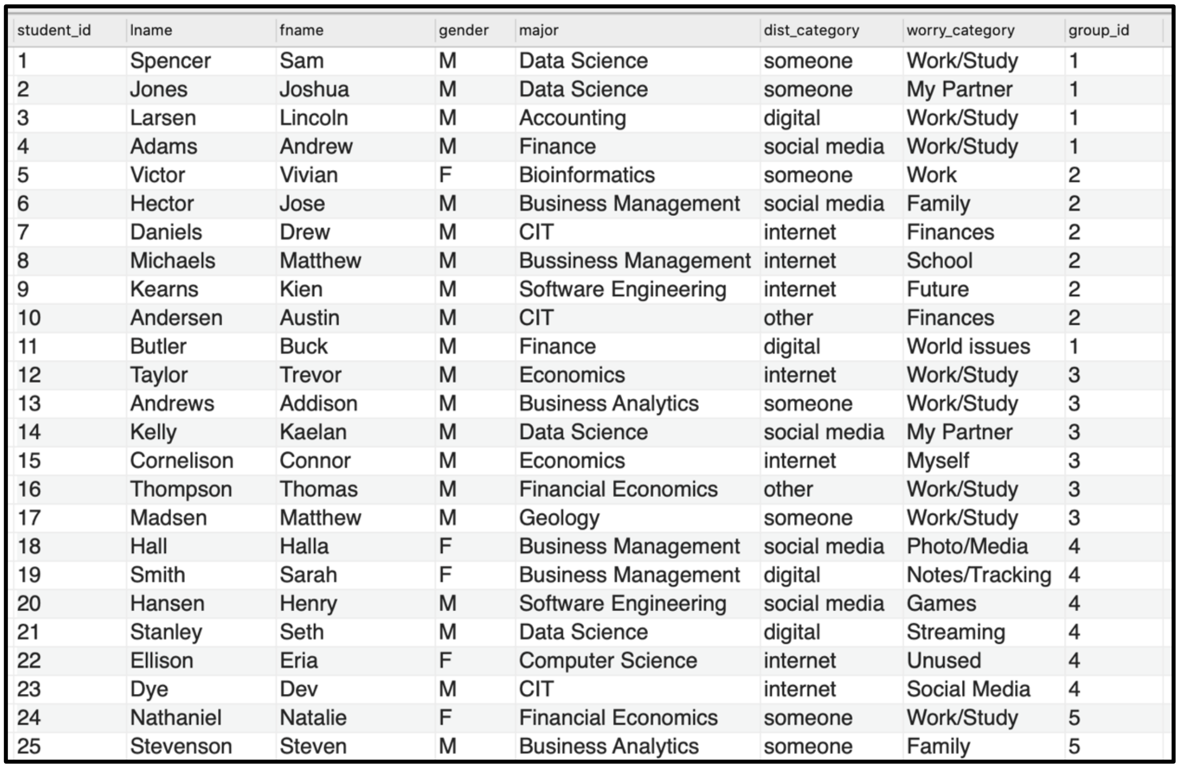
Notice how the primary key of group_id from the student_group entity is being used as a foreign key of the student table. This shows the relationship of those entities. Group_id in the student_group table is the primary key and there for is unique for every row of that table (or in other words each group will have a unique primary key number representing their group).



group_id	group_name	location
1	Zac	back_middle_1
2	Name	middle_right_2
3	The Great Emu War	middle_left_2
4	MIDR	middle_right_1
5	DATA YO-YO	front_left_2
6	The Snacks	front_middle_2
7	Group	back_right_2
8	Uno Players	front_left_1
9	Team Legit	middle_middle_1

The numbers 1 through 9 here are representing the primary keys for each group. There is no repeated numbers here. They are all unique.

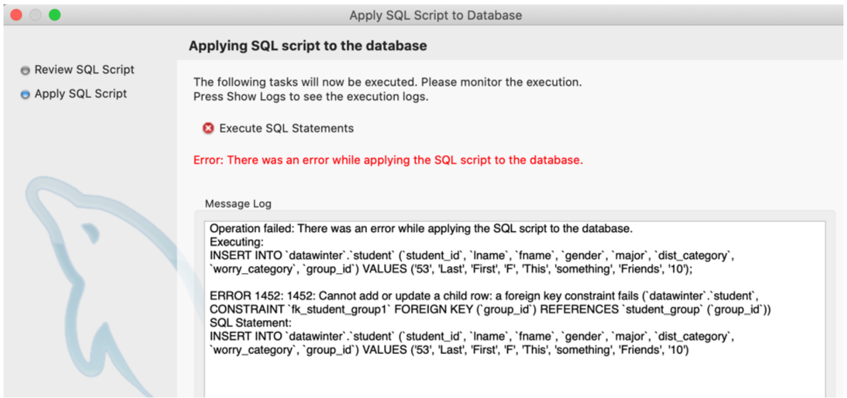
Now let's look at the student table.



student_id	lname	fname	gender	major	dist_category	worry_category	group_id
1	Spencer	Sam	M	Data Science	someone	Work/Study	1
2	Jones	Joshua	M	Data Science	someone	My Partner	1
3	Larsen	Lincoln	M	Accounting	digital	Work/Study	1
4	Adams	Andrew	M	Finance	social media	Work/Study	1
5	Victor	Vivian	F	Bioinformatics	someone	Work	2
6	Hector	Jose	M	Business Management	social media	Family	2
7	Daniels	Drew	M	CIT	internet	Finances	2
8	Michaels	Matthew	M	Bussiness Management	internet	School	2
9	Kearns	Kien	M	Software Engineering	internet	Future	2
10	Andersen	Austin	M	CIT	other	Finances	2
11	Butler	Buck	M	Finance	digital	World issues	1
12	Taylor	Trevor	M	Economics	internet	Work/Study	3
13	Andrews	Addison	M	Business Analytics	someone	Work/Study	3
14	Kelly	Kaelan	M	Data Science	social media	My Partner	3
15	Cornelison	Connor	M	Economics	internet	Myself	3
16	Thompson	Thomas	M	Financial Economics	other	Work/Study	3
17	Madsen	Matthew	M	Geology	someone	Work/Study	3
18	Hall	Halla	F	Business Management	social media	Photo/Media	4
19	Smith	Sarah	F	Business Management	digital	Notes/Tracking	4
20	Hansen	Henry	M	Software Engineering	social media	Games	4
21	Stanley	Seth	M	Data Science	digital	Streaming	4
22	Ellison	Eria	F	Computer Science	internet	Unused	4
23	Dye	Dev	M	CIT	internet	Social Media	4
24	Nathaniel	Natalie	F	Financial Economics	someone	Work/Study	5
25	Stevenson	Steven	M	Business Analytics	someone	Family	5

Sam, Joshua, Lincoln, Andrew, and Buck are all in group #1. We can see that because the foreign key of group_id is referring back to the primary key of student_group. So, Sam, Joshua, Lincoln, Andrew and Buck all belong to the group called 'Zac'. Notice how the foreign key value can repeat again and again for that group because those 5 students were in that group. That is the many part of the relationship.

How does Referential Integrity play into this relationship? The relationship between the student_group and student table is enforced between the primary key of student_group and the foreign key of student. If I were to add a new student #26 and I entered all his information and then in the last column put a 10 as the group_id foreign key value; I have violated referential integrity because there is no group with the primary key of 10. The MySQL Workbench would give us an error and not let us to that. The foreign key must be a value that already exists as a primary key of the related table.


Apply SQL Script to Database
Applying SQL script to the database
Review SQL Script
Apply SQL Script

The following tasks will now be executed. Please monitor the execution.
Press Show Logs to see the execution logs.

Execute SQL Statements

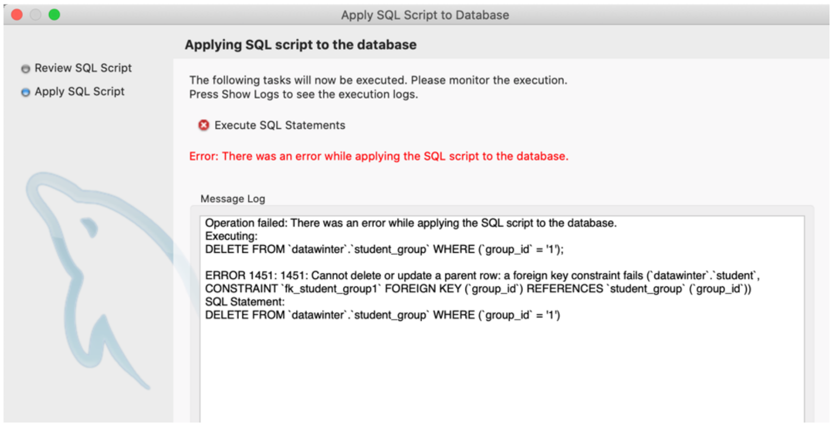
Error: There was an error while applying the SQL script to the database.

Message Log

```
Operation failed: There was an error while applying the SQL script to the database.
Executing:
INSERT INTO `datawinter`.`student` (`student_id`, `lname`, `fname`, `gender`, `major`, `dist_category`,
`worry_category`, `group_id`) VALUES ('53', 'Last', 'First', 'F', 'This', 'something', 'Friends', '10');

ERROR 1452: 1452: Cannot add or update a child row: a foreign key constraint fails (`datawinter`.`student`,
CONSTRAINT `fk_student_group1` FOREIGN KEY (`group_id`) REFERENCES `student_group` (`group_id`))
SQL Statement:
INSERT INTO `datawinter`.`student` (`student_id`, `lname`, `fname`, `gender`, `major`, `dist_category`,
`worry_category`, `group_id`) VALUES ('53', 'Last', 'First', 'F', 'This', 'something', 'Friends', '10')
```

Also, if I wanted to delete a group, for example group 1 'Zac' and I ran a DELETE statement to get rid of that group, my system would again give me an error because I can't delete a group that has student's in it. Or in other words I can't delete an instance of an entity that has its primary key being used as foreign key values in another table. I would first have to go into the student table and delete every student that belongs to that group first and then I could do back to the student_group table and delete the group once it was empty or didn't have any students related to it. It's important to realize how the system will enforce these relationships.


Apply SQL Script to Database
Applying SQL script to the database
Review SQL Script
Apply SQL Script

The following tasks will now be executed. Please monitor the execution.
Press Show Logs to see the execution logs.

Execute SQL Statements

Error: There was an error while applying the SQL script to the database.

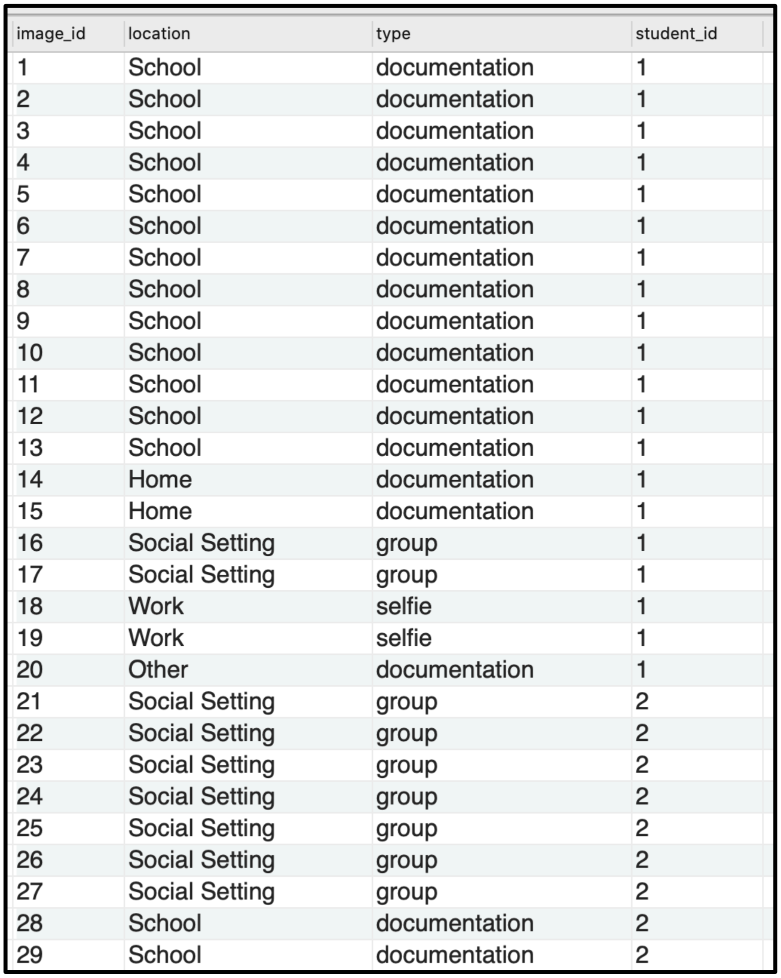
Message Log

```
Operation failed: There was an error while applying the SQL script to the database.
Executing:
DELETE FROM `datawinter`.`student_group` WHERE (`group_id` = '1');

ERROR 1451: 1451: Cannot delete or update a parent row: a foreign key constraint fails (`datawinter`.`student`,
CONSTRAINT `fk_student_group1` FOREIGN KEY (`group_id`) REFERENCES `student_group` (`group_id`))
SQL Statement:
DELETE FROM `datawinter`.`student_group` WHERE (`group_id` = '1')
```

Now let’s look at the relationship between the student entity and the image entity. Each student recorded many different images and each image belongs to one student. Again, a one-to-many relationship.

Here is some of the data in the image table.



image_id	location	type	student_id
1	School	documentation	1
2	School	documentation	1
3	School	documentation	1
4	School	documentation	1
5	School	documentation	1
6	School	documentation	1
7	School	documentation	1
8	School	documentation	1
9	School	documentation	1
10	School	documentation	1
11	School	documentation	1
12	School	documentation	1
13	School	documentation	1
14	Home	documentation	1
15	Home	documentation	1
16	Social Setting	group	1
17	Social Setting	group	1
18	Work	selfie	1
19	Work	selfie	1
20	Other	documentation	1
21	Social Setting	group	2
22	Social Setting	group	2
23	Social Setting	group	2
24	Social Setting	group	2
25	Social Setting	group	2
26	Social Setting	group	2
27	Social Setting	group	2
28	School	documentation	2
29	School	documentation	2

Notice the foreign key of student_id is how it relates to the student table. Each student had a unique number that can then be repeated for each of their images in the image table. So images 1 to 20 were all from the same student, Sam Spencer. His primary key was #1 and that 1 that is repeated again and again tells us that those are all his images. When you get to images 21 through 29 those are Joshua Jones’ images. His primay key was #2 and that 2 as a foreign key tell us those are his images.