CSU34041 Information Management Project

Denisa Patricia Costinas, Std: 19334772

March 25th 2022

Contents

A Description of the Database Application area and ER Model

A.1 Application Description

For this project, I have chosen to represent various camps throughout the United States of America. There are various entities within my project such as Camp Details, Camper Details, Cabin Details, Staff Details, Activity Details and Complaints and Feedback Details. For each camp there are a number of campers and a number of staff. There is one unit leader assigned to each cabin within the various camps. Each camp offers a number of activities depending on the session that the camper attends. Complaints can be made by the campers and when the issue has been resolved they can leave feedback to let the camp know how efficiently the complaint was handled.

There are 8 – relational models which I have chosen to represent and they are:

Camp Details: This table has all the information about each specific camp. The following attributes are listed: **Camp ID (Primary Key)**

Name,mobile,email,Number of Campers, Address (Country, State, City, Street)

Camper Details: This table has all the information on the campers attending each camp.

The following attributes are listed: **Camper ID (Primary Key)**

Name (First and Last names), DOB (date of birth), mobile, Emergency Contact Name, Emergency Contact Mobile, session, Address (Country, State, City, Street (home address)).

Staff Details: This table contains all the information about staff at each camp

The following attributes are listed:**Staff ID (Primary Key)**

Camp ID, Name (First and Last names), mobile, Emergency Contact Name, Emergency Contact Mobile, Session, Role, Email, Address.

Activity Details: This table contains all the information about the activities offered at each camp

The following attributes are listed: **Activity Type (Primary Key), Camp ID (Primary Key)**

Session, Number of Campers, Indoor/Outdoor.

Cabin Details: This table contains all the information about the cabins at each camp

The following attributes are listed: **Cabin ID (Primary Key), Camp ID (Primary Key)**

Occupancy

Camper Camp: This table contains all the information about which camper goes to which camp

The following attributes are listed: **Camper ID (Primary Key), Camp ID (Primary Key)**

Complaints: This table contains all the information about complaints made by campers

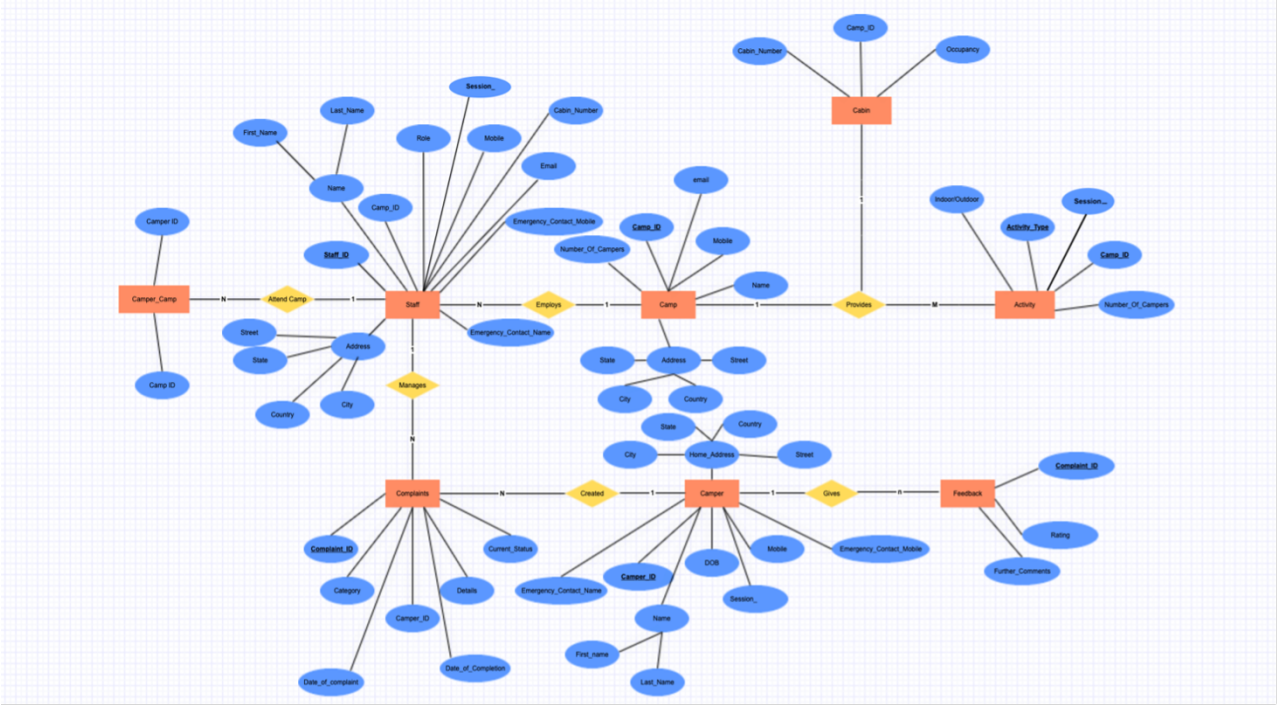
The following attributes are listed: **Complaint ID (Primary Key), Camper ID (Primary Key)**

Date of Complaint, Category, Current Status, Details, Date of Completion

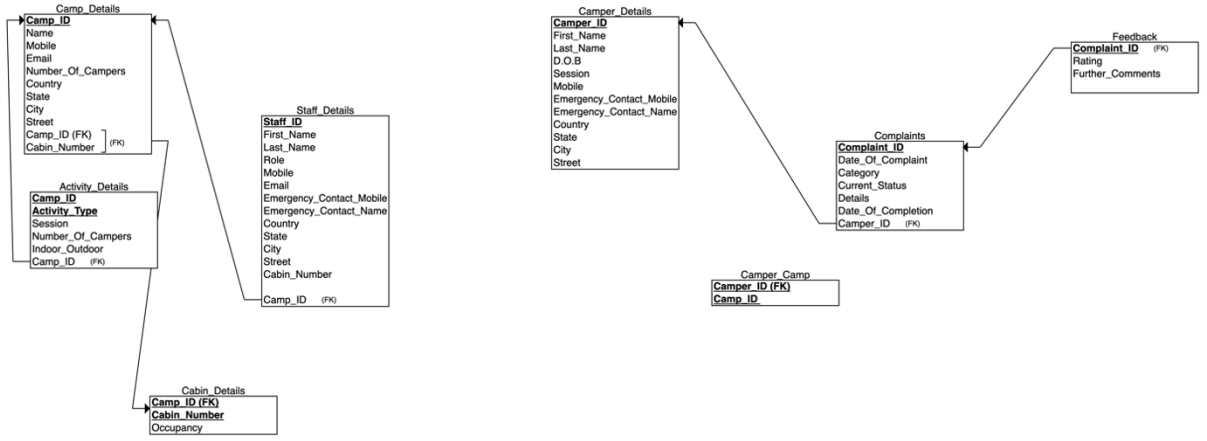
Feedback: This table contains all the information about feedback given by campers about the efficiency of their complaint being resolved.

The following attributes are listed:**Complaint ID (Primary Key)** Rating, Further Comments

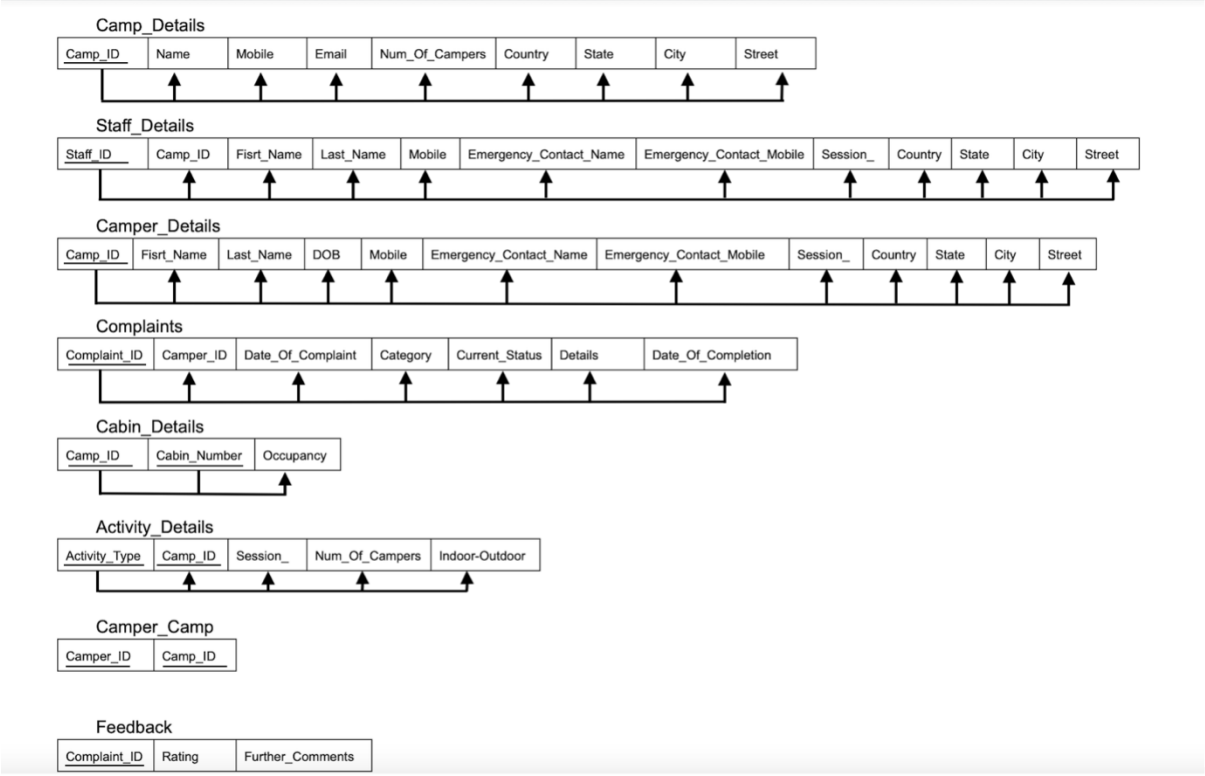
A.2 Entity Relationship Diagram



A.3 Mapping to Relational Schema



A.4 Functional Dependency Diagram



B Explanation of data and SQL Code

B.1 Explanation of one of the SQL Code for Creating one of your database Tables (including any constraints)

```
CREATE TABLE Camp_Details (
  `Camp_ID` int NOT NULL,
  `Name` varchar(100) NOT NULL,
  `Mobile` int NOT NULL,
  `Email` varchar(100) NOT NULL,
  `Num_of_Campers` int NOT NULL,
  `Country` varchar(100) NOT NULL,
  `State` varchar(100) NOT NULL,
  `City` varchar(100) NOT NULL,
  `Street` varchar(100) NOT NULL,
  PRIMARY KEY (`Camp_ID`),
  UNIQUE KEY `Camp_ID` (`Camp_ID`),
  UNIQUE KEY `Mobile` (`Mobile`),
  CHECK ((`Num_of_Campers` > 0))
);
```

I used the command 'CREATE TABLE' in order to create my table. In order to make sure that each camp had a different ID I made sure to use the command 'UNIQUE KEY' in order to avoid any ID duplicates.

I also have a 'CHECK' constraint which I used in order to make sure that the number of campers within the camp is bigger that 0. All of the primary and foreign keys are declared 'NOT NULL'.

B.2 Altering tables

```
265 •   ALTER TABLE Camper_Details
266       ADD Session_ INT NOT NULL
267         AFTER Emergency_Contact_Mobile;
```

When I created my camper details table I did not include the session and so later when I realized I need it in a certain position in the table, I was able to add it on using the 'ALTER' statement.

B.3 Trigger operations

```
245   -- Trigger for Cabin_Details Table
246
247 • CREATE TRIGGER Cabin_Occupancy
248   AFTER INSERT
249   ON Staff_Details
250   FOR EACH ROW
251   UPDATE Cabin_Details r
252   SET r.Occupancy = 'Occupied'
253   WHERE r.Cabin_Number = NEW.Cabin_Number;
254
255   -- Trigger for Cabin_Details Table
256
257 • CREATE TRIGGER Cabin_Vacancy
258   AFTER DELETE
259   ON Staff_Details
260   FOR EACH ROW
261   UPDATE Cabin_Details r
262   SET r.Occupancy = 'Vacant'
263   WHERE r.Cabin_Number = OLD.Cabin_Number;
```

```
229   -- Trigger for Complaints Table
230
231 • CREATE TRIGGER Decide_Status
232   BEFORE INSERT
233   ON Complaints
234   FOR EACH ROW
235   SET New.Current_Status = "Not-Completed";
236
237   -- Trigger for Complaints Table
238
239 • CREATE TRIGGER Decide_Date
240   BEFORE INSERT
241   ON Complaints
242   FOR EACH ROW
243   SET New.Date_Of_Complaint = SYSDATE();
```

I have a total of four triggers in my database. The triggers for my Cabin_Details table were created so that when a new member of staff is introduced into the database and they are given a cabin, then the occupancy will change to reflect this.

A trigger for my Complaints table are used to that when a new complaint is introduced, the not-completed status will appear until the complaint has been dealt with. The trigger regarding the date is used to show the date when a complaint has been made by a camper.

B.4 Creation of Views

```
268 •  CREATE VIEW unitleaders_cabins
269    AS
270    SELECT
271    First_Name AS 'Name',
272    Mobile,
273    Cabin_Number AS 'Cabin Number'
274    FROM
275    Staff_Details
276    WHERE
277    Staff_Details.Role_ = 'Unit Leader';
```

```
281 •  CREATE VIEW vacant_cabins
282    AS
283    SELECT
284    Camp_ID,
285    Cabin_Number
286    FROM Cabin_Details
287    WHERE Occupancy = 'Vacant';
```

I have two views in my database. The first view is the Unit Leaders Cabins View. Unit Leaders are in charge of Camp Counselors and so they are very important in the camp. This view lets me see the unit leaders' cabins, mobiles and names for easy access to them. Any issues that arise are brought to Unit Leaders and so it is vital that this information is easily accessible.

My second view is for any vacancies in cabins. This lets the camp know where to place new staff members when they get hired.

B.5 Populating a Tables

```
105 •  INSERT INTO Camp_Details VALUES (1,'Camp Mohawk',23444555,'mohawk@gmail.com',3000,'United States','Conneticut','Litchfield','Rato Avenue');
106 •  INSERT INTO Camp_Details VALUES (2,'Camp Kiki',23333225,'kiki@gmail.com',12000,'United States','New York','Manhatten','Allen Street');
107 •  INSERT INTO Camp_Details VALUES (3,'Camp Wakim',23283645,'wakim@gmail.com',7000,'United States','New York','Sataten Island','Crystal Avenue');
108 •  INSERT INTO Camp_Details VALUES (4,'Camp Falcon',23999876,'falcon@gmail.com',1000,'United States','Philadelphia','Harrisburg','Connie Street');
109 •  INSERT INTO Camp_Details VALUES (5,'Camp Hammok',24544258,'hammok@gmail.com',50000,'United States','California','San Jose','Market Street');
110 •  INSERT INTO Camp_Details VALUES (6,'Camp Tumble Weed',33524258,'tweed@gmail.com',13000,'United States','California','Los Angeles','Hanley Avenue');
```

I used the 'INSERT INTO' statement in order to populate all my tables. And all the values were inserted in order of the attributes in the create table section.

B.6 Retrieving information from the database

1. **SELECT** * **FROM** Activity_Details;

Activity_Type	Camp_ID	Session_	Number_Of_Campers	Indoor_Outdoor
Archery	4	1	500	Outdoor
Ariel Silks	2	1	10200	Outdoor
Fishing	3	2	4600	Outdoor
Golf	5	2	35000	Outdoor
Kayaking	5	1	44000	Outdoor
Music	6	2	11000	Outdoor
Painting	4	2	780	Indoor
Robotics	1	2	3000	Indoor
Surfing	2	2	11200	Outdoor
Swimming	1	1	2600	Outdoor
Swimming	3	1	2500	Indoor
Tennis	6	1	10500	Outdoor

'SELECT' statement is used to retrieve information from the database. The '*' is used to retrieve all the information.

1. **Joins Example**

```
1  SELECT
2  Camp_Details.Camp_ID,
3  Camp_Details.Name,
4  Staff_Details.Staff_ID,
5  Staff_Details.First_Name,
6  Staff_Details.Role_
7  FROM
8  Camp_Details,
9  Staff_Details
10 WHERE
11 Staff_Details.Role_ = 'Camp Counselor'
12  AND
13  Camp_Details.Camp_ID = 6;
```

Camp_ID	Name	Staff_ID	First_Name	Role_
6	Camp Tumble Weed	1	Hannah	Camp Counselor
6	Camp Tumble Weed	2	Katya	Camp Counselor
6	Camp Tumble Weed	6	Ruby	Camp Counselor
6	Camp Tumble Weed	7	Cirilla	Camp Counselor
6	Camp Tumble Weed	11	Logan	Camp Counselor
6	Camp Tumble Weed	12	Liam	Camp Counselor
6	Camp Tumble Weed	16	Jona	Camp Counselor
6	Camp Tumble Weed	17	Lia	Camp Counselor
6	Camp Tumble Weed	21	Jonas	Camp Counselor
6	Camp Tumble Weed	22	Caitlin	Camp Counselor
6	Camp Tumble Weed	26	Heidi	Camp Counselor
6	Camp Tumble Weed	27	Abby	Camp Counselor

B.7 Security commands

```
291 •  DROP ROLE IF EXISTS 'Unit Leader';
292 •  CREATE ROLE 'Unit Leader';
293 •  GRANT CREATE, DELETE ON Camper_Details TO 'Unit Leader';
294 •  GRANT UPDATE, SELECT ON Camper_Details TO 'Unit Leader' WITH GRANT OPTION;
295 •  GRANT DELETE, CREATE, UPDATE ON Cabin_Details TO 'Unit Leader' WITH GRANT OPTION;
296
297
298 •  DROP ROLE IF EXISTS Warden;
299 •  CREATE ROLE Warden;
300 •  GRANT CREATE, DELETE ON Camper_Details TO Warden;
301 •  GRANT UPDATE, SELECT ON Camper_Details TO Warden WITH GRANT OPTION;
302 •  GRANT DELETE, CREATE, UPDATE ON Cabin_Details TO Warden WITH GRANT OPTION;
```

In my database, only the Unit Leaders and Wardens of each camp have access to certain files about the campers. This ensures that the privacy of the campers is safe and that the data will not be compromised. The roles were assigned using the 'GRANT' statement.

If I wanted to remove them from being able to access the database I would use the below statement:

REVOKE DELETE ON Camper_Details **FROM** Warden;