Cristian Hendriks

CS 355

Professor Thompson

10/30/2020

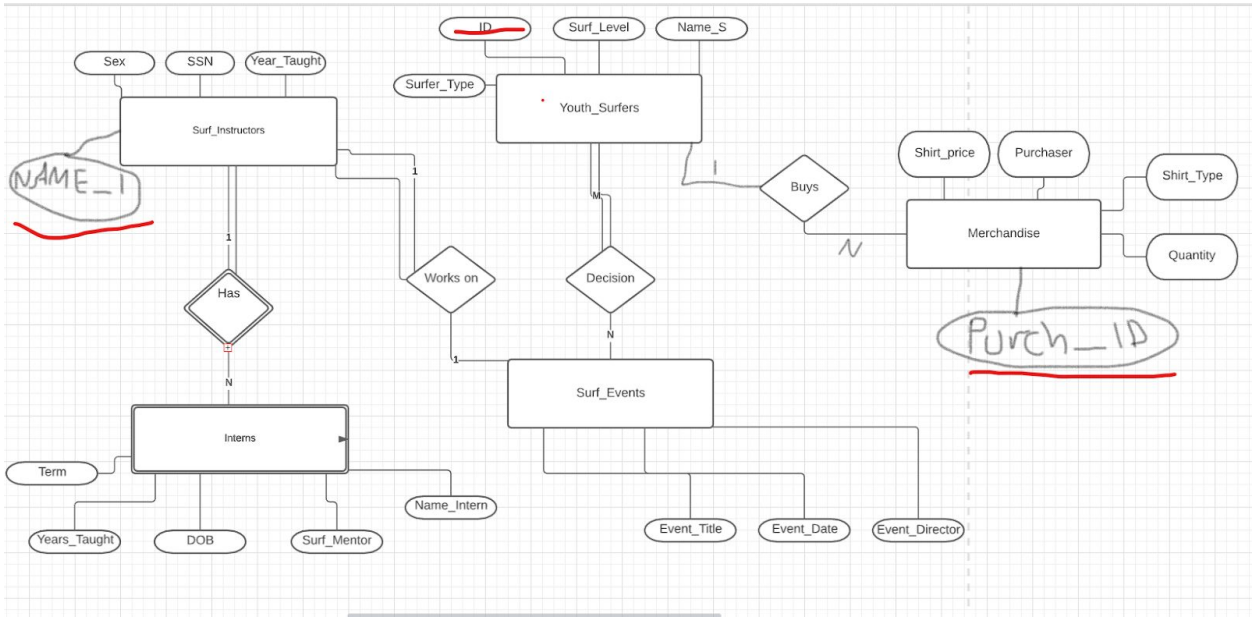
Surf Academy Database

The Goal of this database is two manage and hold information about a Surf Academy company. This database is meant to make the functionality of the company as a whole;easier for them. For example, keeping track of all their clients, employees and events. Our Quota is to ensure the safety, progress and development of all clients. The Database holds more than enough tuples of information concerning each person that is associated with the academy. Though there might be problems with organizing both events and teaching kids due to a large enrollment, the academy also has interns to help run and instruct the youth. Each instructor is assigned to an intern as a mentor, however not every instructor has a mentor. On the other hand every intern is assigned to an instructor. Therefore this makes the intern table a weak entity because an intern cannot exist unless he/she is assigned to a mentor. This is just a part of the learning progression of our interns. We believe that in order to get the best hands on experience they need a figure to look up to.

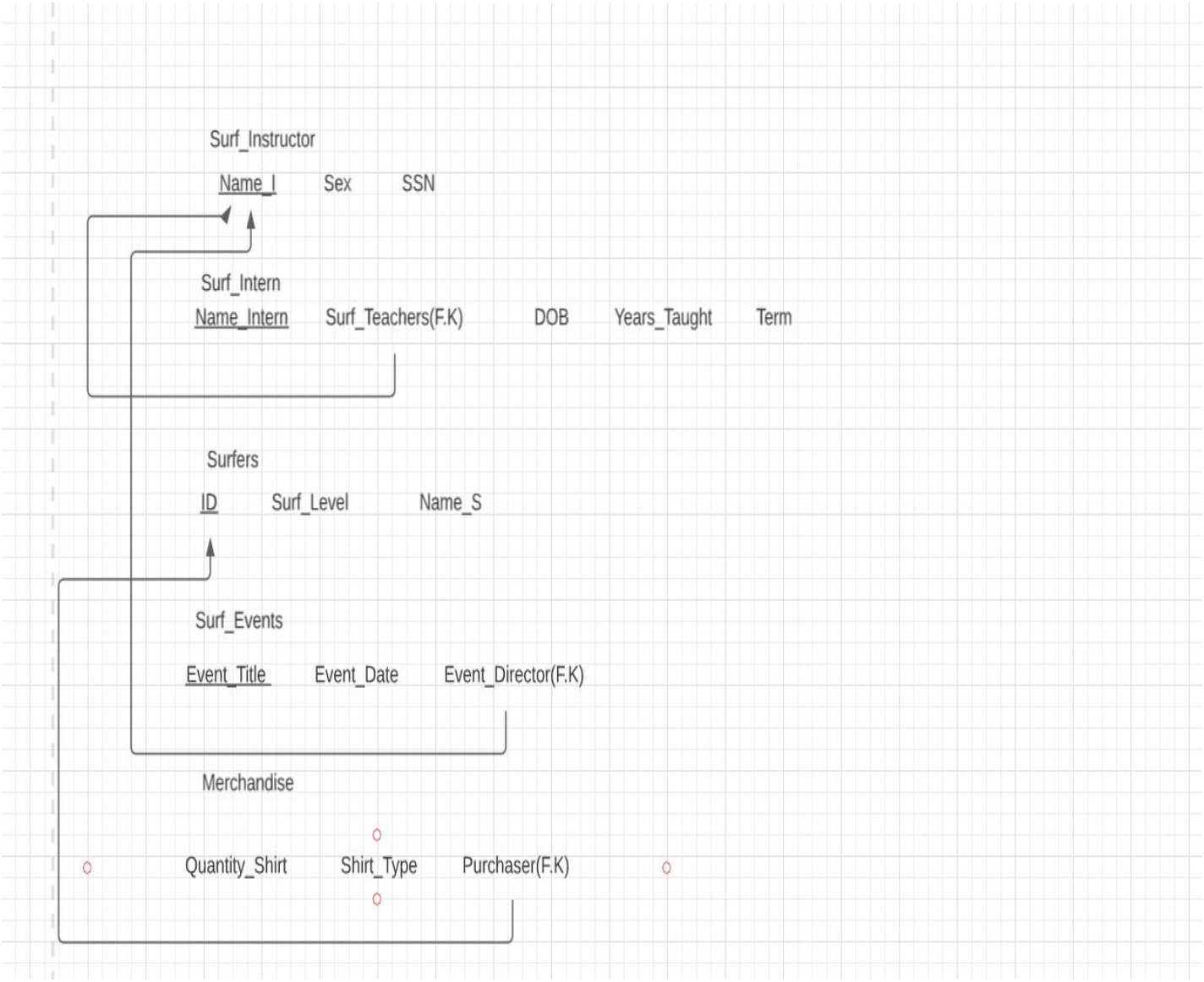
The students are identified in the database as “Surfers”, and at the beginning of their enrollment are assigned both a category of surf type, and a ranking score based on surfing skills. The reason we have this set in the database is to get a grasp on each surfers development, so at the end of their enrollment we can re-rank them and see their progression as a surfer. Instructing youth surfers isn’t the only focus of this business, we have all our employees(including interns) help construct seasonal events for the surfers. Each event has a

special name and date. The surf events are held to help and showcase the skills of the youth surfers to help them prepare for future events. Each event is directed by an instructor, so in the event table there is a foreign key referring back to the instructor table. Also there is a Merchandise table to keep tabs on all inventory items. This table has information about the quantity of items, types of shirts that are selling and the prices. However shirts can only be bought by surfers(students), and they can purchase multiple items. But not every shirt has to be bought or purchased by a surfer(student).

EDR(Sorry for messy drawn in lines I ran outta items in Lucid chart please don't mark me down!!)



Relationship Schema



- There is a Primary key in Merchandise Called “Purchased_ID”(Ran outta Text sorry)

Table 1: Surf_Instructors

Purpose: List basic information about the Surf Instructors in the academy

Attributes: Name_I, sex,SSN, Years_taught

Keys: Name_I is the Primary key in this table

MySQL: Create table Surf_Instructors

```
(

 NAME_I varchar (25) Primary Key,

sex varchar (7) check (sex in ('Male', 'Female')),

 SSN bigint Unique,

 Years_taught bigint

);
```

```
MariaDB [hendriks_cs355fl20]> Select * from Surf_Instructors;
+------------------+--------+----------+--------------+
| NAME_I           | sex    | SSN      | Years_taught |
+------------------+--------+----------+--------------+
| Alani Lopez      | Female |  3454335 |            2 |
| Armaan Shah      | Male   |   564533 |            7 |
| Audrey Castillo  | Female |    70975 |            3 |
| Crystal Martinez | Female |   981911 |            1 |
| Sean Hanlon      | Male   | 12345332 |            4 |
+------------------+--------+----------+--------------+
5 rows in set (0.001 sec)
```

Table 2

Table name: Surf_Instructors

Purpose: List information about the interns working at the surf academy

Attributes: Name_Intern, Surf_Teachers,DOB, Degrees, Term

Keys: Name_Interns(Primary key), Surf_Teachers(Foreign Key)

```
MYSQL: Create table Surf_Interns
(
Name_Intern Varchar (25) Primary Key,
Surf_Teachers varchar(25) ,
DOB Datetime Unique,
Degrees bigint,
Term varchar (10),
Foreign Key (Surf_Teachers)
                        References Surf_Instructors(Name_I)

);
```

```
MariaDB [hendriks_cs355fl20]> Select * from Surf_Interns;
+-----------------+------------------+---------------------+---------+--------+
| Name_Intern     | Surf_Teachers    | DOB                 | Degrees | Term   |
+-----------------+------------------+---------------------+---------+--------+
| Ben Huber       | Sean Hanlon      | 2000-02-12 10:34:09 |       2 | Spring |
| Eric Vela       | Crystal Martinez | 2000-06-12 10:34:09 |       0 | Spring |
| Jacob Aere      | Armaan Shah      | 1998-06-12 10:34:09 |       1 | Spring |
| Steven Hendriks | Audrey Castillo  | 2001-06-12 10:34:09 |       3 | Spring |
| Zack Haupt      | Alani Lopez      | 1997-06-12 10:34:09 |       0 | Fall   |
+-----------------+------------------+---------------------+---------+--------+
5 rows in set (0.001 sec)
```

Table 3

Table name: surfers

Attributes: ID, Surf_Level, Name_S, Surfer_type

Keys: ID(Primary key)

MYSQL: Create table surfers
(
ID bigint Primary Key,
Surf_Level bigint,
Name_S varchar (30),
Surfer_type varchar (25) check (Surfer_type in ('Short Boarder', 'Long Boarder'))
);

```
MariaDB [hendriks_cs355fl20]> Select * from surfers;
+-------+------------+----------------+---------------+
| ID    | Surf_Level | Name_S         | Surfer_type   |
+-------+------------+----------------+---------------+
| 12345 |          7 | Lucas Hendriks | Short Boarder |
| 33488 |          8 | Chance Bennet  | Short Boarder |
| 54321 |         14 | Roddy Rich     | Long Boarder  |
| 87190 |         17 | Tupac Shakur   | Short Boarder |
| 98765 |          6 | Frank Ocean    | Long Boarder  |
+-------+------------+----------------+---------------+
5 rows in set (0.000 sec)
```

Table 4

Table name: Surf_Events

Attributes: Event_Title, Event_Date, Event_Director

Keys: Event_Director(Foreign Key)

MYSQL: Create Table Surf_Events
(

Event_Title varchar(200),
Event_Date Datetime,
Event_Director varchar(30) Not null,
foreign key (Event_Director) references Surf_Instructors(Name_I)
);

```
MariaDB [hendriks_cs355fl20]> Select * from Surf_Events;
+------------------+---------------------+------------------+
| Event_Title      | Event_Date          | Event_Director   |
+------------------+---------------------+------------------+
| Spring PRO-AM    | 2020-03-11 10:34:00 | Armaan Shah      |
| Fall Festival    | 2020-05-11 11:30:00 | Sean Hanlon      |
| Holiday Classic  | 2000-02-12 10:34:09 | Alani Lopez      |
| Easter Special   | 2000-02-12 11:30:09 | Audrey Castillo  |
| Machodo Showcase | 2020-02-12 10:34:09 | Crystal Martinez |
+------------------+---------------------+------------------+
5 rows in set (0.001 sec)
```

Table 5

Table name: Merchandise

Attributes: Quantity_Shirts, Shirt_type,Shirt_Price, Purchaser

Keys: Purchase_ID(Primary key), Purchaser(Foreign key)

MySQL: Create Table Merchandise
(
Purchase_ID bigint primary key,
Quantity_Shirts bigint,
Shirt_type varchar(20) ,
Shirt_Price float,
Purchaser bigint,
Foreign Key (Purchaser)
References surfers(ID)

);

```
MariaDB [hendriks_cs355fl20]> select * from Merchandise;
+-------------+-----------------+--------------+-------------+-----------+
| Purchase_ID | Quantity_Shirts | Shirt_type   | Shirt_Price | Purchaser |
+-------------+-----------------+--------------+-------------+-----------+
|        7777 |               2 | Long Sleeve  |          10 |     12345 |
|        7778 |               4 | Short Sleeve |           5 |     87190 |
|        7779 |               6 | Long Sleeve  |          12 |     33488 |
+-------------+-----------------+--------------+-------------+-----------+
3 rows in set (0.001 sec)
```

Queries

<u>Querie_1:</u> Find a surfer for a competition

<u>Reason:</u> We want to be able to well categorize our surfers because not one surfer is really the same, so for a certain competition we want to find a surfer that is over a level 8 and a short Border for a competition.

<u>MYSQL:</u> Select * from surfers where Surf_Level >8 and Surfer_Type ='Short Boarder';

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	Surf_Level	Name_S	Surfer_type
87190	17	Tupac Shakur	Short Boarder
NULL	NULL	NULL	NULL

<u>Query 2:</u> Wanna have some information on surfers

<u>Reason:</u> For the purpose of finding the right instructor we need a query that will make a subquery to find the average years taught along with counting the years.

<u>MYSQL:</u>

```
Select Name_Intern, AVG(years_taught) from
(Select Name_Intern, count(*) as years_taught
from Surf_Interns) as yt ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name_Intern	AVG(years_taught)
Zack Haupt	5.0000

<u>Query 3:</u> Finding the right teacher based off qualification

<u>Reason:</u> For the sake of the best instruction some students will want a well qualified instructor, this query will pick out the instructor with the most experience

<u>MYSQL:</u>

```
SELECT  Degrees, Name_Intern
FROM Surf_Interns
GROUP BY Term
```

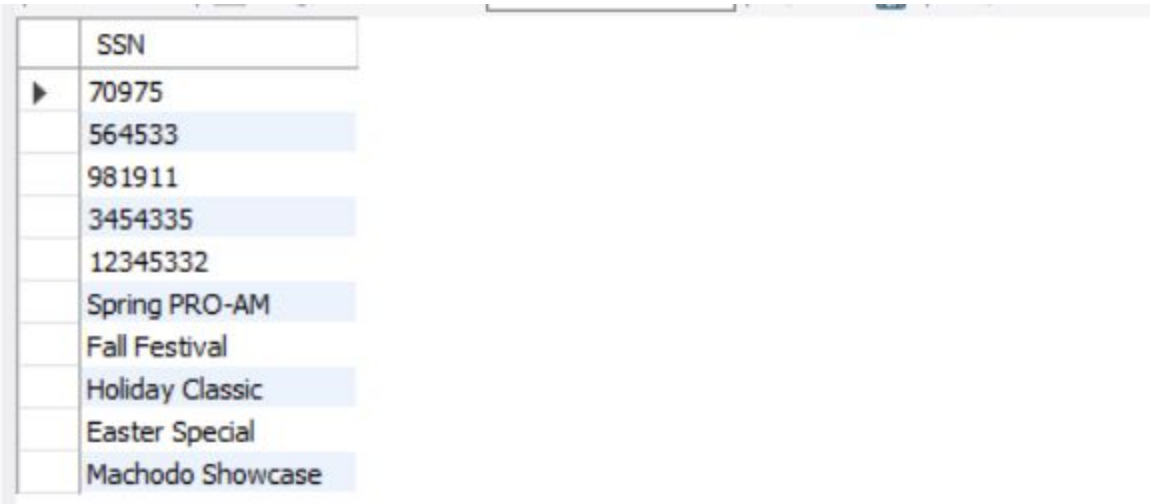
Having degrees;

Degrees	Name_Intern
2	Ben Huber
NULL	NULL

<u>Query 4:</u> show surf instructors to there events

<u>Reason:</u> If instructors want to view there assigned events they are planning this is the query

<u>MYSQL:</u> Select SSN from Surf_Instructors
Union All
Select Event_Title from Surf_Events;



SSN
70975
564533
981911
3454335
12345332
Spring PRO-AM
Fall Festival
Holiday Classic
Easter Special
Machodo Showcase

<u>Query 5:</u> Grabbing a specific surfer information

Reason: For security reasons if a parent or intern need to grab information about an surfer and his or her specific id base of there boarding style this is the query

MYSQL: select distinct ID from surfers where Surfer_type = 'Long Boarder';

```
MariaDB [hendriks_cs355fl20]> select distinct ID from surfers where Surfer_type
= 'Long Boarder';
+-------+
| ID    |
+-------+
| 12345 |
| 54321 |
| 98765 |
+-------+
3 rows in set (0.001 sec)
```

Query 6: Update surfers

Reason: If something changes with a surfer and his/her information, and they need an update in the database

MYSQL:
Update surfers
Set Surfer_type= 'Long Boarder'
Where ID= 12345;

```
MariaDB [hendriks_cs355fl20]> Update surfers
    -> Set Surfer_type= 'Long Boarder'
    -> Where ID= 12345;
Query OK, 1 row affected (0.001 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [hendriks_cs355fl20]>
```

\

Query 7: Join clause

Reason:Join all the same columns that are similar from each table by using a join statement.

MYSQL:

```
select Surf_Interns.Surf_Teachers, Surf_Events.Event_Director, Surf_Interns.Surf_Teachers
from Surf_Instructors
inner join Surf_Interns
On Surf_Instructors.NAME_I= Surf_Interns.Surf_Teachers
join Surf_Events
on Surf_Events.Event_Director=Surf_Interns.Surf_Teachers;
```

```
/* Join*/

select Surf_Interns.Surf_Teachers, Surf_Events.Event_Director, Surf_Interns.Surf_Teachers
from Surf_Instructors
inner join Surf_Interns
On Surf_Instructors.NAME_I= Surf_Interns.Surf_Teachers
join Surf_Events
on Surf_Events.Event_Director=Surf_Interns.Surf_Teachers;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Surf_Teachers	Event_Director	Surf_Teachers
Alani Lopez	Alani Lopez	Alani Lopez
Armaan Shah	Armaan Shah	Armaan Shah
Audrey Castillo	Audrey Castillo	Audrey Castillo
Crystal Martinez	Crystal Martinez	Crystal Martinez
Sean Hanlon	Sean Hanlon	Sean Hanlon

Stored Procedure: Grabs a certain surfer profile

Reason: If needed by the department this stored procedure will grab the surfer_Name and what kinda board they ride.

MYSQL: DROP procedure IF EXISTS `Find_Trainer`;
DELIMITER $$
USE `hendriks_cs355fl20`$$
CREATE PROCEDURE `Find_Trainer` ()
BEGIN
select Surfer_Type, Name_S from surfers where Surfer_Type = 'Long Boarder';

```
/*Procedure*/

DROP procedure IF EXISTS `Find_Trainer`;
DELIMITER $$
USE `hendriks_cs355fl20`$$
CREATE PROCEDURE `Find_Trainer` ()
BEGIN
select Surfer_Type, Name_S from surfers where Surfer_Type = 'Long Boarder';
END$$

/* Calling the procedure*/
call Find_Trainer();
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Surfer_Type	Name_S
Long Boarder	Roddy Rich
Long Boarder	Frank Ocean

View: Finding inventory

Reason: If needed to find certain inventory information this is the query to back track who bought the shirt, and also the max price for the shirt purchased

MYSQL:

```
/*View*/
drop view if exist 'Merch_info';
DELIMITER ;

CREATE OR REPLACE 'merch_info' AS
DELIMITER ;

/*CREATE VIEW `Merch_info` AS*/
select Shirt_Type, Purchaser, max(Shirt_Price) from Merchandise;
```

Function: Finding a ranking of a surfer

Reason: In order to find where a player stands within the surf world this function has an algorithm that will calculate a certain surfers ranking based on information within the database.

```
MySQL: DROP function IF EXISTS `Surf_Rank`;
DELIMITER $$
CREATE FUNCTION `Surf_Rank` (
Surf_Level Decimal(10,2))
RETURNS varchar(30)
BEGIN
Declare Surf_Rank varchar
(20);
if Surf_Level < 10 then
set Surf_Rank = 'beginner';
elseif (Surf_Level <= 10 )
then set Surf_rank = 'Advance';
end if;
RETURN Surf_rank;
END$$
DELIMITER ;
```

```
DROP function IF EXISTS `Surf_Rank`;
DELIMITER $$
CREATE FUNCTION `Surf_Rank` (
Surf_Level Decimal(10,2))
RETURNS varchar(30)
BEGIN
Declare Surf_Rank varchar
(20);
if Surf_Level < 10 then
set Surf_Rank = 'beginner';
elseif (Surf_Level <= 10 )
then set Surf_rank = 'Advance';
end if;
RETURN Surf_rank;
END$$
DELIMITER ;

select Surf_Rank(12);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Surf_Rank(12)
NULL