

# **FITNESS DATABASE MANAGEMENT SYSTEM**

**BY  
ROHITHA GUTTA**

# TABLE OF CONTENTS

BUSINESS APPLICATION DESCRIPTION

BUSINESS VALUE

PRIMARY USERS:

USE CASES

BUSINESS METRICS

LOGICAL MODEL – UML

PHYSICAL MODEL – DATA DICTIONARY

TRIGGERS

VIEWS

SQL – USE CASES QUERIES

SQL - BUSINESS METRICS QUERIES

PROJECT SUMMARY

## BUSINESS APPLICATION DESCRIPTION

As the famous proverb goes by the saying “Health is wealth”, achieving physical and mental fitness has become one of the primary goals of every person's life. Today, the gym and fitness industry are flourishing all over the world as a result of which it is heavily dependent on a large number of databases to function smoothly, be it the list of employee records, customer’s information, guest logins, member subscriptions, etc. This has given rise to an urgent need to capture the data in order to enable its ease of use for various operations that are involved. The key functionalities of our Fitness Data Management System are as follows:

- Providing qualified trainers and supervising their performance
- Customer payment management
- Analyzing and interpreting trends for marketing purpose
- Activity tracking of customers and assessment of their fitness level
- Providing various classes based on participation and interest
- Assessment of the fitness centre based on customer feedback

By providing error-free, secure, reliable and fast data management for these functionalities, we aim to make the fitness regime easier and promise to deliver personal services and special attention to its members to improve customer satisfaction and run the business effectively.

## BUSINESS VALUE

Provides oversight and management of day-to-day operations of the fitness facility, streamlining several backend duties that drain time and resources, thus making the management staff focus on their members.

## PRIMARY USERS

The users interacting with our Fitness Data Management System are as follows:

- **Customer:** Access information related to scheduled classes, trainer’s availability, facilities provided, activity tracker, etc.
- **Trainers:** Responsible for scheduling personal training sessions and member’s activity details.
- **Manager:** Access various information for managing employees and organizing schedules.
- **Accountant:** Access client payment and responsible for employee salary management to calculate revenue per member and gross profit margin.

## USE CASES

### CUSTOMER

- Subscribe for membership or enroll in sessions of one or more classes
- Choose a personal trainer depending on the availability
- View his activity records and membership details

### TRAINER

- Add and view his availability
- View and update his personal details
- View the member's activity records and track progress

### MANAGER

- Organize schedules for trainer and staff
- Layoff trainers based on their performance track
- Review performance and declare bonus
- Manage employee leave

### ACCOUNTANT

- Manage monthly finances
- Report business performance over time to Manager
- Responsible for payroll management

## BUSINESS METRICS

### 1) Members Vs Non-members:

This metric helps bifurcate between the customers who have member subscriptions and those who have enrolled in sessions of one or more classes.

#### Why is it important?

This metric gives us the overall information which can be used to focus on improving our programs as we can see that the customers enrolled for the classes are far less than the ones subscribed for our membership plans. Since customers are the source of revenue for us, increasing our customer base would be our goal and these numbers will give us a sense of direction to meet our targets.

### 2) Early Stage Usage:

Measures how often new members use their memberships during the initial months of signing on with a fitness center.

#### Why is it important?

This will give insights to the fitness centre management for member retention. The more they use it during the initial months the more likely they are to stay as members for a longer period.

### **3) Membership Usage:**

Measures the frequency each member uses their fitness membership.

#### Why is it important?

This metric is important for fitness centers to track because if members use a fitness center on a regular basis, they are more likely to continue their membership with the establishment.

If member usage is low for specific members, the club can use this information to identify ways to get members to use their membership more frequently.

### **4) Lead Source:**

Leads are the number of people who are interested in your fitness centre . In order to reach out to these people, you need to figure out your marketing strategies. Tracking lead source allows you to intelligently build your marketing efforts.

#### Why is it important?

When you monitor these sources, you can see which marketing channels work the best. This will provide useful insights for the marketing team to analyze which channels work the best and which ones need extra effort.

### **5) Member Lifetime Value:**

Refers to the value each member is worth over the entire period of their membership. In order to affect this metric, you can increase retention rates, increase prices, offer additional member services, lower your acquisition costs, etc.

#### Why is it important?

This metric is important because it helps a fitness center estimate how much they can spend on customer retention programs and advertising.

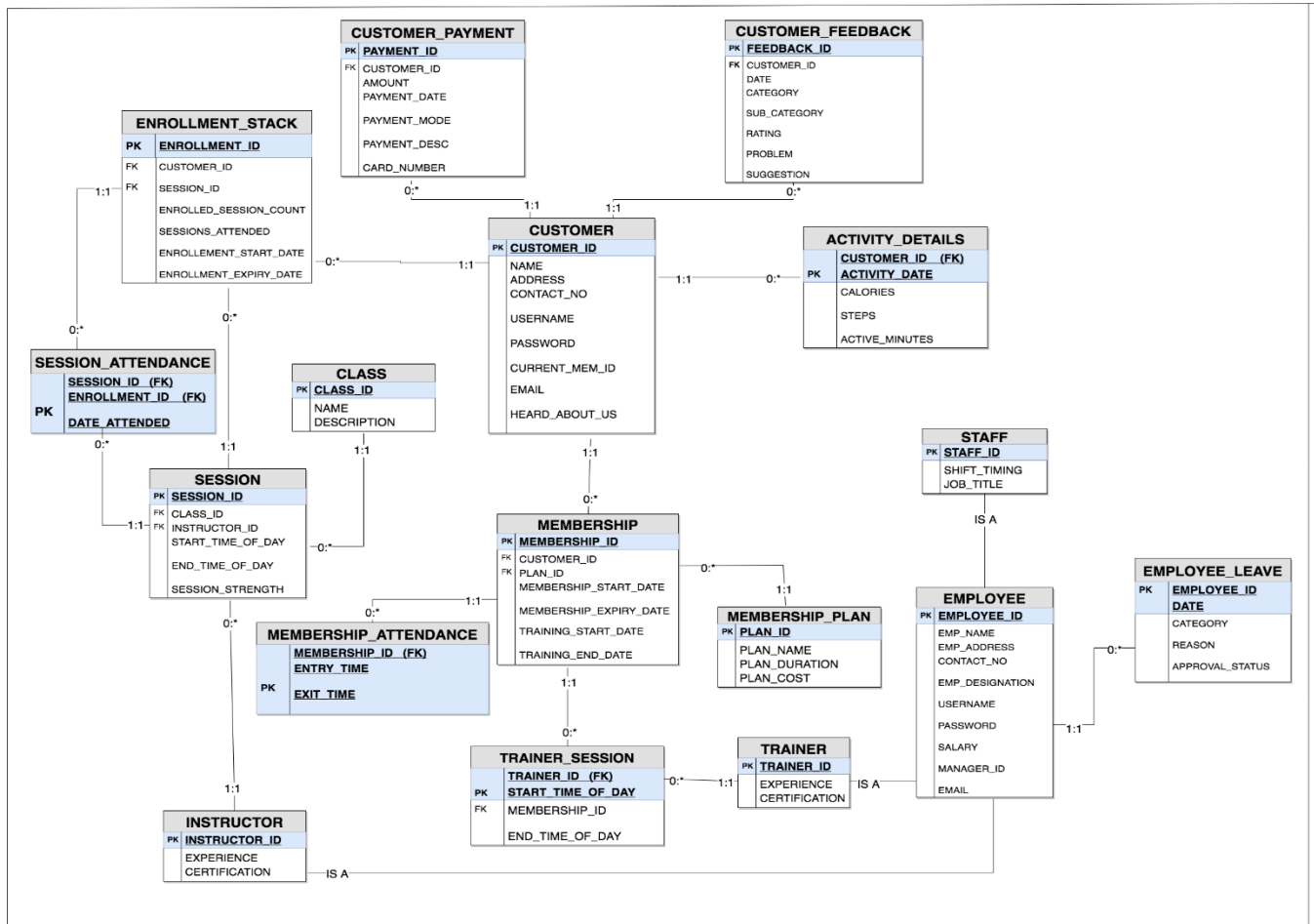
### **6) Customer Satisfaction:**

Measures the level of satisfaction of fitness center customers. The feedback given by the customers include quality of service, availability of equipment, cleanliness, trainer's , etc. It includes overall fitness rating and trainer performance.

#### Why is it important?

It provides actionable data on how fitness centers can improve their services. If the satisfaction is high, center's may have an easier time retaining members. It will also help in bring in new members through current member referrals. Based on the feedback received, average rating for each trainer is calculated. This will give useful insights on trainer's productivity which in turn can we used to add/layoff trainers.

## LOGICAL MODEL – UML



## PHYSICAL MODEL – DATA DICTIONARY

## 1. ACTIVITY DETAILS

NAME	TYPE	CONSTRAINT
W_CUSTOMER_ID	INT	PRIMARY KEY, FOREIGN KEY CUSTOMER(CUSTOMER_ID)
DATE	DATETIME	PRIMARY KEY
CALORIES	FLOAT	
STEPS	INT	
ACTIVE_MINUTES	VARCHAR (45)	

## 2. CLASS

NAME	TYPE	CONSTRAINT
CLASS_ID	INT	PRIMARY KEY
NAME	VARCHAR (45)	NOT NULL
DESCRIPTION	VARCHAR (500)	NOT NULL

## 3. CUSTOMER

NAME	TYPE	CONSTRAINT
CUSTOMER_ID	INT	PRIMARY KEY
NAME	VARCHAR (45)	NOT NULL
ADDRESS	VARCHAR (100)	
CONTACT_NO	VARCHAR (45)	NOT NULL
USERNAME	VARCHAR (45)	NOT NULL
PASSWORD	VARCHAR (100)	NOT NULL
CURRENT_MEM_ID	INT	FOREIGN KEY MEMBERSHIP(MEMBERSHIP_ID)
EMAIL	VARCHAR (45)	NOT NULL

## 4. CUSTOMER FEEDBACK

NAME	TYPE	CONSTRAINT
FEEDBACK_ID	INT	PRIMARY KEY
CUSTOMER_ID	INT	FOREIGN KEY CUSTOMER(CUSTOMER_ID) NOT NULL
DATE	DATE	NOT NULL
CATEGORY	VARCHAR (45)	NOT NULL
SUB_CATEGORY	VARCHAR (45)	NOT NULL
RATING	INT	NOT NULL
PROBLEM	VARCHAR (45)	
SUGGESTION	VARCHAR (45)	

## 5. CUSTOMER\_PAYMENT

NAME	TYPE	CONSTRAINT
PAYMENT_ID	INT	PRIMARY KEY
CUSTOMER_ID	INT	FOREIGN KEY CUSTOMER(CUSTOMER_ID) NOT NULL
AMOUNT	VARCHAR (45)	NOT NULL
PAYMENT_DATE	DATE	NOT NULL
PAYMENT_MODE	VARCHAR (45)	NOT NULL
PAYMENT_DESC	VARCHAR (45)	NOT NULL
CARD_NUMBER	VARCHAR (45)	

## 6. EMPLOYEE

NAME	TYPE	CONSTRAINT
EMPLOYEE_ID	INT	PRIMARY KEY
EMP_NAME	VARCHAR (45)	NOT NULL
EMP_ADDRESS	VARCHAR (45)	
CONTACT_NO	VARCHAR (45)	NOT NULL
EMP_DESIGNATION	VARCHAR (45)	NOT NULL
USERNAME	VARCHAR (45)	NOT NULL
PASSWORD	VARCHAR (45)	NOT NULL
SALARY	VARCHAR (45)	NOT NULL
MANAGER_ID	INT	
EMAIL	VARCHAR (45)	

## 7. EMPLOYEE LEAVE

NAME	TYPE	CONSTRAINT
EMPLOYEE_ID	INT	PRIMARY KEY, FOREIGN KEY EMPLOYEE(EMPLOYEE_ID)
DATE	DATE	PRIMARY KEY



CATEGORY	VARCHAR (45)	
REASON	VARCHAR (45)	
APPROVAL_STATUS	VARCHAR (45)	

## 8. ENROLLMENT\_STACK

NAME	TYPE	CONSTRAINT
ENROLLMENT_ID	INT	PRIMARY KEY
CUSTOMER_ID	INT	FOREIGN KEY CUSTOMER (CUSTOMER_ID) NOT NULL
SESSION_ID	INT	FOREIGN KEY SESSION (SESSION_ID) NOT NULL
ENROLLED_SESSION_COUNT	INT	NOT NULL
SESSIONS_ATTENDED	INT	
ENROLLMENT_START_DATE	DATE	NOT NULL
ENROLLMENT_EXPIRY_DATE	DATE	NOT NULL

## 9. INSTRUCTOR

NAME	TYPE	CONSTRAINT
INSTRUCTOR_ID	INT	PRIMARY KEY, FOREIGN KEY EMPLOYEE (EMPLOYEE_ID)
EXPERIENCE	VARCHAR (45)	
CERTIFICATION	VARCHAR (45)	

## 10. MEMBER ATTENDANCE

NAME	TYPE	CONSTRAINT
MEMBERSHIP_ID	INT	PRIMARY KEY, FOREIGN KEY MEMBER(MEMBERSHIP_ID)
ENTRY_TIME	DATETIME	PRIMARY KEY
EXIT_TIME	DATETIME	PRIMARY KEY

## 11. MEMBERSHIP

NAME	TYPE	CONSTRAINT
MEMBERSHIP_ID	INT	PRIMARY KEY
CUSTOMER_ID	INT	FOREIGN KEY CUSTOMER(CUSTOMER_ID) NOT NULL
PLAN_ID	INT	FOREIGN KEY MEMBERSHIP_PLAN(PLAN_ID) NOT NULL
MEMBERSHIP_START_DATE	DATE	NOT NULL
MEMBERSHIP_EXPIRY_DATE	DATE	NOT NULL
TRAINING_START_DATE	DATE	
TRAINING_END_DATE	DATE	

## 12. MEMBERSHIP PLAN

NAME	TYPE	CONSTRAINT
PLAN_ID	INT	PRIMARY KEY
PLAN_NAME	VARCHAR (45)	NOT NULL
PLAN_DURATION	INT	NOT NULL
PLAN_COST	FLOAT	NOT NULL

## 13. SESSION

NAME	TYPE	CONSTRAINT
SESSION_ID	INT	PRIMARY KEY
CLASS_ID	INT	FOREIGN KEY CLASS(CLASS_ID) NOT NULL
INSTRUCTOR_ID	INT	FOREIGN KEY INSTRUCTOR(INSTRUCTOR_ID) NOT NULL
START_TIME_OF_DAY	TIME (2)	NOT NULL
END_TIME_OF_DAY	TIME (2)	NOT NULL
SESSION_STRENGTH	INT	NOT NULL

#### 14. SESSION ATTENDANCE

NAME	TYPE	CONSTRAINT
SESSION_ID	INT	PRIMARY KEY, FOREIGN KEY SESSION(SESSION_ID)
ENROLLMENT_ID	INT	PRIMARY KEY, FOREIGN KEY ENROLLMENT(ENROLLMENT_ID)
DATE_ATTENDED	DATETIME	PRIMARY KEY

#### 15. STAFF

NAME	TYPE	CONSTRAINT
STAFF_ID	INT	PRIMARY KEY, FOREIGN KEY EMPLOYEE(EMPLOYEE_ID)
SHIFT_TIMING	VARCHAR (45)	NOT NULL
JOB_TITLE	VARCHAR (45)	NOT NULL

#### 16. TRAINER

NAME	TYPE	CONSTRAINT
TRAINER_ID	INT	PRIMARY KEY, FOREIGN KEY EMPLOYEE(EMPLOYEE_ID)
EXPERIENCE	VARCHAR (100)	
CERTIFICATION	VARCHAR (45)	

#### 17. TRAINER SESSION

NAME	TYPE	CONSTRAINT
TRAINER_ID	INT	PRIMARY KEY, FOREIGN KEY TRAINER(TRAINER_ID)
START_TIME_OF_DAY	TIME (2)	PRIMARY KEY
END_TIME_OF_DAY	TIME (2)	NOT NULL
MEMBERSHIP_ID	INT	FOREIGN KEY MEMBERSHIP (MEMBERSHIP_ID)

## TRIGGERS

**1. It will update the Current membership id in the customer table whenever a new membership id is added to the membership table for that customer.**

```
DELIMITER $$
CREATE TRIGGER CUST_MEM_UPDATE
AFTER INSERT ON MEMBERSHIP FOR EACH ROW
BEGIN
IF NEW.MEMBERSHIP_EXPIRY_DATE >= SYSDATE()
THEN
UPDATE CUSTOMER
SET CURRENT_MEM_ID = NEW.MEMBERSHIP_ID
where NEW.CUSTOMER_ID = CUSTOMER.CUSTOMER_ID;
END IF;
END
$$ DELIMITER
```

**2. This will update the session attended value for each customer in the Enrolment Stack table based on every session attended by the customer (tracked in Session\_Attendance table)**

```
DELIMITER $$
CREATE TRIGGER UPDATE_SESSION_ATTENDED
AFTER INSERT ON SESSION_ATTENDANCE FOR EACH ROW
BEGIN
UPDATE ENROLLMENT_STACK
SET SESSIONS_ATTENDED = SESSIONS_ATTENDED + 1
where NEW.ENROLLMENT_ID = ENROLLMENT_STACK.ENROLLMENT_ID AND
NEW.SESSION_ID = ENROLLMENT_STACK.SESSION_ID
and ENROLLMENT_STACK.ENROLLED_SESSION_COUNT >=
ENROLLMENT_STACK.SESSIONS_ATTENDED;
END
$$ DELIMITER
```

## VIEWS

### **VIEW NAME : EARLY\_STAGE\_USAGE**

```
CREATE VIEW EARLY_STAGE_USAGE AS SELECT MA.MEMBERSHIP_ID,
R1.MEMBERSHIP_START_DATE, R1.MEMBERSHIP_EXPIRY_DATE,
30 AS 'TOTAL NO. OF DAYS', COUNT(ENTRY_TIME) AS 'DAYS_ATTENDED'
FROM (SELECT * FROM MEMBERSHIP WHERE CUSTOMER_ID IN (SELECT CUSTOMER_ID
FROM CUSTOMER WHERE CURRENT_MEM_ID IS NOT NULL)) R1, MEMBER_ATTENDANCE
MA
WHERE R1.MEMBERSHIP_ID=MA.MEMBERSHIP_ID AND
MA.ENTRY_TIME>=R1.MEMBERSHIP_START_DATE AND
```

```
EXIT_TIME<=date_add(MEMBERSHIP_START_DATE, interval 30 day)
GROUP BY R1. MEMBERSHIP_ID
ORDER BY MA.MEMBERSHIP_ID;
```

#### **VIEW NAME : MEMBERSHIP\_USAGE**

```
CREATE VIEW MEMBERSHIP_USAGE AS SELECT R.MEMBERSHIP_ID,
COUNT(MA.ENTRY_TIME), (datediff(sysdate(),R.MEMBERSHIP_START_DATE))
FROM (SELECT * FROM MEMBERSHIP WHERE CUSTOMER_ID IN (SELECT CUSTOMER_ID
FROM CUSTOMER WHERE CURRENT_MEM_ID IS NOT NULL))R, MEMBER_ATTENDANCE
MA WHERE R.MEMBERSHIP_ID=MA.MEMBERSHIP_ID
GROUP BY MEMBERSHIP_ID;
```

#### **VIEW NAME: CURRENT\_MEMBERS**

```
CREATE VIEW CURRENT_MEMBERS AS SELECT * FROM MEMBERSHIP
WHERE CUSTOMER_ID IN (SELECT CUSTOMER_ID
FROM CUSTOMER WHERE CURRENT_MEM_ID IS NOT NULL);
```

#### **VIEW NAME: CUSTOMER\_OVERALL\_RATING**

```
CREATE VIEW CUSTOMER_OVERALL_RATING AS SELECT R1.MONTH, R1.RATING,
COUNT(*) FROM (SELECT *, MONTHNAME(DATE) AS MONTH
FROM CUSTOMER_FEEDBACK WHERE EXTRACT(YEAR FROM DATE)= 2018) R1
WHERE R1.CATEGORY = 'FITNESS CENTRE' AND R1.SUB_CATEGORY='OVERALL'
GROUP BY R1.MONTH, R1.RATING;
```

#### **VIEW NAME : CUSTOMER\_TRAINER\_RATING**

```
CREATE VIEW CUSTOMER_TRAINER_RATING AS SELECT SUB_CATEGORY, AVG(RATING)
AS AVG_RATING FROM CUSTOMER_FEEDBACK
WHERE CATEGORY='TRAINER' AND EXTRACT(YEAR FROM DATE) = 2018
GROUP BY SUB_CATEGORY
ORDER BY SUB_CATEGORY;
```

#### **VIEW NAME : CUSTOMER\_CLASS\_RATING**

```
CREATE VIEW CUSTOMER_CLASS_RATING AS SELECT SUB_CATEGORY, AVG(RATING)
AS AVG_RATING FROM CUSTOMER_FEEDBACK
WHERE CATEGORY='CLASS' AND EXTRACT(YEAR FROM DATE) = 2018
GROUP BY SUB_CATEGORY
ORDER BY SUB_CATEGORY;
```

#### **VIEW NAME : CUSTOMER\_ENROLLED\_FOR\_ALL\_SESSIONS**

```
CREATE VIEW CUSTOMER_ENROLLED_COUNT AS SELECT C.NAME AS CLASS_NAME,
R1.SESSION_ID, R1.CUSTOMERS_COUNT FROM CLASS C, SESSION S, (SELECT SESSION_ID,
COUNT(*) AS CUSTOMERS_COUNT FROM ENROLLMENT_STACK
WHERE ENROLLMENT_EXPIRY_DATE>=SYSDATE()
GROUP BY SESSION_ID) R1
WHERE R1.SESSION_ID=S.SESSION_ID AND S.CLASS_ID=C.CLASS_ID;
```

# SQL – USE CASES QUERIES

## CUSTOMER

1. Customer can register through website. On registration, customer can login by providing the username and password.

INSERT INTO CUSTOMER (CUSTOMER\_ID, NAME, ADDRESS, CONTACT\_NO, EMAIL, USERNAME, PASSWORD, CURRENT\_MEM\_ID) VALUES (100,'KARL JABLONSKI', '305 - 14TH AVE SUITE', '4089628421', 'ABC@GMAIL.COM', 'KJABLONSKI', 'K&JBLON8524',2010)

Before new customer registration:

CUSTOMER_ID	NAME	ADDRESS	CONTACT_NO	USERNAME	PASSWORD	CURRENT_MEM_ID	EMAIL
62	Hilda Dare	1810 Nellie Forest Bailevber...	778-698-3584x435	offriesen	a509141c4ca9c0d6c71961d9f373fd3798dd550e	1080	monte.b
63	Yoshiko Bover	214 Erika Wells Haenesmout...	(496)713-2982x746	bruce42	213246077fa0a46ed3d79908c67f578066bc25db	1080	deondre
64	Ismael Harber	6976 Koeloin Extensions Runt...	630-827-8732	monserrat.zboncak	8cd48097378235beb51dd056a94d93633df9b07	1080	thea.ma
65	Dr. Misael Feil DDS	9719 Armstrong Brook Suite 2...	00650953784	maeve.swaniawski	bef4e7d57dac848a942b948ca189c61df60a9e3d	1080	lizeth02
66	Mr. Fred Stroman	1595 Cartwright Wall Aot. 18...	123.619.5532x4700	valentine.hintz	50d4f14dc1d608ae153de216e721a5d9e7d05ce9	1080	zhowe8
67	Erika Hills	94692 Ezequiel Causewav Ao...	1-797-789-4764x085	chad.lubowitz	a5fbc0881f0bc4e53465d5a60b3cb666332ac91f	1081	aharvev
68	Brandt Hamill PhD	085 Timothy Mall Vonland. HI...	312-836-4205x1740	roaahn.dorothea	5fb5e9b46bfdba7aaac2fb9e02cd7bfc5062279f	1081	bart.bei
69	Monica Jenkins	130 Shanahan Neck Aot. 276...	(336)960-6953	block.desmond	5b712525925ed06f62b2d20870c4b45a81823506	1082	abeche
70	Eva Remoel	21058 Osinski Summit Suite 9...	1-786-360-9041x890	zmarvin	6f39c4b2f2975b80d29ca07b9763fa6924dfdfc8	1082	felton.h
71	Treva Stoltenberg	9294 Trent Drives Aot. 981 S...	02730130442	rubve.conrov	c4347cc348ce73a8fcd1a111095997cca0ed4973	1083	dennis9
72	Gloria Harber	8125 Bover Ford Suite 493 M...	047.210.4997x57506	luelwitz.carrie	aebd22aaaf8a00e9e7eb566a9b90984161e8199a	1083	crist.am
73	Clementine Yost PhD	30475 Carole Island Suite 814...	+41(4)1147422917	martine54	8ddfed2ac9c723c785d8dc505c791e19e9f42a91	1084	harmon
74	Prof. Gordon Abbott	267 Brenna Cove Lake Cade...	1-203-680-5797x418	nathan46	1d4097168079eaeaf95beef0f2783ba224cebc78	1084	amv578
75	Prof. Austyn Gleichne...	53925 Bradlev Court Emileshir...	1-586-652-4970x075	burnice.abbott	ec319ff3bd8e17465438abe61350b705a666996b	1085	leuschke

After new customer registration:

CUSTOMER_ID	NAME	ADDRESS	CONTACT_NO	USERNAME	PASSWORD	CURRENT_MEM_ID	EMAIL	HEARD_AB
90	Pearline Schroeder III	2254 Watsica Foroes Aot. 68...	(493)604-5663	ahmed87	27bf4e68022e67829f9c40dd37837234f4a7a5e	1075	sister46@example.oro	1085
51	Mr. Montana Fahev	0036 Schroeder Shores South...	1-382-613-6628	rhianna32	665941aa85fcdcb86334202d91172934293311c	1076	lowe.brandon@example.com	SOCIAL MEET
52	Kavlin Ferrv	553 Arlene Lodoe North Trev...	(612)380-8294	nikolas87	1e53b4fa103dd6882418ff6f9ac8ec8cf3d450d0	1085	alfredo.kozev@example.oro	1085
53	Keeaan Hills	2260 Elrov Center Suite 588 L...	288.759.6938	tristan85	559c792403271c33be1d8e41557e6eae47a896c2	1085	dante45@example.com	1085
54	Tess Nikolaus	2119 Gladvice Ramo New Gab...	+41(1)7018761945	chuel	a27f9c8422b621a8c77414ac97ad7d4e90a1d2f4	1085	ierkins.brandvn@example.com	REFERRAL
55	Gunner Schowalter	6321 Mandv Causewav Aot. ...	272-994-2339x3395	finn.d'amore	0bfdd03c554c88f8542cb60edd4fc3755b5f646f3	1077	outmann.trent@example.net	1085
56	Mia Goodwin	80527 Johathan Island Suite ...	117-167-9784x62284	manuela.kshlerin	32970bd9d39316fee97b86593802782d91bbd6f83	1077	tohnoad76@example.net	REFERRAL
57	Brailo Grant	114 Manuela Ramo Millsvie...	+45(3)2410315113	toro.adalberto	3568b0ffbbdf3a9ccc7889ce93f696a617b43a7	1085	irma.tilman@example.com	1085
58	Abbioal Zboncak DVM	74388 Rick Hills West Ashlev...	964-806-1388x638	melissa.cartwright	dbca756de06d83b13eb28c36bc7fa7de7c5fa6a	1078	keven.welch@example.net	SOCIAL MEET
59	Jace Mante	14246 Oswaldo Green Aot. 6...	421-895-7577x9940	susan.larson	cf48e7f15e4d0a8ab4b96f7d12c693b930dba63d	1085	xzavier92@example.net	1085
60	Gerhard Boehm	8862 Monahan Villaoe East Ne...	02531310035	aida.bosco	566c94fcb8c95689ecb5d52021de28fb4bb0707	1085	acummerata@example.com	ADVERTISEP
61	Dave Veum	77509 Bauch Islands Meadha...	025-860-4916x2561	cassandra.sanford	4302dc99540616cbcb6924fc6c63fb00d0379243	1079	loanne42@example.oro	1085
62	Hilda Dare	1810 Nellie Forest Bailevber...	778-698-3584x435	offriesen	a509141c4ca9c0d6c71961d9f373fd3798dd550e	1080	monte.bins@example.com	ADVERTISEP
63	Yoshiko Bover	214 Erika Wells Haenesmout...	(496)713-2982x746	bruce42	213246077fa0a46ed3d79908c67f578066bc25db	1080	deondre.23@example.com	1085
64	Ismael Harber	6976 Koeloin Extensions Runt...	630-827-8732	monserrat.zboncak	8cd48097378235beb51dd056a94d93633df9b07	1080	thea.maoio@example.net	ADVERTISEP
65	Dr. Misael Feil DDS	9719 Armstrong Brook Suite 2...	00650953784	maeve.swaniawski	bef4e7d57dac848a942b948ca189c61df60a9e3d	1080	lizeth02@example.oro	1085
66	Mr. Fred Stroman	1595 Cartwright Wall Aot. 18...	123.619.5532x4700	valentine.hintz	50d4f14dc1d608ae153de216e721a5d9e7d05ce9	1080	zhowe8@example.com	SOCIAL MEET
67	Erika Hills	94692 Ezequiel Causewav Ao...	1-797-789-4764x085	chad.lubowitz	a5fbc0881f0bc4e53465d5a60b3cb666332ac91f	1081	aharvev@example.com	1085
68	Brandt Hamill PhD	085 Timothy Mall Vonland. HI...	312-836-4205x1740	roaahn.dorothea	5fb5e9b46bfdba7aaac2fb9e02cd7bfc5062279f	1081	bart.beattv@example.net	REFERRAL
69	Monica Jenkins	130 Shanahan Neck Aot. 276...	(336)960-6953	block.desmond	5b712525925ed06f62b2d20870c4b45a81823506	1082	abechtelar@example.net	1085
70	Eva Remoel	21058 Osinski Summit Suite 9...	1-786-360-9041x890	zmarvin	6f39c4b2f2975b80d29ca07b9763fa6924dfdfc8	1082	felton.hibert@example.com	SOCIAL MEET
71	Treva Stoltenberg	9294 Trent Drives Aot. 981 S...	02730130442	rubve.conrov	c4347cc348ce73a8fcd1a111095997cca0ed4973	1083	dennis97@example.oro	1085
72	Gloria Harber	8125 Bover Ford Suite 493 M...	047.210.4997x57506	luelwitz.carrie	aebd22aaaf8a00e9e7eb566a9b90984161e8199a	1083	crist.america@example.oro	SOCIAL MEET
73	Clementine Yost PhD	30475 Carole Island Suite 814...	+41(4)1147422917	martine54	8ddfed2ac9c723c785d8dc505c791e19e9f42a91	1084	harmonv87@example.oro	1085
74	Prof. Gordon Abbott	267 Brenna Cove Lake Cade...	1-203-680-5797x418	nathan46	1d4097168079eaeaf95beef0f2783ba224cebc78	1084	amv57@example.oro	REFERRAL
75	Prof. Austyn Gleichne...	53925 Bradlev Court Emileshir...	1-586-652-4970x075	burnice.abbott	ec319ff3bd8e17465438abe61350b705a666996b	1085	leuschke.odell@example.net	1085
100	Karl Jablonski	305 - 14th Ave Suite	4089628421	kiablonski	k&jblon8524	2010	abc@gmail.com	SOCIAL MEET

## Login

SELECT NAME,USERNAME, PASSWORD  
FROM CUSTOMER  
WHERE USERNAME='KJABLONSKI' AND PASSWORD= 'K&JBLON8524'

NAME	USERNAME	PASSWORD
Karl Jablonski	kiablonski	k&jblon8524

## 2. Customer can subscribe from various membership plans available

SELECT \* FROM MEMBERSHIP\_PLAN;

INSERT INTO MEMBERSHIP (MEMBERSHIP\_ID, CUSTOMER\_ID, PLAN\_ID,  
MEMBERSHIP\_START\_DATE, MEMBERSHIP\_EXPIRY\_DATE)  
VALUES ('2010','100','3','2017-01-01','2017-12-31')

Before new customer opts for membership:

MEMBERSHIP_ID	CUSTOMER_ID	PLAN_ID	MEMBERSHIP_START_DATE	MEMBERSHIP_EXPIRY_DATE	TRAINING_START_DATE	TRAINING_END_DATE
1079	61	3	2018-06-03	2019-06-02	NULL	NULL
1080	64	2	2018-07-20	2019-01-19	2018-08-01	2019-01-01
1081	67	1	2018-04-09	2018-07-08	NULL	NULL
1082	69	3	2017-07-01	2018-06-30	NULL	NULL
1083	71	3	2017-08-21	2018-08-20	NULL	NULL
1084	73	1	2018-06-23	2018-09-22	NULL	NULL
1085	75	2	2018-07-20	2019-01-19	2018-07-20	2019-01-19
1086	11	2	2017-10-12	2018-03-11	NULL	NULL
1087	13	1	2017-04-09	2017-07-08	NULL	NULL
1088	23	3	2017-06-03	2018-06-02	NULL	NULL
1089	37	1	2017-03-04	2017-06-03	NULL	NULL
1090	52	2	2017-02-15	2017-08-14	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

After new customer has opted for membership

MEMBERSHIP_ID	CUSTOMER_ID	PLAN_ID	MEMBERSHIP_START_DATE	MEMBERSHIP_EXPIRY_DATE	TRAINING_START_DATE	TRAINING_END_DATE
1080	64	2	2018-07-20	2019-01-19	2018-08-01	2019-01-01
1081	67	1	2018-04-09	2018-07-08	NULL	NULL
1082	69	3	2017-07-01	2018-06-30	NULL	NULL
1083	71	3	2017-08-21	2018-08-20	NULL	NULL
1084	73	1	2018-06-23	2018-09-22	NULL	NULL
1085	75	2	2018-07-20	2019-01-19	2018-07-20	2019-01-19
1086	11	2	2017-10-12	2018-03-11	NULL	NULL
1087	13	1	2017-04-09	2017-07-08	NULL	NULL
1088	23	3	2017-06-03	2018-06-02	NULL	NULL
1089	37	1	2017-03-04	2017-06-03	NULL	NULL
1090	52	2	2017-02-15	2017-08-14	NULL	NULL
2010	100	3	2017-01-01	2017-12-31	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

## 3. Customer can join various classes based on availability.

SELECT T2.NAME,T2.DESCRPTION, T1.SESSION\_ID, T1.START\_TIME\_OF\_DAY,  
T1.END\_TIME\_OF\_DAY, T1.AVAILABILITY FROM (SELECT R2.SESSION\_ID,R2.CLASS\_ID,  
R2.START\_TIME\_OF\_DAY, R2.END\_TIME\_OF\_DAY, (R2.SESSION\_STRENGTH)-  
(R1.TOTAL\_COUNT) AS AVAILABILITY FROM  
(SELECT MS.SESSION\_ID, COUNT(MS.ENROLLMENT\_ID) AS TOTAL\_COUNT FROM  
ENROLLMENT\_STACK MS  
WHERE MS.ENROLLMENT\_EXPIRY\_DATE>=SYSDATE() GROUP BY MS.SESSION\_ID) R1,  
SESSION R2  
WHERE R1.SESSION\_ID=R2.SESSION\_ID AND R1.TOTAL\_COUNT<R2.SESSION\_STRENGTH)  
T1, CLASS T2 WHERE T1.CLASS\_ID=T2.CLASS\_ID;



	NAME	DESCRIPTION	SESSION_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	AVAILABILITY
	Yoga	Yoga blends balance, strength, flexibility and o...	1	07:00:00.00	08:00:00.00	1
	Yoga	Yoga blends balance, strength, flexibility and o...	2	09:00:00.00	10:00:00.00	3
	Zumba	Zumba features aerobic/fitness interval traino...	3	18:00:00.00	19:00:00.00	6
	Aerobics	Aerobics is a form of physical exercise that com...	4	08:00:00.00	09:00:00.00	2
	Aerobics	Aerobics is a form of physical exercise that com...	5	19:00:00.00	20:00:00.00	6

#### 4. Customer can select personal trainer and choose time slots according to trainer's availability.

To view which trainer is available:

```
SELECT E.EMP_NAME, E.CONTACT_NO, E.EMAIL, T.EXPERIENCE, T.CERTIFICATION,
TS.START_TIME_OF_DAY, TS.END_TIME_OF_DAY
FROM EMPLOYEE E, TRAINER T, TRAINER_SESSION TS
WHERE TS.TRAINER_ID=T.TRAINER_ID AND T.TRAINER_ID=E.EMPLOYEE_ID
AND MEMBERSHIP_ID IS NULL;
```

	EMP_NAME	CONTACT_NO	EMAIL	EXPERIENCE	CERTIFICATION	START_TIME_OF_DAY	END_TIME_OF_DAY
	Samantha	(049)375-5630x9472	tvrique51@example.net	3 YEARS	NASM.ACE	07:30:00.00	08:30:00.00
	Samantha	(049)375-5630x9472	tvrique51@example.net	3 YEARS	NASM.ACE	10:30:00.00	11:30:00.00
	Samantha	(049)375-5630x9472	tvrique51@example.net	3 YEARS	NASM.ACE	15:00:00.00	16:00:00.00
	Samantha	(049)375-5630x9472	tvrique51@example.net	3 YEARS	NASM.ACE	19:30:00.00	20:30:00.00
	Danielle	(883)945-1017x47399	dkulas@example.ora	2 YEARS	NSCA.ACE	07:30:00.00	08:30:00.00
	Danielle	(883)945-1017x47399	dkulas@example.ora	2 YEARS	NSCA.ACE	09:00:00.00	10:00:00.00
	Danielle	(883)945-1017x47399	dkulas@example.ora	2 YEARS	NSCA.ACE	15:00:00.00	16:00:00.00
	Danielle	(883)945-1017x47399	dkulas@example.ora	2 YEARS	NSCA.ACE	18:00:00.00	19:00:00.00
	Kevshawn	395.501.6504	iosefina18@example.ora	4 YEARS	ACSM	07:30:00.00	08:30:00.00
	Kevshawn	395.501.6504	iosefina18@example.ora	4 YEARS	ACSM	09:00:00.00	10:00:00.00
	Kevshawn	395.501.6504	iosefina18@example.ora	4 YEARS	ACSM	15:00:00.00	16:00:00.00
	Kevshawn	395.501.6504	iosefina18@example.ora	4 YEARS	ACSM	18:00:00.00	19:00:00.00
	Gavin	017.628.5541	wolf.ieramv@example.ora	0 YEARS	NASM	06:00:00.00	07:00:00.00
	Gavin	017.628.5541	wolf.ieramv@example.ora	0 YEARS	NASM	09:00:00.00	10:00:00.00
	Gavin	017.628.5541	wolf.ieramv@example.ora	0 YEARS	NASM	15:00:00.00	16:00:00.00
	Gavin	017.628.5541	wolf.ieramv@example.ora	0 YEARS	NASM	18:00:00.00	19:00:00.00
	Kathlrvn	052.301.2053x734	mavis34@example.net	1 YEAR	NFPT.ACE	06:00:00.00	07:00:00.00
	Kathlrvn	052.301.2053x734	mavis34@example.net	1 YEAR	NFPT.ACE	09:00:00.00	10:00:00.00
	Kathlrvn	052.301.2053x734	mavis34@example.net	1 YEAR	NFPT.ACE	15:00:00.00	16:00:00.00
	Kathlrvn	052.301.2053x734	mavis34@example.net	1 YEAR	NFPT.ACE	18:00:00.00	19:00:00.00

Before select a trainer:

	TRAINER_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	MEMBERSHIP_ID
	5	06:00:00.00	07:00:00.00	1002
	5	07:30:00.00	08:30:00.00	NULL
	5	09:00:00.00	10:00:00.00	1039
	5	10:30:00.00	11:30:00.00	NULL
	5	15:00:00.00	16:00:00.00	NULL
	5	16:30:00.00	17:30:00.00	1005
	5	18:00:00.00	19:00:00.00	1080
	5	19:30:00.00	20:30:00.00	NULL
	7	06:00:00.00	07:00:00.00	1032

After selecting a trainer:

```
UPDATE TRAINER_SESSION SET MEMBERSHIP_ID='1008'
WHERE TRAINER_ID='5' AND START_TIME_OF_DAY='07:30:00'
AND END_TIME_OF_DAY='08:30:00';
SELECT * FROM TRAINER_SESSION;
```



	TRAINER_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	MEMBERSHIP_ID
	5	06:00:00.00	07:00:00.00	1002
	5	07:30:00.00	08:30:00.00	1008
	5	09:00:00.00	10:00:00.00	1039
	5	10:30:00.00	11:30:00.00	NULL
	5	15:00:00.00	16:00:00.00	NULL
	5	16:30:00.00	17:30:00.00	1005
	5	18:00:00.00	19:00:00.00	1080
	5	19:30:00.00	20:30:00.00	NULL

## 5. Customer can view his activity records and health reports.

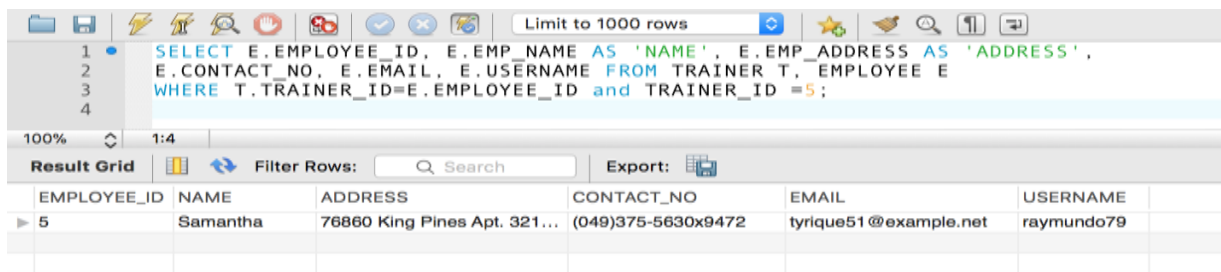
SELECT \* FROM ACTIVITY\_DETAILS WHERE CUSTOMER\_ID=25;

	CUSTOMER_ID	ACTIVITY_DATE	CALORIES	STEPS	ACTIVE_MINUTES
	25	2018-05-20 10:30:00	467	974	56
	25	2018-05-21 10:30:00	453	958	51
	25	2018-05-22 10:30:00	486	927	56
	25	2018-05-23 10:30:00	493	948	52
	25	2018-05-24 10:30:00	457	957	58
	25	2018-05-25 10:30:00	428	893	48
	25	2018-05-26 10:30:00	439	874	47
	25	2018-05-27 10:30:00	471	932	48
	25	2018-05-28 10:30:00	548	950	43
	25	2018-05-29 10:30:00	542	950	47

## TRAINER

### 1. Trainer can view his profile (personal details).

SELECT E.EMPLOYEE\_ID, E.EMP\_NAME AS 'NAME', E.EMP\_ADDRESS AS 'ADDRESS',  
E.CONTACT\_NO, E.EMAIL, E.USERNAME FROM TRAINER T, EMPLOYEE E  
WHERE T.TRAINER\_ID=E.EMPLOYEE\_ID and TRAINER\_ID =5;



The screenshot shows a database management tool interface. At the top, there's a toolbar with various icons and a 'Limit to 1000 rows' dropdown. Below the toolbar, a SQL query is entered in a text area:

```
SELECT E.EMPLOYEE_ID, E.EMP_NAME AS 'NAME', E.EMP_ADDRESS AS 'ADDRESS',
E.CONTACT_NO, E.EMAIL, E.USERNAME FROM TRAINER T, EMPLOYEE E
WHERE T.TRAINER_ID=E.EMPLOYEE_ID and TRAINER_ID =5;
```

Below the query, there's a 'Result Grid' section. It includes a 'Filter Rows' search bar and an 'Export' button. The results are displayed in a table with the following columns: EMPLOYEE\_ID, NAME, ADDRESS, CONTACT\_NO, EMAIL, and USERNAME. The first row shows data for employee ID 5, named Samantha.

EMPLOYEE_ID	NAME	ADDRESS	CONTACT_NO	EMAIL	USERNAME
5	Samantha	76860 King Pines Apt. 321...	(049)375-5630x9472	tyrique51@example.net	raymundo79

### 2. Trainer can update his personal details if required.

Before updating his address:

SELECT E.EMPLOYEE\_ID, E.EMP\_NAME AS 'NAME', E.EMP\_ADDRESS AS 'ADDRESS',  
E.CONTACT\_NO, E.EMAIL, E.USERNAME FROM TRAINER T, EMPLOYEE E  
WHERE T.TRAINER\_ID=E.EMPLOYEE\_ID and TRAINER\_ID =5;

EMPLOYEE_ID	NAME	ADDRESS	CONTACT_NO	EMAIL	USERNAME
▶ 5	Samantha	76860 King Pines Apt. 321...	(049)375-5630x9472	tyrique51@example.net	raymundo79

After Updating his address:

UPDATE EMPLOYEE E SET E.EMP\_ADDRESS = '6384 Clara Plaza Devonville, VT 84375-7494'  
WHERE E.EMPLOYEE\_ID=5;

SELECT E.EMPLOYEE\_ID, E.EMP\_NAME AS 'NAME', E.EMP\_ADDRESS AS 'ADDRESS',  
E.CONTACT\_NO, E.EMAIL, E.USERNAME FROM TRAINER T, EMPLOYEE E  
WHERE T.TRAINER\_ID=E.EMPLOYEE\_ID and TRAINER\_ID =5;

EMPLOYEE_ID	NAME	ADDRESS	CONTACT_NO	EMAIL	USERNAME
▶ 5	Samantha	6384 Clara Plaza Devonville, VT 84375-7494	(049)375-5630x9472	tyrique51@example.net	raymundo79

### 3. Trainer can add and view his availability in trainer session.

Before adding his availability:

SELECT \* FROM TRAINER\_SESSION WHERE TRAINER\_ID=11

	TRAINER_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	MEMBERSHIP_ID
▶	11	06:00:00.00	07:00:00.00	NULL
	11	07:30:00.00	08:30:00.00	1085
	11	09:00:00.00	10:00:00.00	NULL
	11	10:30:00.00	11:30:00.00	1044
	11	15:00:00.00	16:00:00.00	NULL
	11	16:30:00.00	17:30:00.00	1077
	11	18:00:00.00	19:00:00.00	NULL
	11	19:30:00.00	20:30:00.00	1047

After adding his availability:

INSERT INTO TRAINER\_SESSION (TRAINER\_ID, START\_TIME\_OF\_DAY,  
END\_TIME\_OF\_DAY) VALUES (11, '21:00:00', '22:00:00');

	TRAINER_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	MEMBERSHIP_ID
	11	06:00:00.00	07:00:00.00	NULL
	11	07:30:00.00	08:30:00.00	1085
	11	09:00:00.00	10:00:00.00	NULL
	11	10:30:00.00	11:30:00.00	1044
	11	15:00:00.00	16:00:00.00	NULL
	11	16:30:00.00	17:30:00.00	1077
	11	18:00:00.00	19:00:00.00	NULL
	11	19:30:00.00	20:30:00.00	1047
▶	11	21:00:00.00	22:00:00.00	NULL

#### 4. Trainer can view his member's schedule.

```
SELECT * FROM TRAINER_SESSION WHERE TRAINER_ID=9 AND
MEMBERSHIP_ID IS NOT NULL;
```

	TRAINER_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	MEMBERSHIP_ID
▶	9	06:00:00.00	07:00:00.00	1019
	9	10:30:00.00	11:30:00.00	1026
	9	16:30:00.00	17:30:00.00	1073
	9	19:30:00.00	20:30:00.00	1037

#### 5. Trainer can view his member's activity records on daily basis.

```
SELECT * FROM ACTIVITY_DETAILS WHERE CUSTOMER_ID=(SELECT M.CUSTOMER_ID
FROM TRAINER_SESSION TS, MEMBERSHIP M WHERE TS.MEMBERSHIP_ID=1026
AND TS.MEMBERSHIP_ID=M.MEMBERSHIP_ID);
```

	CUSTOMER_ID	ACTIVITY_DATE	CALORIES	STEPS	ACTIVE_MINUTES
▶	25	2018-05-20 10:30:00	467	974	56
	25	2018-05-21 10:30:00	453	958	51
	25	2018-05-22 10:30:00	486	927	56
	25	2018-05-23 10:30:00	493	948	52
	25	2018-05-24 10:30:00	457	957	58
	25	2018-05-25 10:30:00	428	893	48
	25	2018-05-26 10:30:00	439	874	47
	25	2018-05-27 10:30:00	471	932	48
	25	2018-05-28 10:30:00	548	950	43
	25	2018-05-29 10:30:00	542	950	47
	25	2018-05-30 10:30:00	345	840	50
	25	2018-05-31 10:30:00	458	867	45
	25	2018-06-01 10:30:00	467	847	44
	25	2018-06-02 10:30:00	483	839	46
	25	2018-06-03 10:30:00	492	957	52

#### 6. Instructor should be able to view the sessions he is assigned, its timings and number of customers enrolled for that session.

```
SELECT S.INSTRUCTOR_ID, C.NAME AS CLASS_NAME, S.SESSION_ID,
S.START_TIME_OF_DAY, S.END_TIME_OF_DAY, R1.TOTAL_CUSTOMERS_ENROLLED
FROM SESSION S, CLASS C,(SELECT SESSION_ID, COUNT(*) AS
TOTAL_CUSTOMERS_ENROLLED FROM ENROLLMENT_STACK ES
WHERE ES.SESSION_ID IN (SELECT SESSION_ID FROM SESSION
WHERE INSTRUCTOR_ID=14) AND ES.ENROLLMENT_EXPIRY_DATE>=SYSDATE()
GROUP BY SESSION_ID) R1 WHERE R1.SESSION_ID=S.SESSION_ID AND
S.CLASS_ID=C.CLASS_ID AND INSTRUCTOR_ID=14;
```

	INSTRUCTOR_ID	CLASS_NAME	SESSION_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	TOTAL_CUSTOMERS_ENROLLED
▶	14	Yoga	1	07:00:00.00	08:00:00.00	9
	14	Yoga	2	09:00:00.00	10:00:00.00	7

## MANAGER

1. Manager can review customer's feedback and track the performance of trainers, for forecasting addition or layoff of trainers to the fitness centre.

### #LAYOFF

```
DELETE FROM EMPLOYEE WHERE EMP_NAME IN (SELECT R.SUB_CATEGORY
FROM (SELECT CATEGORY, SUB_CATEGORY,AVG(RATING) AS 'AVERAGE_RATING'
FROM CUSTOMER_FEEDBACK WHERE CATEGORY = 'TRAINER' AND
YEAR(DATE)='2018' GROUP BY SUB_CATEGORY
HAVING AVERAGE_RATING<3.0)R);
```

### Average customer feedback

	year	CATEGORY	trainer name	AVERAGE_RATING
	2018	TRAINER	DANIELLE	4.1429
	2018	TRAINER	GAVIN	4.7273
	2018	TRAINER	KATHLYN	4.2727
	2018	TRAINER	KAYSHAWN	3.7778
	2018	TRAINER	SAMANTHA	2.7143

After removing employee with bad rating:

217	01:28:11	DELETE FROM EMPLOYEE WHERE EMP_NAME IN (SELECT R.SUB_Catego...	1 row(s) affected	0.141 sec
-----	----------	--	-------------------	-----------

2. Manager can generate reports for performance review and declare awards or bonus.

```
UPDATE EMPLOYEE SET SALARY = 1.1*SALARY WHERE EMP_NAME IN (SELECT
YEAR(DATE) AS 'YEAR', CATEGORY, SUB_CATEGORY AS 'TRAINER NAME',
AVG(RATING) AS 'AVERAGE_RATING' FROM CUSTOMER_FEEDBACK
WHERE CATEGORY = 'TRAINER' GROUP BY SUB_CATEGORY, YEAR(DATE)
HAVING AVERAGE_RATING>=4);
```

### Salary before update:

	EMP_NAME	SALARY
	Samantha	5000
	Danielle	5000
	Kevshawn	5000
	Gavin	5000
	Kathlyn	5000

### Salary after increment for trainers with good rating:

	EMPLOYEE_ID	EMP_NAME	SALARY
	7	Danielle	5500
	9	Kevshawn	5000
	11	Gavin	5500
	13	Kathlyn	5500
	NULL	NULL	NULL

### 3. Manager can manage trainer and staff leaves.

```
UPDATE EMPLOYEE_LEAVE SET APPROVAL_STATUS = 'YES'
WHERE EMPLOYEE_ID = '12' AND CATEGORY='SICK' AND
DATE IN( SELECT DATE FROM EMPLOYEE_LEAVE WHERE YEAR(DATE)=2018);
```

Before manager approves leave:

	EMPLOYEE_ID	DATE	CATEGORY	REASON	APPROVAL_STATUS
	6	2018-05-31	SICK	FEVER	YES
	7	2017-09-15	SICK	FEVER	YES
	7	2018-04-14	SICK	FEVER	YES
	11	11 7-10-30	CASUAL	VACATION	YES
	11	2018-03-23	CASUAL	VACATION	NO
	12	2017-05-23	CASUAL	VACATION	NO
	12	2018-06-03	SICK	FEVER	NULL
	17	2017-11-27	SICK	FEVER	YES
	17	2018-05-15	SICK	FEVER	YES
	19	2017-12-15	CASUAL	VACATION	NO
	19	2018-04-28	CASUAL	VACATION	NO
	NULL	NULL	NULL	NULL	NULL

After manager approves leave:

	EMPLOYEE_ID	DATE	CATEGORY	REASON	APPROVAL_STATUS
	1	2018-06-12	CASUAL	VACATION	YES
	2	2017-07-21	CASUAL	VACATION	YES
	2	2017-08-21	CASUAL	VACATION	NO
	3	2017-08-12	SICK	FEVER	YES
	4	2017-06-04	SICK	FEVER	YES
	5	2017-06-18	CASUAL	VACATION	YES
	5	2017-12-31	SICK	FEVER	YES
	6	2018-07-24	SICK	FEVER	YES

### 4. Manager can organize schedules for trainer and staff.

```
INSERT INTO TRAINER_SESSION (TRAINER_ID, START_TIME_OF_THE_DAY,
END_TIME_OF_DAY) VALUES ('5','21:00:00','22:00:00');
```

```
UPDATE TRAINER_SESSION SET START_TIME_OF_THE_DAY = '21:30:00',
END_TIME = '22:30:00'
WHERE TRAINER_ID = 5 AND START_TIME_OF_DAY = '21:00:00';
```

```
UPDATE STAFF SET SHIFT_TIMING = '10AM-5PM' WHERE EMPLOYEE_ID =2;
Trainer time before update:
```

	TRAINER_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	MEMBERSHIP_ID
	5	15:00:00.00	16:00:00.00	NULL
	5	16:30:00.00	17:30:00.00	1005
	5	18:00:00.00	19:00:00.00	1080
	5	19:30:00.00	20:30:00.00	NULL
	5	21:00:00.00	22:00:00.00	NULL
	7	06:00:00.00	07:00:00.00	1032
	7	07:30:00.00	08:30:00.00	NULL
	7	09:00:00.00	10:00:00.00	NULL

Trainer time after update:

	TRAINER_ID	START_TIME_OF_DAY	END_TIME_OF_DAY	MEMBERSHIP_ID
	5	16:30:00.00	17:30:00.00	1005
	5	18:00:00.00	19:00:00.00	1080
	5	19:30:00.00	20:30:00.00	NULL
	5	21:30:00.00	22:30:00.00	NULL
	7	06:00:00.00	07:00:00.00	1032
	7	07:30:00.00	08:30:00.00	NULL
	7	09:00:00.00	10:00:00.00	NULL
	7	10:30:00.00	11:30:00.00	1009

Staff time before update:

	STAFF_ID	SHIFT_TIMING	JOB_TITLE
	8	3PM-12AM	RECEPTIONIST
	10	5AM-12PM	HOUSE KEEPING
	12	5AM-12PM	HOUSE KEEPING
	15	12PM-5PM	HOUSE KEEPING
	17	12PM-5PM	HOUSE KEEPING
	19	5PM-12AM	HOUSE KEEPING
	20	5PM-12AM	HOUSE KEEPING
	NULL	NULL	NULL

Staff time after update:

	STAFF_ID	SHIFT_TIMING	JOB_TITLE
	8	3PM-12AM	RECEPTIONIST
	10	5AM-12PM	HOUSE KEEPING
	12	5AM-12PM	HOUSE KEEPING
	15	12PM-5PM	HOUSE KEEPING
	17	12PM-5PM	HOUSE KEEPING
	19	5PM-12AM	HOUSE KEEPING
	19	10AM-5PM	HOUSE KEEPING
	NULL	NULL	NULL

## ACCOUNTANT

**1. Accountant will be responsible for reporting weekly/monthly finances and business performance reports over time to the manager.**

SELECT \* FROM CUSTOMER\_PAYMENT;

#YEARLY PERFORMANCE REPORT

SELECT YEAR(PAYMENT\_DATE) AS 'YEAR', PAYMENT\_DESC, 'USD' AS 'CURRENCY',

SUM(AMOUNT) FROM CUSTOMER\_PAYMENT

GROUP BY YEAR(PAYMENT\_DATE), PAYMENT\_DESC;

Yearly report:

	YEAR	PAYMENT_DESC	CURRENCY	SUM(AMOUNT)
	2017	MEMBERSHIP FEE	USD	3200
	2017	TRAINER FEE	USD	200
	2018	EQUIPMENT FEE	USD	380
	2018	LOCKER FEE	USD	780
	2018	MEMBERSHIP FEE	USD	12650
	2018	TRAINER FEE	USD	1400

#Monthly report:

```
SELECT YEAR(PAYMENT_DATE) AS 'YEAR',MONTH(PAYMENT_DATE) AS  
'MONTH',PAYMENT_DESC,'USD' AS 'CURRENCY',SUM(AMOUNT)  
FROM CUSTOMER_PAYMENT  
GROUP BY YEAR(PAYMENT_DATE),MONTH(PAYMENT_DATE),  
PAYMENT_DESC;
```

Monthly report:

	YEAR	MONTH	PAYMENT_DESC	CURRENCY	SUM(AMOUNT)
	2017	7	MEMBERSHIP FEE	USD	1600
	2017	8	MEMBERSHIP FEE	USD	800
	2017	11	MEMBERSHIP FEE	USD	800
	2017	12	TRAINER FEE	USD	200
	2018	1	MEMBERSHIP FEE	USD	2400
	2018	1	TRAINER FEE	USD	400
	2018	2	MEMBERSHIP FEE	USD	1300
	2018	3	MEMBERSHIP FEE	USD	2600
	2018	3	TRAINER FEE	USD	400
	2018	4	LOCKER FEE	USD	200
	2018	4	MEMBERSHIP FEE	USD	1800
	2018	4	TRAINER FEE	USD	200
	2018	5	LOCKER FEE	USD	70
	2018	5	MEMBERSHIP FEE	USD	1500
	2018	6	EQUIPMENT FEE	USD	100
	2018	6	LOCKER FEE	USD	260
	2018	6	MEMBERSHIP FEE	USD	2550
	2018	6	TRAINER FEE	USD	200
	2018	7	EQUIPMENT FEE	USD	180
	2018	7	LOCKER FEE	USD	90
	2018	7	MEMBERSHIP FEE	USD	500
	2018	7	TRAINER FEE	USD	200
	2018	8	LOCKER FEE	USD	30
	2018	9	EQUIPMENT FEE	USD	80
	2018	9	LOCKER FEE	USD	50
	2018	10	EQUIPMENT FEE	USD	20
	2018	10	LOCKER FEE	USD	30
	2018	11	LOCKER FEE	USD	50



## SQL - BUSINESS METRICS QUERIES

### 1. Members Vs Non-members:

```
SELECT COUNT(*) FROM CUSTOMER;
```

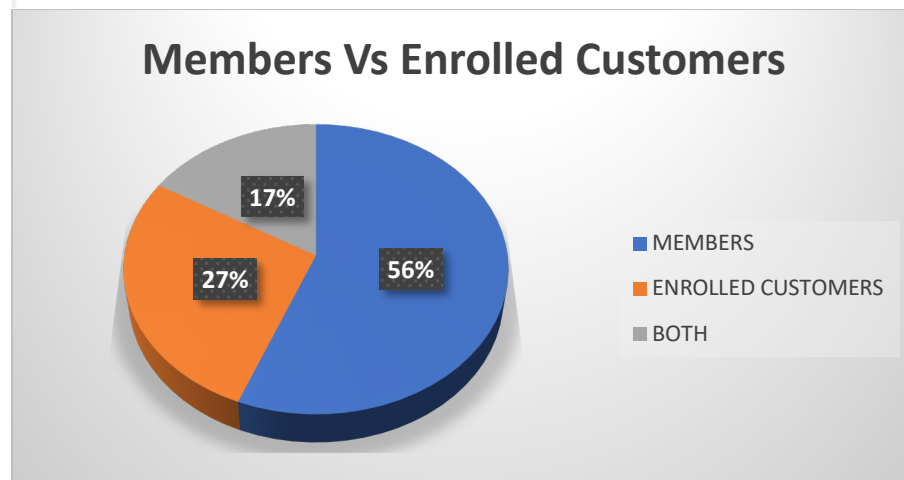
	COUNT(*)
	76

```
SELECT COUNT(C.CUSTOMER_ID) AS 'NON-MEMBERS'  
FROM CUSTOMER C WHERE C.CUSTOMER_ID  
NOT IN (SELECT DISTINCT M.CUSTOMER_ID FROM MEMBERSHIP M);
```

	NON-MEMBERS
	25

```
SELECT COUNT(DISTINCT M.CUSTOMER_ID) AS 'BOTH'  
FROM ENROLLMENT_STACK ES, MEMBERSHIP M  
WHERE ES.CUSTOMER_ID=M.CUSTOMER_ID;
```

	BOTH
	15

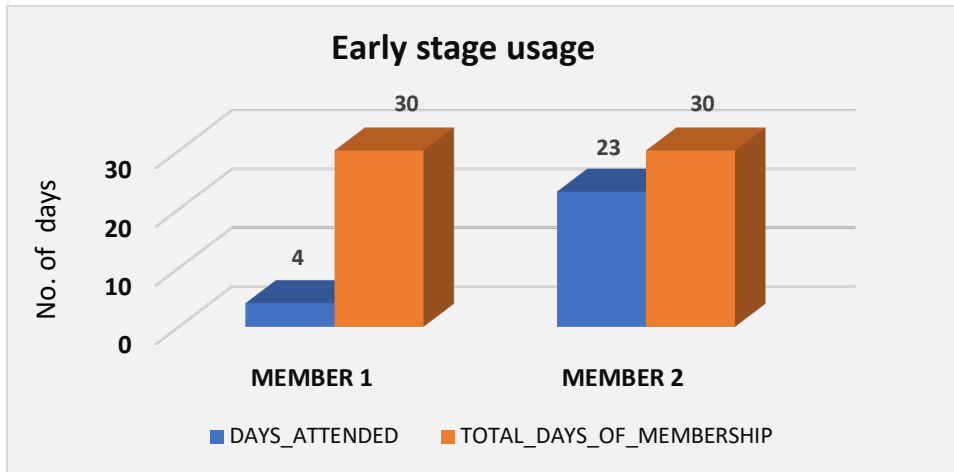


### 2. Early Stage Usage:

```
SELECT MA.MEMBERSHIP_ID, R1.MEMBERSHIP_START_DATE,  
R1.MEMBERSHIP_EXPIRY_DATE,  
30 AS 'TOTAL NO. OF DAYS', COUNT(ENTRY_TIME) AS 'DAYS_ATTENDED'  
FROM (SELECT * FROM MEMBERSHIP WHERE CUSTOMER_ID IN (SELECT CUSTOMER_ID  
FROM CUSTOMER WHERE CURRENT_MEM_ID IS NOT NULL)) R1, MEMBER_ATTENDANCE  
MA WHERE R1.MEMBERSHIP_ID=MA.MEMBERSHIP_ID AND  
MA.ENTRY_TIME>=R1.MEMBERSHIP_START_DATE AND  
EXIT_TIME<=date_add(MEMBERSHIP_START_DATE, interval 30 day)  
GROUP BY R1.MEMBERSHIP_ID ORDER BY MA.MEMBERSHIP_ID;
```



	MEMBERSHIP_ID	MEMBERSHIP_START_DATE	MEMBERSHIP_EXPIRY_DATE	TOTAL NO. OF DAYS	DAYS_ATTENDED
	1021	2018-03-10	2018-06-30	30	4
	1049	2018-03-10	2018-06-09	30	23



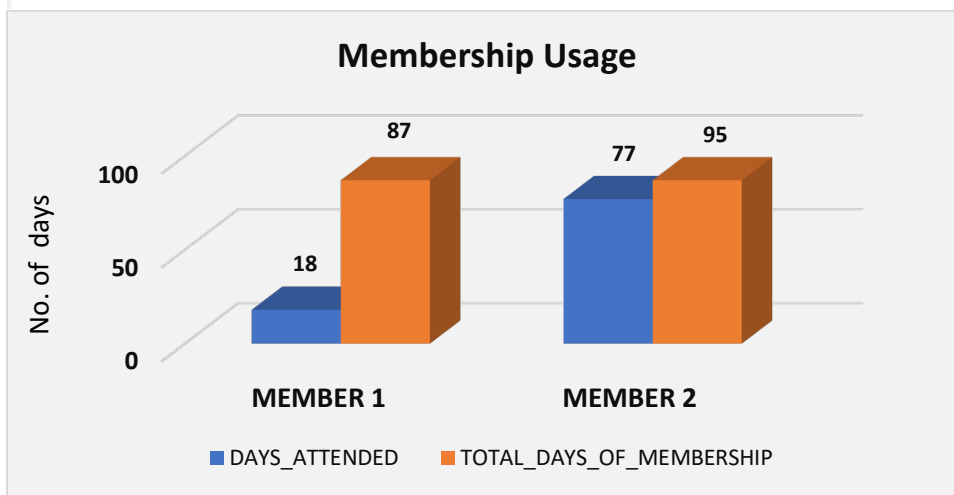
### 3. Membership Usage:

```

SELECT R.MEMBERSHIP_ID, COUNT(MA.ENTRY_TIME),
(datediff(sysdate(),R.MEMBERSHIP_START_DATE))
FROM (SELECT * FROM MEMBERSHIP WHERE CUSTOMER_ID IN
(SELECT CUSTOMER_ID FROM CUSTOMER WHERE CURRENT_MEM_ID IS NOT NULL))R,
MEMBER_ATTENDANCE MA WHERE R.MEMBERSHIP_ID=MA.MEMBERSHIP_ID
GROUP BY MEMBERSHIP_ID;

```

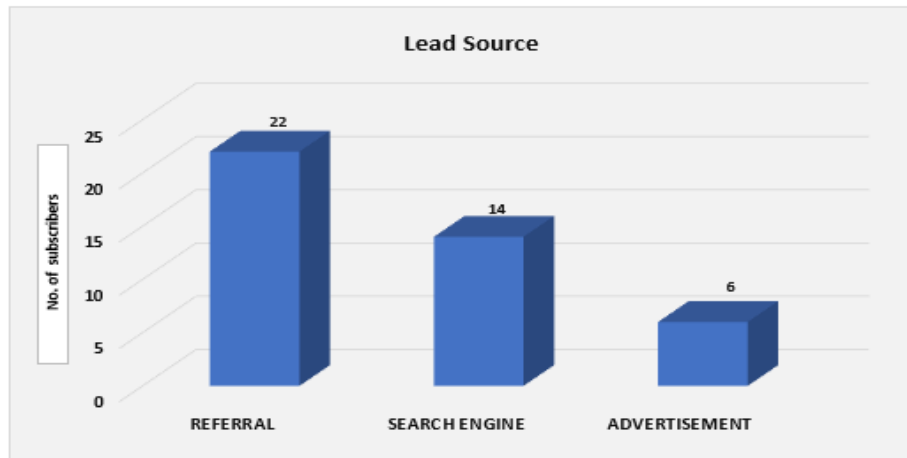
	MEMBERSHIP_ID	COUNT(MA.ENTRY_TIME)	(datediff(sysdate(),R.MEMBERSHIP_START_DATE))
	1021	18	88
	1049	77	88



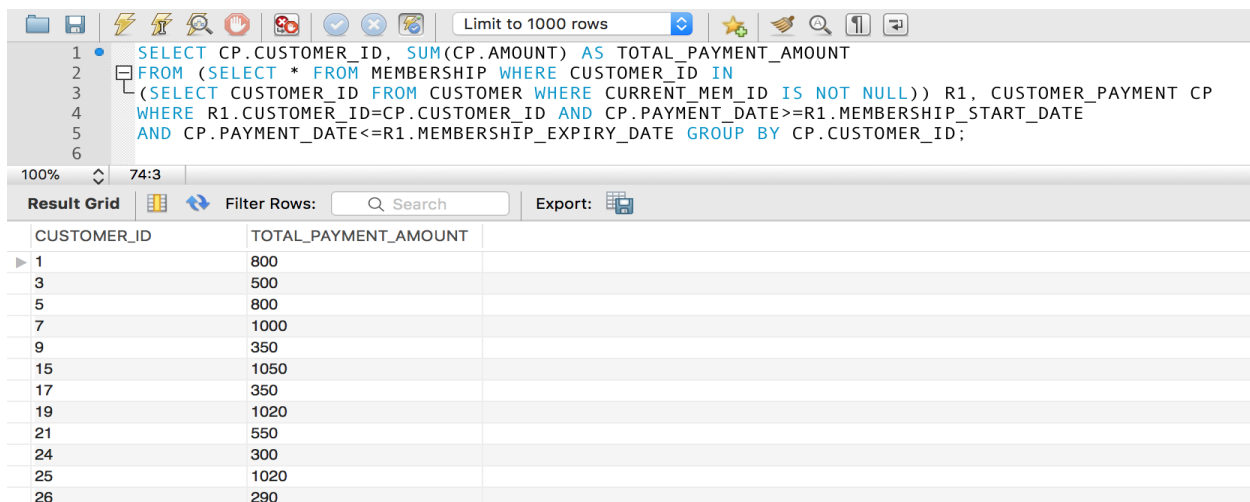
#### 4. Lead Source:

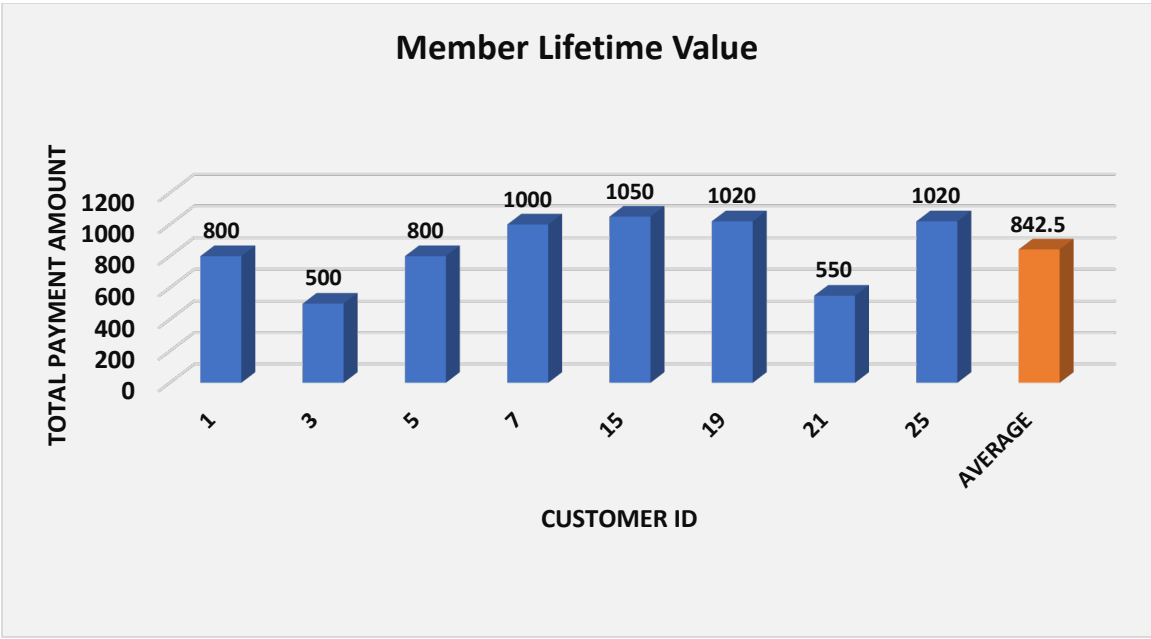
SELECT HEARD\_ABOUT\_US, COUNT(\*) AS TOTAL\_COUNT FROM CUSTOMER WHERE HEARD\_ABOUT\_US IS NOT NULL GROUP BY HEARD\_ABOUT\_US ORDER BY TOTAL\_COUNT DESC;

HEARD_ABOUT_US	TOTAL_COUNT
REFERRAL	22
SEARCH ENGINE	14
ADVERTISEMENT	6



#### 5. Member Lifetime Value

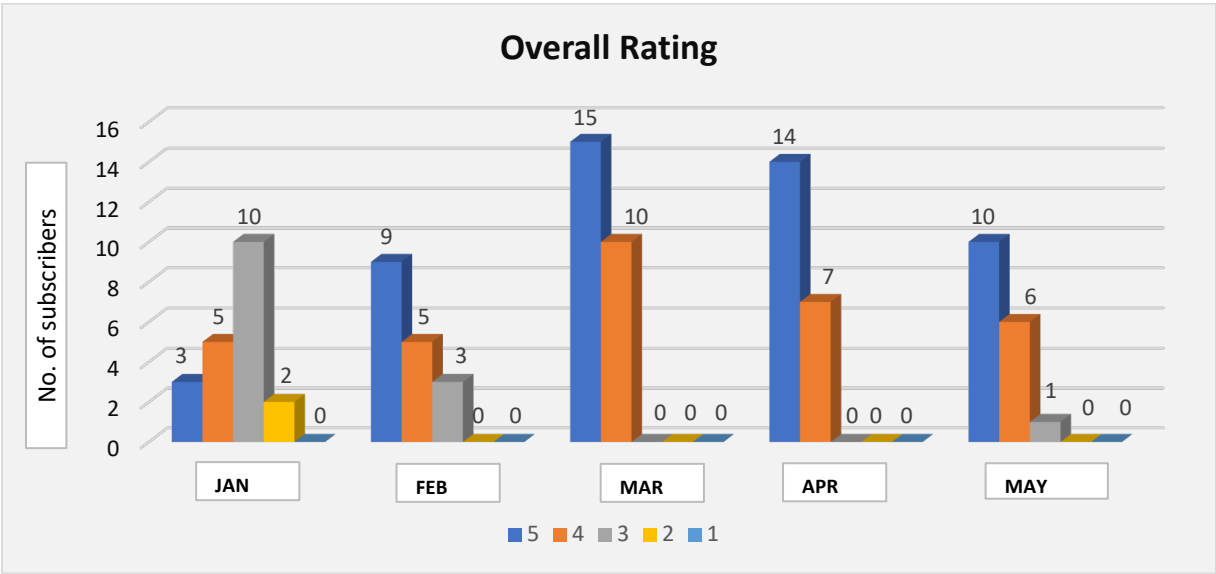
																											
<pre>1 SELECT CP.CUSTOMER_ID, SUM(CP.AMOUNT) AS TOTAL_PAYMENT_AMOUNT 2 FROM (SELECT * FROM MEMBERSHIP WHERE CUSTOMER_ID IN 3 (SELECT CUSTOMER_ID FROM CUSTOMER WHERE CURRENT_MEM_ID IS NOT NULL)) R1, CUSTOMER_PAYMENT CP 4 WHERE R1.CUSTOMER_ID=CP.CUSTOMER_ID AND CP.PAYMENT_DATE&gt;=R1.MEMBERSHIP_START_DATE 5 AND CP.PAYMENT_DATE&lt;=R1.MEMBERSHIP_EXPIRY_DATE GROUP BY CP.CUSTOMER_ID;</pre>																											
<table><tr><th>CUSTOMER_ID</th><th>TOTAL_PAYMENT_AMOUNT</th></tr><tr><td>1</td><td>800</td></tr><tr><td>3</td><td>500</td></tr><tr><td>5</td><td>800</td></tr><tr><td>7</td><td>1000</td></tr><tr><td>9</td><td>350</td></tr><tr><td>15</td><td>1050</td></tr><tr><td>17</td><td>350</td></tr><tr><td>19</td><td>1020</td></tr><tr><td>21</td><td>550</td></tr><tr><td>24</td><td>300</td></tr><tr><td>25</td><td>1020</td></tr><tr><td>26</td><td>290</td></tr></table>		CUSTOMER_ID	TOTAL_PAYMENT_AMOUNT	1	800	3	500	5	800	7	1000	9	350	15	1050	17	350	19	1020	21	550	24	300	25	1020	26	290
CUSTOMER_ID	TOTAL_PAYMENT_AMOUNT																										
1	800																										
3	500																										
5	800																										
7	1000																										
9	350																										
15	1050																										
17	350																										
19	1020																										
21	550																										
24	300																										
25	1020																										
26	290																										



**6. Customer Feedback:**

**a) Overall rating:**

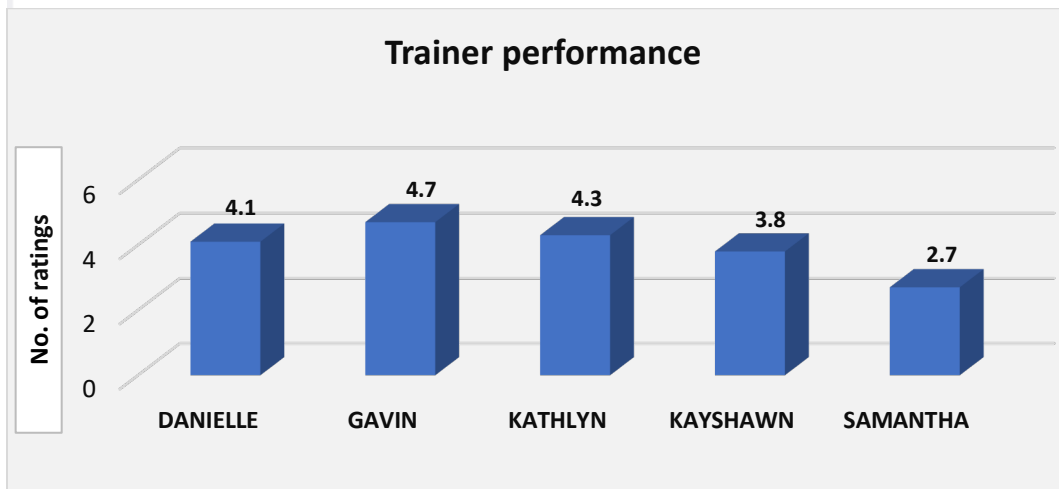
	MONTH	RATING	COUNT(*)
	April	4	7
	April	5	14
	Februarv	3	3
	Februarv	4	5
	Februarv	5	9
	Januarv	2	2
	Januarv	3	10
	Januarv	4	5
	Januarv	5	3
	March	4	10



**b) Trainer Performance:**

```
SELECT SUB_CATEGORY, AVG(RATING) AS AVG_RATING
FROM CUSTOMER_FEEDBACK
WHERE CATEGORY='TRAINER' AND EXTRACT(YEAR FROM DATE) = 2018
GROUP BY SUB_CATEGORY ORDER BY SUB_CATEGORY;
```

	SUB_CATEGORY	AVG_RATING
	DANIELLE	4.1429
	GAVIN	4.7273
	KATHLYN	4.2727
	KAYSHAWN	3.7778
	SAMANTHA	2.7143

**7. Enrollment/class:**

Each class and their corresponding sessions with the strength of each session:

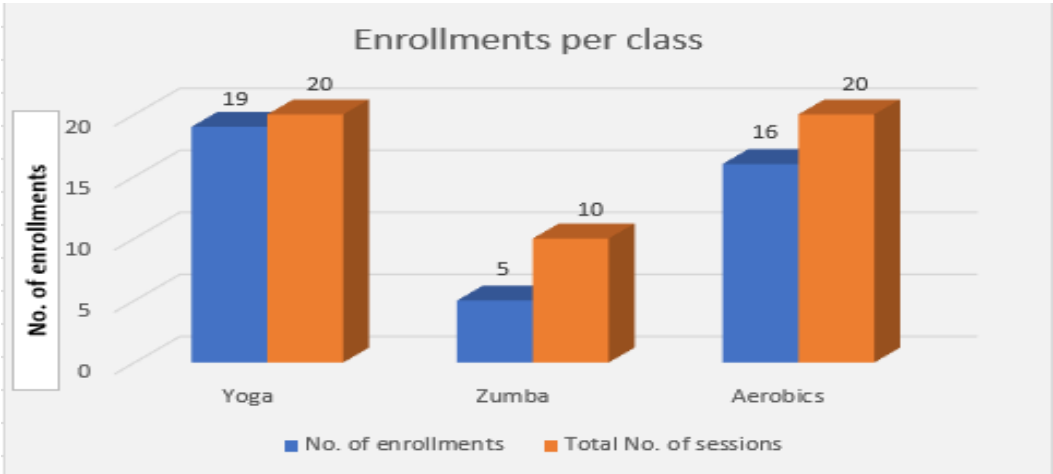
```
SELECT C.NAME, COUNT(DISTINCT S.SESSION_ID) AS 'NO. OF SESSIONS',
S.SESSION_STRENGTH
FROM ENROLLMENT_STACK ES, SESSION S, CLASS C
WHERE ES.SESSION_ID=S.SESSION_ID AND S.CLASS_ID=C.CLASS_ID
GROUP BY C.CLASS_ID, S.SESSION_STRENGTH;
```

	NAME	NO. OF SESSIONS	SESSION_STRENGTH
	Yoga	2	10
	Zumba	1	10
	Aerobics	2	10

No. of sessions enrolled out of total sessions available:

```
SELECT C.CLASS_ID, COUNT(ES.SESSION_ID) AS 'NO. OF SESSIONS ENROLLED',
COUNT(DISTINCT S.SESSION_ID)*SESSION_STRENGTH AS 'TOTAL SESSIONS'
FROM ENROLLMENT_STACK ES, SESSION S, CLASS C
WHERE ES.SESSION_ID=S.SESSION_ID AND S.CLASS_ID=C.CLASS_ID
GROUP BY C.CLASS_ID;
```

	NAME	NO. OF SESSIONS ENROLLED	TOTAL SESSIONS
	Yoga	19	20
	Zumba	5	10
	Aerobics	16	20



**THANK YOU**