Instruction

Name Surname: Yusuf Güngör

Student ID: 20170703099

Group with: Alone

Job

We took an assignment and, in this assignment, being wanted to build a database to manage zoos.

In this assignment, customers go to buy ticket and with this ticket they can enter zoos. Every costumer has a unique id, and every customers name, surname, email, address, credit card no must be stored.

Every ticket has a valid date, price and every ticket bought by customer has a ticket number. One customer can buy more than one ticket and tickets are manages by employees.

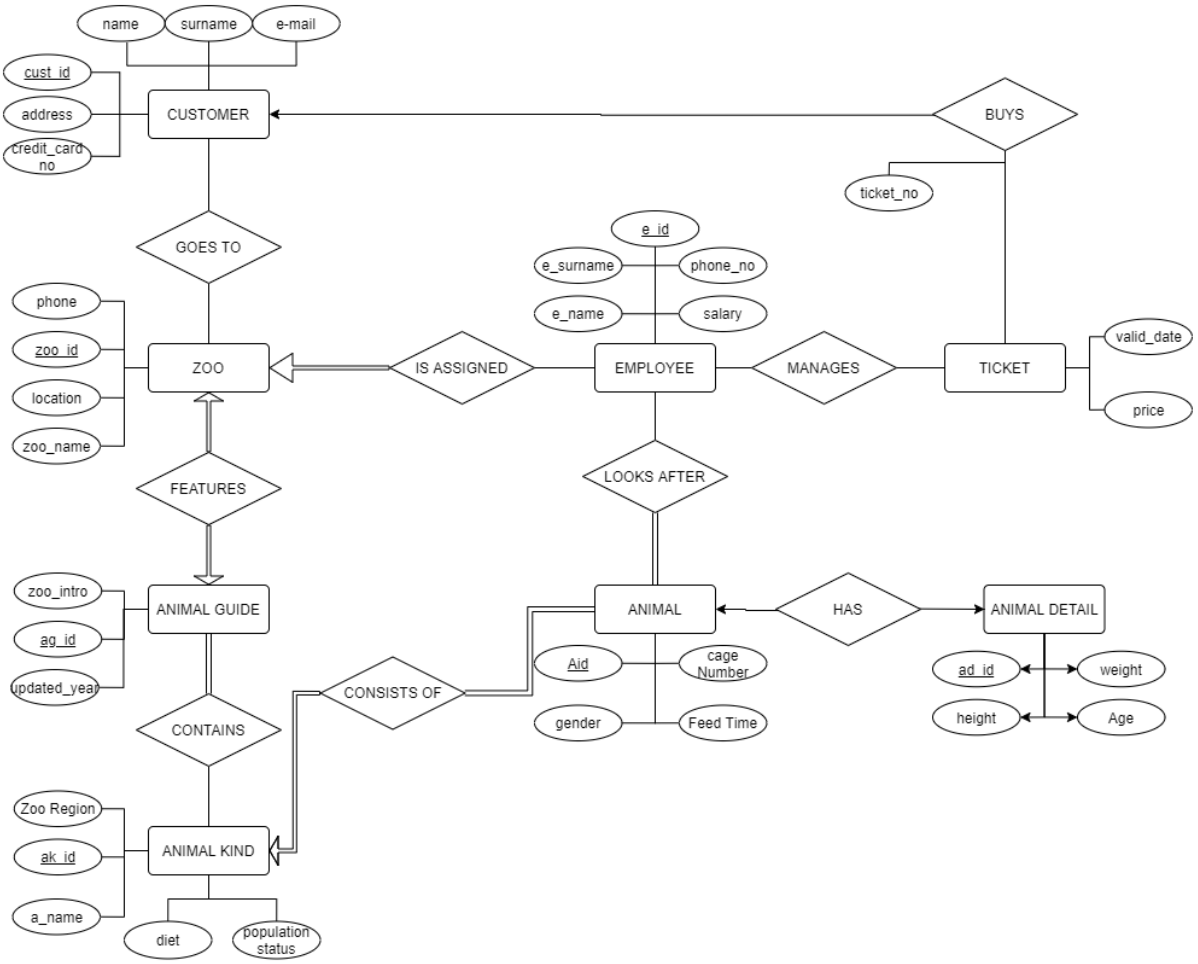
Employees are assigned zoos and every zoo has at least one employee. Each employee's name, surname, phone number, and salary must be stored and also every employee has ana unique id number that assigned itself. Also, employees look after animal and al least one animal must assign an employee.

Zoos has phone numbers, unique id, location, name. Each zoo has an animal guide and animal guides keep some information about zoo. In guides zoo intro data and updated year info must be stored. Each guide has a unique id.

Each animal guide keeps information about animal kind and animal kind find at least one animal guide. Every animal kind keeps the data that zoo region, kind name, diet and population status. Also, each one has a unique id.

Animals consist of animal kind so each animal consist of animal kind. Each animal has a unique id, cage number, gender, feed time. Each animal has an animal detail. In this animal detail we are desired to store height, weight, age that animal. In addition, each animal detail has an id number.

E-R Diagram


name
surname
e-mail
cust_id
address
credit_card no
CUSTOMER
BUYS
ticket_no
GOES TO
e_id
e_surname
phone_no
e_name
salary
phone
zoo_id
location
zoo_name
ZOO
IS ASSIGNED
EMPLOYEE
MANAGES
TICKET
valid_date
price
FEATURES
LOOKS AFTER
zoo_intro
ag_id
updated_year
ANIMAL GUIDE
ANIMAL
HAS
ANIMAL DETAIL
Aid
cage Number
gender
Feed Time
ad_id
weight
height
Age
CONSISTS OF
CONTAINS
Zoo Region
ak_id
a_name
ANIMAL KIND
diet
population status

CREATING TABLES SQL QUERY

```
CREATE TABLE CUSTOMERS(
CUST_ID INT PRIMARY KEY,
ADDRESS VARCHAR(15),
NAME VARCHAR(15) NOT NULL,
SURNAME VARCHAR(15) NOT NULL,
EMAIL VARCHAR(15) NOT NULL,
C_CARDNO VARCHAR(15)
)

CREATE TABLE TICKETS(

TICKET_NO INT PRIMARY KEY,
VALID_DATE SMALLDATETIME NOT NULL,
PRICE MONEY,
CUST_NUM INT
)

CREATE TABLE ZOO(

ZOO_ID INT PRIMARY KEY,
LOCATION VARCHAR(50),
ZOO_NAME VARCHAR(15) NOT NULL,
PHONE VARCHAR(15) UNIQUE
)



CREATE TABLE ANIMALGUIDE(

AG_ID INT PRIMARY KEY,
ZOO_INTRO VARCHAR(100),
UPDATEDYEAR SMALLDATETIME,
ZOO_ID INT UNIQUE
)

CREATE TABLE ANIMALKIND(
AK_ID INT PRIMARY KEY,
ZOO_REGION VARCHAR(15),
KIND_NAME VARCHAR(15),
DIET VARCHAR(15),
POPULATION_STATUS VARCHAR(10),
)


CREATE TABLE ANIMAL(
A_KIND_ID INT,
ANIMAL_ID INT PRIMARY KEY,
CAGE_NUMBER INT,
GENDER CHAR(1),
FEEDTIME DATETIME,
)


CREATE TABLE ANIMAL_DETAIL(
AD_ID INT PRIMARY KEY,
A_ID INT,
HEIGHT FLOAT,
WEIGHT FLOAT,
AGE TINYINT,
)
```

```
CREATE TABLE EMPLOYEE(
E_ID INT PRIMARY KEY,
NAME VARCHAR(15),
SURNAME VARCHAR(15),
PHONE_NUM VARCHAR(15),
SALARY MONEY,
ZOO_ID INT
)

CREATE TABLE LOOKSAFTER(
EMP_ID INT NOT NULL,
ANIMAL_ID INT NOT NULL
)


CREATE TABLE GOESTO(
C_ID INT NOT NULL,
Z_ID INT NOT NULL
)
```

INSERTING DATA

```
INSERT INTO CUSTOMERS VALUES (
1,
'ISTANBUL',
'CENGIZ',
'HATIP',
'ch@gmail.com',
123)

INSERT INTO CUSTOMERS VALUES (
2,
'ANKARA',
'BURAK',
'KARA',
'brk@gmail.com',
1234)

INSERT INTO CUSTOMERS VALUES (
3,
'EDIRNE',
'YASIN',
'TRKYLMZ',
'97htt@gmail.com',
1235)

INSERT INTO  TICKETS VALUES(
1,
'08-16-2020',
5,
2
)
INSERT INTO  TICKETS VALUES(
2,
'08-16-2021',
```

```
5,
1
)
INSERT INTO  TICKETS VALUES(
3,
'01-16-2020',
5,
3
)
INSERT INTO  TICKETS VALUES(
4,
'08-16-2022',
5,
NULL
)


INSERT INTO  ZOO VALUES(
1,
'ISTANBUL',
'DARICA',
'2122800066'
)
INSERT INTO  ZOO VALUES(
2,
'ISTANBUL',
'EYUP',
'2123250066'
)
INSERT INTO  ZOO VALUES(
3,
'ISTANBUL',
'KAGITHANE',
'2124272792'
)

INSERT INTO GOESTO VALUES(1,3)
INSERT INTO GOESTO VALUES(2,1)
INSERT INTO GOESTO VALUES(3,2)


INSERT INTO ANIMALGUIDE VALUES(
1,
'DARICA ZOO IS A...',
'08-16-2005',
1)
INSERT INTO ANIMALGUIDE VALUES(
2,
'Kağıthane ZOO IS A...',
'10-01-2006',
2)
INSERT INTO ANIMALGUIDE VALUES(
3,
'Eyüp ZOO IS A...',
'11-10-2012',
3)

/* ANIMLA KIND */
INSERT INTO ANIMALKIND VALUES(
1,
'SAFARI',
'CHEETAH',
```

```
'MEAT',
'WIDE')

INSERT INTO ANIMALKIND VALUES(
2,
'FOREST',
'PANDA',
'VEGETABLES',
'RARE')
INSERT INTO ANIMALKIND VALUES(
3,
'EGYPT',
'CAMEL',
'VEGETABLES',
'WIDE')



/*ANIMAL*/

INSERT INTO ANIMAL VALUES(
1,1,100,'M','01-01-2000 16:00')

INSERT INTO ANIMAL VALUES(
2,2,101,'F','01-01-2000 17:00')

INSERT INTO ANIMAL VALUES(
3,3,103,'M','01-01-2000 16:00')

/* ANIMAL DETAIL */

INSERT INTO ANIMAL_DETAIL VALUES(
1,1,1.5,50,5)
INSERT INTO ANIMAL_DETAIL VALUES(
2,2,1,30,4)
INSERT INTO ANIMAL_DETAIL VALUES(
3,3,2.1,70,8)


/*EMPLOYEE*/

INSERT INTO EMPLOYEE VALUES(
1,'YUSUF','GUNGOR','21221221',1500,1)


INSERT INTO EMPLOYEE VALUES(
2,'ESRA','LARA','21225621',2500,2)

INSERT INTO EMPLOYEE VALUES(
3,'YASIN','UMUT','21267221',1000,3)


/*looks after*/

INSERT INTO LOOKSAFTER VALUES(1,2)
INSERT INTO LOOKSAFTER VALUES(2,1)
INSERT INTO LOOKSAFTER VALUES(3,3)
```

QUERIES

- A query that applies a row and column selection to a single table

```
/* SHOW ANIMAL IDS OF THE ANIMAL WHO ARE MALE*/


SELECT ANIMAL_ID
FROM ANIMAL
WHERE GENDER = 'M'
```

- A query that takes join of two tables and does row and column selection and ordering.

```
/* SHOW CAGENUMBER AND HEIGHT  OF ANIMALS THAT AGES ARE GREATER THAN 5 AND ORDER THEM
WITH ASCENDING ORDER BY  HEIGHT*/


SELECT CAGE_NUMBER, HEIGHT
FROM ANIMAL JOIN ANIMAL_DETAIL ON ANIMAL_ID = A_ID
WHERE AGE>5
ORDER BY HEIGHT ASC
```

- Repeat the previous task but this time your query should take join of three tables

```
/*SHOW NAME OF THE EMPLOYEES WHO LOOK AFTER ANIMALS THAT OVER 5 AGE AND ORDER THEM
WİTH ASCENDING ORDER*/


SELECT EMPLOYEE.NAME
FROM EMPLOYEE JOIN LOOKSAFTER ON EMP_ID=E_ID JOIN ANIMAL_DETAIL ON ANIMAL_ID=A_ID
WHERE AGE>5
ORDER BY NAME ASC
```

- Write a query involving outer join and row selection.

```
/* show the name and surname of the employees which is works on Istanbul zoo and how
are salary is greater than 1000*/



SELECT EMPLOYEE.NAME,EMPLOYEE,SURNAME
FROM EMPLOYEE FULL OUTER JOIN ZOO ON EMPLOYEE.ZOO_ID = ZOO.ZOO_ID
WHERE LOCATION = 'Istanbul' and SALARY>1000
```

- Write two queries involving summary functions and row selection and join operations.

```
/*mean of animals age which are male */

SELECT AVG(age)
FROM ANIMAL JOIN ANIMAL_DETAIL ON A_ID=ANIMAL_ID
WHERE gender='M'


/*COUNT THE NUMBER OF EMPLOYEE WHO LOOKS AFTER THE ANIMALS WHİCH ARE IN THE CAGE
NUMBER 101*/

SELECT COUNT(DISTINCT EMP_ID)
FROM ANIMAL JOIN LOOKSAFTER ON ANIMAL.ANIMAL_ID = LOOKSAFTER.ANIMAL_ID
WHERE CAGE_NUMBER = 101
```

- Write two queries involving join, grouping, row selection and ordering

```
/* SHOW ANIMAL KINDS AND THE NUMBERS OF ANIMAL THAT'S TYPE WİTH ASCENDİNG ORDER
ACCORDING TO NUMBER OF THAT TYPE ANIMAL */

SELECT KIND_NAME,COUNT(ANIMAL_ID) AS COUNT
FROM ANIMALKIND JOIN ANIMAL ON AK_ID=A_KIND_ID
GROUP BY KIND_NAME
ORDER BY COUNT(ANIMAL_ID) ASC

/* SHOW NAME OF CUSTOMERS AND THE AVG PAID MONEY FOR TICKET THAT CUSTOMER AND ORDER
THEM WITH DESCENDING ORDER WITH RESPECT TO PAID MONEY
*/

SELECT NAME,AVG(PRICE) AS AVG_PAID_MONEY
FROM CUSTOMERS JOIN TICKETS ON CUST_NUM=CUST_ID
GROUP BY CUSTOMERS.NAME
ORDER BY AVG(PRICE) DESC
```

- Write two queries involving join, grouping and group selection

```
/* SHOW ANIMAL KINDS AND THE NUMBERS OF ANIMAL THAT'S TYPE WİTH ASCENDİNG ORDER
ACCORDING TO NUMBER OF THAT TYPE ANIMAL  WHICH ARE NUMBER IS GREATER THAN 50*/

SELECT KIND_NAME,COUNT(ANIMAL_ID) AS COUNT
FROM ANIMALKIND JOIN ANIMAL ON AK_ID=A_KIND_ID
GROUP BY KIND_NAME
HAVING COUNT(ANIMAL_ID)>50
ORDER BY COUNT(ANIMAL_ID) ASC

/* SHOW NAME OF CUSTOMERS WHO BOUGHT MORE THAN TWO TICKET AND THE AVG PAID MONEY FOR
TICKET THAT CUSTOMER AND ORDER THEM WITH DESCENDING ORDER WITH RESPECT TO PAID MONEY
*/

SELECT NAME,AVG(PRICE) AS AVG_PAID_MONEY
FROM CUSTOMERS JOIN TICKETS ON CUST_NUM=CUST_ID
GROUP BY CUSTOMERS.NAME
HAVING COUNT(TICKET_NO)>2
ORDER BY AVG(PRICE) DESC
```

- Write a query involving join, grouping, group selection and row selection

```
/* SHOW ANIMAL KINDS WHICH ARE FEMALE AND THE NUMBERS OF ANIMAL THAT'S TYPE WİTH
ASCENDİNG ORDER ACCORDING TO NUMBER OF THAT TYPE ANIMAL  WHICH ARE NUMBER IS GREATER
THAN 50*/

SELECT KIND_NAME,COUNT(ANIMAL_ID) AS COUNT
FROM ANIMALKIND JOIN ANIMAL ON AK_ID=A_KIND_ID
WHERE GENDER='F'
GROUP BY KIND_NAME
HAVING COUNT(ANIMAL_ID)>50
ORDER BY COUNT(ANIMAL_ID) ASC

/* SHOW NAME OF CUSTOMERS WHO BOUGHT MORE THAN TWO TICKET WHICH IS GREATER THAN 50
DOLAR AND THE AVG PAID MONEY FOR TICKET THAT CUSTOMER AND ORDER THEM WITH DESCENDING
ORDER WITH RESPECT TO PAID MONEY
*/

SELECT NAME,AVG(PRICE) AS AVG_PAID_MONEY
FROM CUSTOMERS JOIN TICKETS ON CUST_NUM=CUST_ID
WHERE PRICE>50
GROUP BY CUSTOMERS.NAME
HAVING COUNT(TICKET_NO)>2
ORDER BY AVG(PRICE) DESC
```