|  |
| --- |
|  |
| Implementation of Database for an Animal Clinic |
| ASSIGNMENT-2 |
|  |
|  |
|  |

|  |
| --- |
|  |

Student Name: ARUN BABU KOLLASSERY

Student ID : 30020744

Date : 14/12/2019

Table of Contents

|  |  |  |
| --- | --- | --- |
| 1. | Introduction | 3 |
| 2. | Physical Design | 4 |
| 2.1 Data Volume Map | 4 |
| 2.2 Create Queries for Tables | 5 |
| 2.3 Insertion Queries | 7 |
| 2.4 Database Diagram | 9 |
| 2.5 Index Files Creation | 10 |
| 2.6 De-normalisation | 11 |
| 2.7 Report | 12 |
| 2.8 Data Dictionary | 13 |

1. Introduction

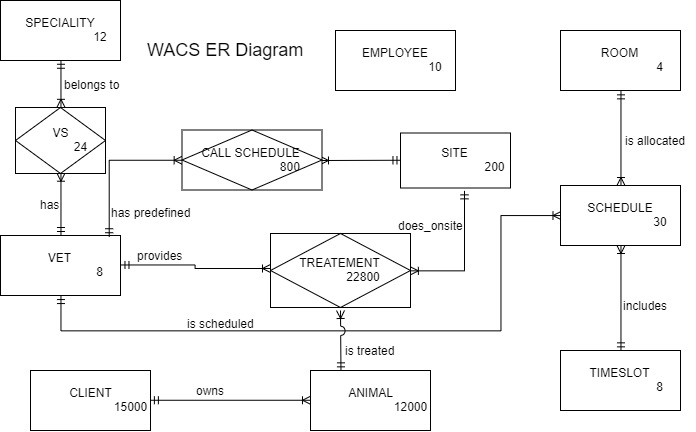
WACS (Wopwop Animal Care Service) is an animal clinic, aims to be the leading animal care in the region. Their prime goal is to build a new database system that’s to be more robust. Thereby, the system can deliver accurate information to the employees and clients regarding the availability of vets and the treatments given to animals.

WACS used a paper-based system to keep track of duty schedules of veterinarians, allocations of animals to vets, and the treatments given to the animals. This system often caused problems or failed to maintain the track of services specified above. So, the clinic is expecting the proposed system will improve the efficiency of WACS services as based on the increase in client satisfaction. The manager of the clinic wishes a rise of 15% in the annual profit after the implementation of a new database system.

2. Physical Design

2.1 Data Volume Map

There are certain requirements given in the case study, based on that, the data volume map of the database is shown below.



|  |  |  |  |
| --- | --- | --- | --- |
| Entity | Size of each row | Records | Total Bytes |
| Employee | 136 | 10 | 1360 |
| Room | 36 | 4 | 144 |
| Time slot | 14 | 8 | 112 |
| Vet | 140 | 8 | 1120 |
| Speciality | 36 | 12 | 432 |
| VS | 8 | 24 | 192 |
| Client | 140 | 15000 | 2100000 |
| Animal | 40 | 12000 | 480000 |
| Schedule | 48 | 30 | 1440 |
| Call Schedule | 18 | 800 | 14400 |
| Treatment | 61 | 22800 | 1390800 |
| Site | 104 | 200 | 20800 |

2.2 Create Queries for Tables

Create table queries for generating tables in the new database using Transact –SQL commands in Microsoft SQL server.

Requirements to meet:-

* Specify primary keys and referential integrity constraints.
* Use appropriate on-delete action if a table has foreign key(s).

CREATE TABLE employee(

E# INT IDENTITY(1,1)NOT NULL,

E\_Firstname VARCHAR(30) NULL,

E\_Surname VARCHAR(30) NULL,

StreetNo INT NULL,

StreetName VARCHAR(30) NULL,

Suburb VARCHAR(30) NULL,

CONSTRAINT E\_pk PRIMARY KEY (E#));

CREATE TABLE room(

R# INT IDENTITY(1,1)NOT NULL,

RoomName VARCHAR(30) NULL,

CONSTRAINT R\_pk PRIMARY KEY(R#))

CREATE TABLE speciality(

SP# INT IDENTITY(1,1)NOT NULL,

description VARCHAR(30) NULL,

CONSTRAINT spec\_pk PRIMARY KEY (SP#))

CREATE TABLE timeslot(

slot# INT IDENTITY(1,1)NOT NULL,

frm TIME NULL,

tto TIME NULL,

CONSTRAINT slot\_PK PRIMARY KEY (slot#))

CREATE TABLE vet(

V# INT IDENTITY(1,1)NOT NULL,

V\_Firstname VARCHAR(30) NULL,

V\_Surname VARCHAR(30) NULL,

Streetno INT NULL,

StreetName VARCHAR(30) NULL,

Suburb VARCHAR(30) NULL,

ContactNo INT NULL,

CONSTRAINT vet\_PK PRIMARY KEY (V#))

CREATE TABLE vs(

V# INT NOT NULL,

SP# INT NOT NULL,

CONSTRAINT vs\_pk PRIMARY KEY(V#,SP#),

CONSTRAINT v\_fk FOREIGN KEY(V#) REFERENCES vet(V#)ON DELETE CASCADE,

CONSTRAINT sp\_fk FOREIGN KEY(SP#) REFERENCES speciality(SP#)ON DELETE CASCADE)

CREATE TABLE client(

C# INT IDENTITY(1,1)NOT NULL,

c\_firstname VARCHAR(30),

c\_surname VARCHAR(30),

streetno INT,

streetname VARCHAR(30),

suburb VARCHAR(30),

contactno INT,

CONSTRAINT client\_pk PRIMARY KEY(C#))

CREATE TABLE animal(

A# INT IDENTITY(1,1)NOT NULL,

aname VARCHAR(30),

C# INT NOT NULL,

CONSTRAINT animal\_pk PRIMARY KEY(A#),

CONSTRAINT animal\_client\_fk FOREIGN KEY(C#) REFERENCES client(C#)ON DELETE CASCADE)

CREATE TABLE schedule (

sch# INT IDENTITY(1,1)NOT NULL,

v# INT,

slot# INT,

R# INT,

oncall VARCHAR(30),

CONSTRAINT schedule\_pk PRIMARY KEY(sch#),

CONSTRAINT sch\_room\_fk FOREIGN KEY(R#) REFERENCES room(R#)ON DELETE CASCADE,

CONSTRAINT sch\_vet\_fk FOREIGN KEY(V#) REFERENCES vet(V#)ON DELETE CASCADE,

CONSTRAINT sch\_timeslot\_fk FOREIGN KEY(slot#) REFERENCES timeslot(slot#)ON DELETE CASCADE)

CREATE TABLE site(

site# INT IDENTITY(1,1)NOT NULL,

sitetype VARCHAR(30),

streetno INT,

streetname VARCHAR(30),

suburb VARCHAR(30),

CONSTRAINT site\_pk PRIMARY KEY(site#))

CREATE TABLE call\_schedule(

V# INT NOT NULL,

site# INT NOT NULL,

from\_time TIME,

to\_time TIME,

CONSTRAINT call\_pk PRIMARY KEY(V#,site#),

CONSTRAINT call\_vet\_fk FOREIGN KEY(V#) REFERENCES vet(V#)ON DELETE CASCADE,

CONSTRAINT call\_site\_fk FOREIGN KEY(site#) REFERENCES site(site#)ON DELETE CASCADE)

CREATE TABLE treatment(

T# INT IDENTITY(1,1)NOT NULL,

date\_of\_treatment DATE,

V# INT NOT NULL,

A# INT NOT NULL,

site# INT,

treatment\_given VARCHAR(40),

CONSTRAINT treatment\_pk PRIMARY KEY(T#),

CONSTRAINT treatment\_vet\_fk FOREIGN KEY(V#) REFERENCES vet(V#)ON DELETE CASCADE,

CONSTRAINT treatment\_animal\_fk FOREIGN KEY(A#) REFERENCES animal(A#)ON DELETE CASCADE,

CONSTRAINT treatment\_site\_fk FOREIGN KEY(site#) REFERENCES site(site#)ON DELETE CASCADE)

2.3 Insertion Queries

Insert commands for populating records (at least 5) into the tables created above.

INSERT INTO employee VALUES ('EDWIN','PAUL',111,'KENT ST','GREERTON'),

('JOHN','PAUL',122,'SHEPPARD ST','GREERTON'),

('SERA','JAMES',133,'SEFTON ST','BETHLEHEM'),

('STEVE','MARK',144,'SHERWIN ST','TE PUNA'),

('MARIA','GEORGE',155,'SHELLEY ST','PAPAMOA')

INSERT INTO room VALUES ('CONSULTATION ROOM 1'),

('CONSULTATION ROOM 2'),

('SURGICAL ROOM 1'),

('SURGICAL ROOM 2'),

('PREP ROOM 2')

INSERT INTO speciality VALUES ('Small animals'),

('Large animals'),

('Exotic'),

('Equine'),

('Wild')

INSERT INTO timeslot VALUES ('9:00','10:00'),

('10:00','11:00'),

('11:00','12:00'),

('13:00','14:00'),

('14:00','15:00')

INSERT INTO vet VALUES ('SAM','THOMAS',121,'Paine St','Pyes Pa',123456789),

('KEN','GEORGE',122,'Park St','Matapihi',234567891),

('SACHARIA','JOHNSON',123,'David St','Otumoetai',345678912),

('JULI','JUDE',124,'Hynds Rd','Greerton',456789123),

('TEENA','SMITH',125,'Waratah St','Bethlehem',567891234)

INSERT INTO vs VALUES (2,1),

(3,4),

(5,3),

(1,2),

(4,5)

INSERT INTO client VALUES ('Jack','Daniel',123,'Sherwin St','Te Puna',678901234),

('Jacob','Samuel',133,'Sefton St','Bethlehem',678901234),

('Charlie','William',143,'Waratah St','Bethlehem',678901234),

('Ella','Jordan',153,'Shelley St','Papamoa',678901234),

('Amelia','Benjamin',163,'Park St','Matapihi',678901234)

INSERT INTO animal VALUES ('Tobby',3),

('Rosy',5),

('Cezar',2),

('Tiger',1),

('Showy',4)

INSERT INTO schedule VALUES (2,1,2,''),

(3,4,4,''),

(1,2,1, 'AVAILABE'),

(5,3,5,''),

(4,5,2,'')

INSERT INTO site VALUES ('Clinic',100,'Pyes Pa','GREERTON'),

('Farm',101,'Gate Pa','GREERTON'),

('Zoo',102,'Spring St','Tauranga South'),

('Hospital',103,'Selwyn St','Hairini'),

('House',104,'Durham St','Tauranga South')

INSERT INTO call\_schedule VALUES (1,2,'10:00:00','11:00:00'),

(2,5,'09:00:00','10:00:00'),

(3,3,'13:00:00','14:00:00'),

(4,1,'14:00:00','15:00:00'),

(5,4,'11:00:00','12:00:00')

INSERT INTO treatment VALUES ('20191125',2,1,5,'Leg operation/Antibiotic'),

('20191126',4,2,1,'Pain killer/Antibiotic'),

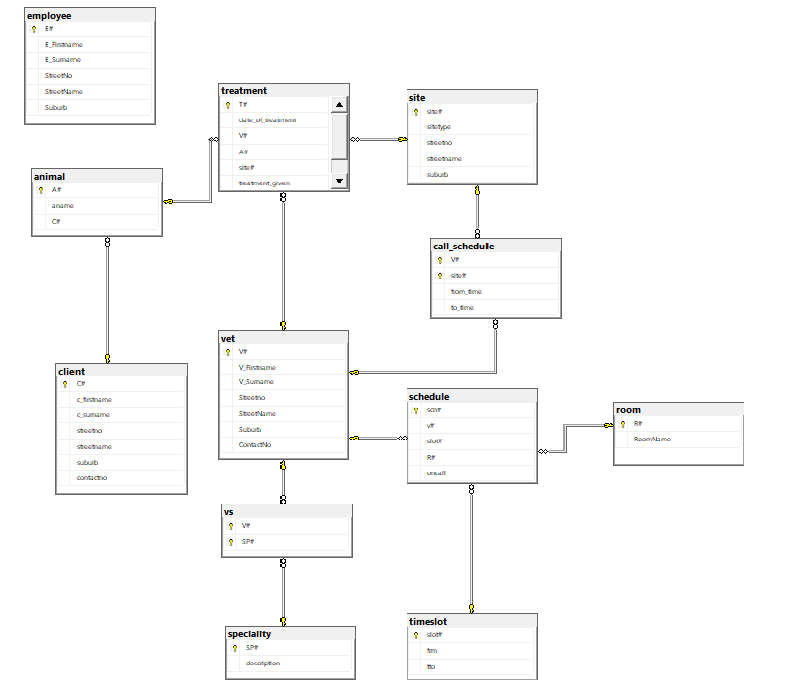
('20191127',1,4,2,'Ham Operation/Antibiotic'),

('20191128',3,5,3,'Bacterial Infection/TT'),

('20191129',5,3,4,'Fungal Infection/Antifungals')

2.4 Database Diagram

The database diagram of 12 tables in the wacs database is shown below.



2.5 Index Files Creation

In this case study, to improve the performance of retrieving data can be done easily using index files. It is assumed that there are at least 20000 records in a table. If so, the commands for creating index files on different tables are given below.

Employee

Here in the employee table, the number of records will increase considerably in the future as a result of new employment. So, creating an index on the employee table will make it faster to fetch data from the database.

CREATE INDEX indx\_emp

ON employee (E#);

SELECT \* FROM employee WITH(INDEX(indx\_emp))

Client

Day by day the number of clients, who comes to the veterinary will also increase and so the database should capable to store and retrieve data effectively. For that, creating an index on the client table should be considered.

CREATE INDEX indx\_C#

ON client (C#);

SELECT \* FROM client WITH(INDEX(indx\_C#))

Animal

Even if it is an animal, the database should keep all the records of animals visited in the veterinary from the time of database created. Hence, the index creation on the animal table will improve the overall performance.

CREATE INDEX indx\_A#

ON animal (A#);

SELECT \* FROM animal WITH(INDEX(indx\_A#))

Vet

Here the vet stands veterinarian. It’s also similar to the case of client and employee tables that the record of vets will gradually increase in the future. Thus, the index on the vet table will make it faster to retrieve data.

CREATE INDEX indx\_vet

ON vet (V#);

SELECT \* FROM vet WITH(INDEX(indx\_vet))

Site

Due to the rise of population and animals, the number of veterinary sites will also increase. Similarly, it’s necessary to create an index for the site table to get data quickly.

CREATE INDEX indx\_site

ON site (site#);

SELECT \* FROM site WITH(INDEX(indx\_site))

2.6 De-normalisation

a) Suggestion

In this database, many tables referenced the primary key of other tables. Hence, it might cause redundancy/duplication issues. The use of de-normalisation technique by merging tables in a suitable format will help to reduce the tables; as a result, it can overcome these issues as well.

b) Example

Consider the tables vet, vs and speciality.

The vet table contains:-

* V# (Primary Key)
* V\_Firstname
* V\_Surname
* Streetno
* StreetName
* Suburb
* ContactNo

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| vet | | | | | | |
| V# | V\_Firstname | V\_Surname | Streetno | StreetName | Suburb | ContactNo |

The vs table contains:-

* V# (Primary Key, Foreign Key)
* SP# (Primary Key, Foreign Key)

|  |  |
| --- | --- |
| vs | |
| V# | SP# |

The speciality table contains:-

* SP# (Primary Key)
* description

|  |  |
| --- | --- |
| speciality | |
| SP# | description |

De-normalised relation:-

If we apply the de-normalisation technique in the above tables, then we will get the de-normalised relation like this

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| vet | | | | | | |
| V# | V\_Firstname | V\_Surname | Streetno | StreetName | Suburb | ContactNo |

|  |  |  |
| --- | --- | --- |
| vs | | |
| V# | SP# | description |

c) Pros of de-normalisation:-

* It can reduce the number of tables.
* Improve overall performance as a result of faster execution.

d) Cons of de-normalisation:-

* Limiting the flexibility for faster execution.
* Insertion and update queries will be difficult.
* It will affect the readability since the merged table contains unrelated data.

2.7 Report

The reports for the following requirements are given below.

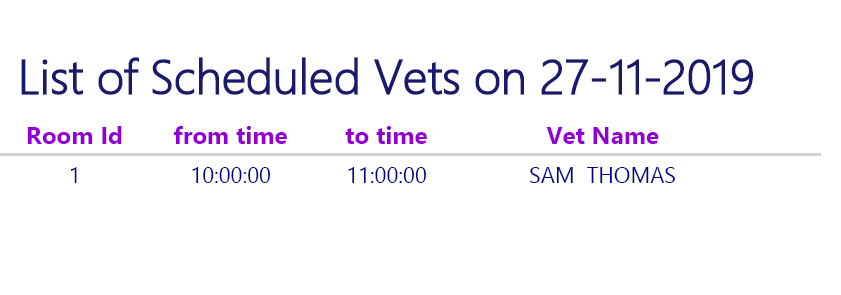
a. List of Clients (first name, surname), animals and treatments given.

b. List of scheduled vets (first name, surname) on a particular day.

Name of the database: - wacs

a)



b)

2.8 Data Dictionary

The data dictionary of 12 tables of the WACS database system is shown below.























