TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CUỐI KÌ MÔN HỆ CƠ SỞ DỮ LIỆU**

**Báo cáo cuối kì môn hệ cơ sở dữ liệu**

*Người hướng dẫn*:**THẦY DZOÃN XUÂN THANH**

*Người thực hiện*: **PHẠM THÁI HỌC - 52000047**

Lớp **: 20050201**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CUỐI KÌ MÔN HỆ CƠ SỞ DỮ LIỆU**

**Báo cáo cuối kì môn hệ cơ sở dữ liệu**

*Người hướng dẫn*: **THẦY DZOÃN XUÂN THANH**

Người thực hiện:  **PHẠM THÁI HỌC - 52000047**

Lớp **: 20050201**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022**

LỜI CẢM ƠN

Em xin gữi lời cảm ơn đến thầy Dzoãn Xuân Thanh đã hổ trợ và giảng dạy cho tụi em để có đủ kiến thức hoàn thành môn học.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi và được sự hướng dẫn của thầy Dzoãn Xuân Thanh. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 14 tháng 06 năm 2022*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Phạm Thái Học*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Tổng hợp kiến thức môn học qua các bài tập.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc106135857)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc106135858)

[TÓM TẮT iv](#_Toc106135859)

[MỤC LỤC 1](#_Toc106135860)

[Phần 1. (3 điểm) 4](#_Toc106135861)

[Đề: 4](#_Toc106135862)

[Câu Trả lời: 5](#_Toc106135863)

[1. Viết các câu lệnh SQL để thực hiện các thao tác sao: (0.5 điểm) 5](#_Toc106135864)

[Kết quả của code: 7](#_Toc106135865)

[2. Viết các hàm sau: (1.0 điểm) 7](#_Toc106135866)

[a. Viết hàm phát sinh mã số tự động cho bảng thuê sân theo nguyên tắc: Ngày thuê sân + số thứ tự gồm 3 chữ số tăng dần. Ví dụ: Ngày thuê sân là ngày 20/06/2022 và hiện đã có lần thuê sân thứ 3 (Bảng Thuê sân đang có mã số là: “20062022003” thì mã số tự động tiếp theo sẽ là “20062022004” (0.5 điểm) 7](#_Toc106135867)

[b. Viết hàm trả về số tiền thuê sân khi biết được giờ vào sân và giờ ra khỏi sân (0.5 điểm) 8](#_Toc106135868)

[Kết quả code: 9](#_Toc106135869)

[3. Viết các thủ tục sau: (0.5 điểm) 9](#_Toc106135870)

[a. Thêm vào bảng Giá tiền thuê sân có kiểm tra khoá chính, giờ vào phải nhỏ hơn giờ ra. (0.25 điểm) 9](#_Toc106135871)

[b. Cập nhật đơn giá cho dịch vụ thuê áo thêm 10.000 (0.25 điểm). 9](#_Toc106135872)

[Kết quả của code: 10](#_Toc106135873)

[4. Viết 1 trigger sau: (1 điểm) 11](#_Toc106135874)

[Khi thêm, xoá, sửa dữ liệu trong bảng Chi tiết thuê sân: 11](#_Toc106135875)

[Kết quả của code: 13](#_Toc106135876)

[Phần 2: (2 điểm) 15](#_Toc106135877)

[Đề: 15](#_Toc106135878)

[Câu Trả lời: 15](#_Toc106135879)

[a) Tìm tất cả các khóa của lược đồ quan hệ: 15](#_Toc106135880)

[b. Tìm phủ tối thiểu của tập phụ thuộc hàm (0.5 điểm) 16](#_Toc106135881)

[c. Xác định dạng chuẩn cao nhất của lược đồ quan hệ (0.5 điểm) 17](#_Toc106135882)

[d. Nếu lược đồ quan hệ chưa đạt dạng chuẩn BC, đưa lược đồ về dạng chuẩn BC (0.5 điểm) 18](#_Toc106135883)

[Phần 3: (3 điểm) 18](#_Toc106135884)

[Đề: 18](#_Toc106135885)

[Câu trả lời: 18](#_Toc106135886)

[a. Viết đặc tả: 18](#_Toc106135887)

[b. Vẽ ERD: 20](#_Toc106135888)

[c. Mô hình quan hệ: 20](#_Toc106135889)

[Phần 4: (2 điểm) 21](#_Toc106135890)

[Đề: 21](#_Toc106135891)

[Câu trả lời (python): 21](#_Toc106135892)

[1 Cho lược đồ quan cho biết các khóa của lược đồ quan hệ: 21](#_Toc106135893)

[**Ý tưởng code:** 21](#_Toc106135894)

[**Source Code và comment code (được viết bằng python):** 22](#_Toc106135895)

[**Kết quả chạy code dựa trên bài mẫu ở phần 2: xuất ra tập khóa và siêu khóa** 27](#_Toc106135896)

[2. Cho LĐCSDL, cho biết phủ tối thiểu của LĐCSDL (0.75 điểm) 27](#_Toc106135897)

[**Ý tưởng code:** 27](#_Toc106135898)

[**Source Code:** 28](#_Toc106135899)

[**Kết quả chạy code:** 28](#_Toc106135900)

[3. Cho LĐCSDL, cho biết dạng chuẩn cao nhất của LĐCSDL. Giải thích tại sao. (0.75 điểm) 28](#_Toc106135901)

[**Ý tưởng code:** 28](#_Toc106135902)

[**Source Code:** 31](#_Toc106135903)

[**Kết quả chạy code:** 39](#_Toc106135904)

Phần 1. (3 điểm)

Đề:

**Cho lược đồ CSDL quản lý Sân bóng đá mini như sau:**

* Sân bóng (mã sân bóng, tên sân bóng, loại sân)

*Tân từ:* Sân bóng có mã sân bóng là khoá chính, tên sân bóng, loại sân cho biết sân đó được thi đấu bao nhiêu người. Ví dụ: sân 5, sân 7, sân 11, …

* Dịch vụ (mã dịch vụ, tên dịch vụ, số lượng, đơn giá)

*Tân từ:* Bảng dịch vụ thể hiện các dịch vụ mà sân bóng cung cấp, như: thuê áo đấu, giày thi đấu, nước uống, ….

* Giá tiền thuê sân (Mã giá tiền, từ giờ, đến giờ, số tiền)

*Tân từ:* Tuỳ thuộc vào khung giờ thi đấu mà có giá tiền khác nhau. Ví dụ: Từ 6:00 AM đến 8:00 AM thì mỗi giờ tốn 300.000, từ 10:00 AM đến 11:59 AM thì mỗi giờ tốn 100.000.

* Thuê sân (Mã thuê sân, điện thoại khách hàng, mã sân, ngày thuê sân, giờ bắt đầu, giờ kết thúc, tổng tiền)

*Tân từ:* Khi khách hàng đến thuê sân, hệ thống lưu lại các thông tin như: mã thuê sân, số điện thoại khách hàng thuê sân, mỗi lần thuê sân chỉ được thuê 1 sân (mã sân), ngày thuê sân, giờ bắt đầu và giờ kết thúc trận đấu; tổng tiền cho biết tổng tiền thuê sân thi đấu và những dịch vụ đã sử dụng.

* Chi tiết thuê sân (Mã thuê sân, mã dịch vụ, số lượng, đơn giá)

*Tân từ:* Mỗi lần thuê sân thi đấu khách hàng có thể sử dụng một số dịch vụ mà sân bóng cung cấp, như: uống nước, thuê áo, …

Tân từ:

1. **Viết các câu lệnh SQL để thực hiện các thao tác sao: (0.5 điểm)**
   1. Tạo Cơ sở dữ liệu, tạo bảng bao gồm khoá chính, khoá ngoại. (0.25 điểm)
   2. Thêm dữ liệu cho các bảng theo đúng trình tự bên 1 trước bên nhiều sau. Mỗi bảng ít nhất 5 dòng dữ liệu. (0.25 điểm).
2. **Viết các hàm sau: (1.0 điểm)**
3. Viết hàm phát sinh mã số tự động cho bảng thuê sân theo nguyên tắc: Ngày thuê sân + số thứ tự gồm 3 chữ số tăng dần. Ví dụ: Ngày thuê sân là ngày 20/06/2022 và hiện đã có lần thuê sân thứ 3 (Bảng Thuê sân đang có mã số là: “20062022003” thì mã số tự động tiếp theo sẽ là “20062022004” (0.5 điểm)
4. Viết hàm trả về số tiền thuê sân khi biết được giờ vào sân và giờ ra khỏi sân (0.5 điểm)
5. **Viết các thủ tục sau: (0.5 điểm)**
   1. Thêm vào bảng Giá tiền thuê sân có kiểm tra khoá chính, giờ vào phải nhỏ hơn giờ ra. (0.25 điểm)
   2. Cập nhật đơn giá cho dịch vụ thuê áo thêm 10.000 (0.25 điểm).
6. **Viết 1 trigger sau: (1 điểm)**

Khi thêm, xoá, sửa dữ liệu trong bảng Chi tiết thuê sân:

* Cập nhật lại Tổng tiền trong bảng thuê sân. Trong đó, tổng tiền = tiền thuê sân + tiền sử dụng dịch vụ của lần thuê sân đó (0.5 điểm)
* Đơn giá trong bảng Chi tiết thuê sân phải bằng với đơn giá trong bảng dịch vụ (0.5 điểm)

Câu Trả lời:

1. Viết các câu lệnh SQL để thực hiện các thao tác sao: (0.5 điểm)
   1. Tạo Cơ sở dữ liệu, tạo bảng bao gồm khoá chính, khoá ngoại. (0.25 điểm)
   2. Thêm dữ liệu cho các bảng theo đúng trình tự bên 1 trước bên nhiều sau. Mỗi bảng ít nhất 5 dòng dữ liệu. (0.25 điểm).

CREATE DATABASE QUANLI\_SANBONG

GO

USE QUANLI\_SANBONG

GO

CREATE TABLE SANBONG(

MASANBONG CHAR(15),

TENSANBONG NVARCHAR(200),

LOAISAN NVARCHAR(50),

PRIMARY KEY(MASANBONG)

)

CREATE TABLE DICHVU(

MADICHVU CHAR(15),

TENDICHVU NVARCHAR(200),

SOLUONG INT,

DONGIA MONEY,

PRIMARY KEY(MADICHVU)

)

CREATE TABLE GIATIENTHUESAN(

MAGIATIEN CHAR(15),

TUGIO TIME,

DENGIO TIME,

SOTIEN MONEY,

PRIMARY KEY(MAGIATIEN)

)

CREATE TABLE THUESAN(

MATHUESAN CHAR(15),

DIENTHOAIKH NVARCHAR(20),

MASANBONG CHAR(15),

NGAYTHUESAN DATE,

GIOBATDAU TIME,

GIOKETTHUC TIME,

TONGTIEN MONEY,

PRIMARY KEY(MATHUESAN) ,

FOREIGN KEY (MASANBONG) REFERENCES SANBONG(MASANBONG)

)

CREATE TABLE CHITIETTHUESAN(

MATHUESAN CHAR(15),

MADICHVU CHAR(15),

SOLUONG INT,

DONGIA MONEY,

PRIMARY KEY (MATHUESAN, MADICHVU),

FOREIGN KEY (MATHUESAN) REFERENCES THUESAN(MATHUESAN),

FOREIGN KEY (MADICHVU) REFERENCES DICHVU(MADICHVU)

)

--1. Viết các câu lệnh SQL để thực hiện các thao tác sao: (0.5 điểm)

--a. Tạo Cơ sở dữ liệu, tạo bảng bao gồm khoá chính, khoá ngoại. (0.25 điểm)

--b. Thêm dữ liệu cho các bảng theo đúng trình tự bên 1 trước bên nhiều sau. Mỗi bảng ít nhất 5 dòng dữ liệu. (0.25 điểm).

INSERT INTO GIATIENTHUESAN VALUES('GIATIEN001','6:00', '8:00',200000)

INSERT INTO GIATIENTHUESAN VALUES('GIATIEN002','8:00', '10:00',150000)

INSERT INTO GIATIENTHUESAN VALUES('GIATIEN003','10:00', '14:00',100000)

INSERT INTO GIATIENTHUESAN VALUES('GIATIEN004','14:00', '17:00',150000)

INSERT INTO GIATIENTHUESAN VALUES('GIATIEN005','17:00', '22:00',200000)

INSERT INTO SANBONG VALUES('SANBONG001', N'SÂN BÓNG A', N'SÂN 5')

INSERT INTO SANBONG VALUES('SANBONG002', N'SÂN BÓNG B', N'SÂN 5')

INSERT INTO SANBONG VALUES('SANBONG003', N'SÂN BÓNG C', N'SÂN 5')

INSERT INTO SANBONG VALUES('SANBONG004', N'SÂN BÓNG D', N'SÂN 7')

INSERT INTO SANBONG VALUES('SANBONG005', N'SÂN BÓNG E', N'SÂN 11')

INSERT INTO DICHVU VALUES('DICHVU001', N'NƯỚC UỐNG', 100, 10000)

INSERT INTO DICHVU VALUES('DICHVU002', N'ĐỒ ĂN', 100, 30000)

INSERT INTO DICHVU VALUES('DICHVU003', N'GIÀY THI ĐẤU', 100, 100000)

INSERT INTO DICHVU VALUES('DICHVU004', N'ÁO PHÔNG CHIA ĐỘI', 100, 10000)

INSERT INTO DICHVU VALUES('DICHVU005', N'GIỮ XE', 100, 10000)

SET DATEFORMAT DMY

GO

INSERT INTO THUESAN VALUES('20012022001', '0343XXXXXX', 'SANBONG002', '20-01-2022', '17:00', '19:00',400000)

INSERT INTO THUESAN VALUES('20062022001', '0342XXXXXX', 'SANBONG001', '20-06-2022', '16:00', '17:00',200000)

INSERT INTO THUESAN VALUES('20012022002', '0344XXXXXX', 'SANBONG001', '20-01-2022', '18:00', '19:00',150000)

INSERT INTO THUESAN VALUES('26052022001', '0340XXXXXX', 'SANBONG001', '26-05-2022', '6:00', '7:00',200000)

INSERT INTO THUESAN VALUES('26052022002', '0341XXXXXX', 'SANBONG001', '26-05-2022', '7:00', '8:00',200000)

SELECT \* FROM THUESAN

DELETE FROM THUESAN

INSERT INTO CHITIETTHUESAN VALUES('20012022001', 'DICHVU001', 1, 10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022001', 'DICHVU005', 12, 10000)

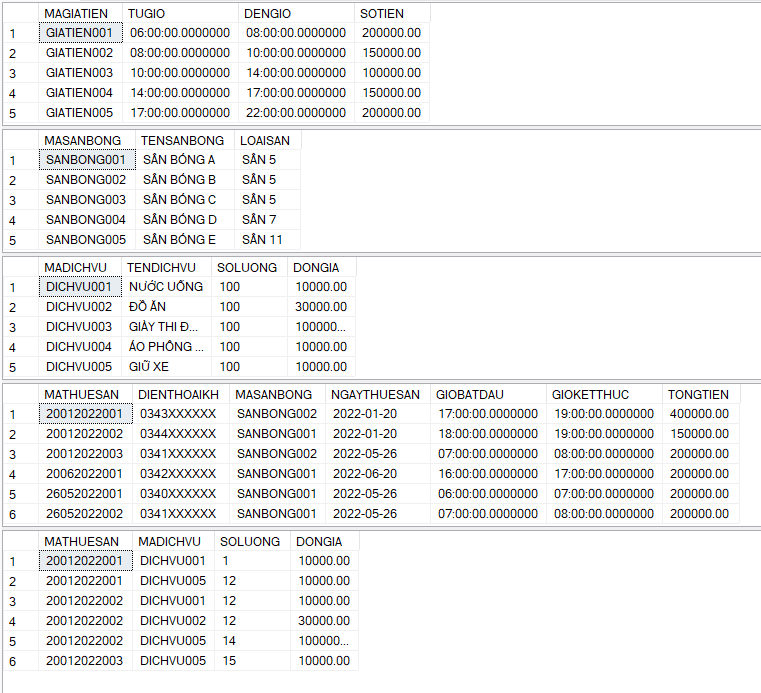
INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU005', 14, 100000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU002', 12,30000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU001', 12,10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022003', 'DICHVU005', 15, 10000)

Kết quả của code:



1. Viết các hàm sau: (1.0 điểm)
2. Viết hàm phát sinh mã số tự động cho bảng thuê sân theo nguyên tắc: Ngày thuê sân + số thứ tự gồm 3 chữ số tăng dần. Ví dụ: Ngày thuê sân là ngày 20/06/2022 và hiện đã có lần thuê sân thứ 3 (Bảng Thuê sân đang có mã số là: “20062022003” thì mã số tự động tiếp theo sẽ là “20062022004” (0.5 điểm)
3. Viết hàm trả về số tiền thuê sân khi biết được giờ vào sân và giờ ra khỏi sân (0.5 điểm)

--2. Viết các hàm sau: (1.0 điểm)

--a. Viết hàm phát sinh mã số tự động cho bảng thuê sân theo nguyên tắc:

--Ngày thuê sân + số thứ tự gồm 3 chữ số tăng dần.

--Ví dụ: Ngày thuê sân là ngày 20/06/2022 và hiện đã có lần thuê sân thứ 3

--(Bảng Thuê sân đang có mã số là: “20062022003” thì

--mã số tự động tiếp theo sẽ là “20062022004” (0.5 điểm)

SET DATEFORMAT DMY

GO

CREATE OR ALTER FUNCTION AUTO\_ID(@NGAYTHUE SMALLDATETIME)

RETURNS CHAR(15)

AS

BEGIN

DECLARE @ID VARCHAR(15)

IF(SELECT COUNT(MATHUESAN)

FROM THUESAN WHERE NGAYTHUESAN = @NGAYTHUE) = 0

SET @ID = '0'

ELSE

SELECT @ID = (SELECT COUNT(MATHUESAN) FROM THUESAN WHERE NGAYTHUESAN = @NGAYTHUE)

SELECT @ID = CASE

WHEN @ID >=0 AND @ID < 9 THEN REPLACE(CONVERT(VARCHAR,@NGAYTHUE,104),'.','') +'00'+ CONVERT(CHAR,CONVERT(INT, @ID) +1)

WHEN @ID >=9 AND @ID < 99 THEN REPLACE(CONVERT(VARCHAR,@NGAYTHUE,104),'.','') +'0'+ CONVERT(CHAR,CONVERT(INT, @ID) +1)

WHEN @ID >= 99 THEN REPLACE(CONVERT(VARCHAR,@NGAYTHUE,104),'.','') + CONVERT(CHAR,CONVERT(INT, @ID) +1)

END

RETURN CAST(@ID AS CHAR)

END

GO

SELECT \* FROM THUESAN

SET DATEFORMAT DMY

PRINT dbo.AUTO\_ID('26/05/2022')

INSERT INTO THUESAN VALUES(dbo.AUTO\_ID('26-05-2022'), '0341XXXXXX', 'SANBONG002', '26-05-2022', '8:00', '10:00',null)

--b. Viết hàm trả về số tiền thuê sân khi biết được giờ vào sân và giờ ra khỏi sân (0.5 điểm)

--7h - 15h

GO

CREATE OR ALTER FUNCTION TIENTHUESAN(@GIOVAO TIME, @GIORA TIME)

RETURNS MONEY

AS

BEGIN

DECLARE @TIENTHUE FLOAT

IF EXISTS(SELECT 1 FROM GIATIENTHUESAN WHERE @GIOVAO >= TUGIO AND @GIORA <= DENGIO)

BEGIN

SELECT @TIENTHUE = SOTIEN\*(CAST(DATEDIFF(MINUTE,@GIOVAO,@GIORA) AS FLOAT))/60

FROM GIATIENTHUESAN

WHERE @GIOVAO >= TUGIO AND @GIORA <= DENGIO

END

ELSE

BEGIN

--tính trong khoảng giờ đầu tiên

SELECT @TIENTHUE = SOTIEN\*(CAST(DATEDIFF(MINUTE,@GIOVAO,DENGIO) AS FLOAT))/60

FROM GIATIENTHUESAN

WHERE @GIOVAO >= TUGIO AND @GIOVAO <= DENGIO

--tính trong khoảng giờ cuối cùng

SELECT @TIENTHUE = @TIENTHUE + SOTIEN\*(CAST(DATEDIFF(MINUTE,TUGIO,@GIORA) AS FLOAT))/60

FROM GIATIENTHUESAN

WHERE @GIORA >= TUGIO AND @GIORA <= DENGIO

--tính khoảng giữa

SELECT @TIENTHUE = @TIENTHUE + SOTIEN\*(DATEDIFF(MINUTE,TUGIO,DENGIO)/60 )

FROM GIATIENTHUESAN

WHERE (@GIOVAO < TUGIO AND @GIORA > DENGIO)

END

RETURN @TIENTHUE

END

GO

PRINT dbo.TIENTHUESAN('17:00', '20:00') --900000

UPDATE THUESAN

SET TONGTIEN = dbo.TIENTHUESAN(GIOBATDAU,GIOKETTHUC)

SELECT \* FROM GIATIENTHUESAN

SELECT \* FROM THUESAN

Kết quả code:

**a.**

26052022004

**b.**

600000.00

1. Viết các thủ tục sau: (0.5 điểm)
   1. Thêm vào bảng Giá tiền thuê sân có kiểm tra khoá chính, giờ vào phải nhỏ hơn giờ ra. (0.25 điểm)
   2. Cập nhật đơn giá cho dịch vụ thuê áo thêm 10.000 (0.25 điểm).

--3. Viết các thủ tục sau: (0.5 điểm)

--a. Thêm vào bảng Giá tiền thuê sân có kiểm tra khoá chính, giờ vào phải nhỏ hơn giờ ra. (0.25 điểm)

GO

CREATE OR ALTER PROC CAU3A\_THEMGIATIEN @MAGIATIEN CHAR(15), @TUGIO TIME, @DENGIO TIME, @SOTIEN MONEY

AS

BEGIN

IF (@MAGIATIEN IS NULL OR EXISTS (SELECT MAGIATIEN FROM GIATIENTHUESAN WHERE MAGIATIEN = @MAGIATIEN))

PRINT N'KHÓA CHÌNH BỊ RỖNG HOẶC TRÙNG'

ELSE IF (@TUGIO >= @DENGIO)

PRINT N'GIỜ VÀO PHẢI NHỎ HƠN GIỜ RA'

ELSE

INSERT INTO GIATIENTHUESAN VALUES(@MAGIATIEN, @TUGIO, @DENGIO, @SOTIEN)

END

GO

EXEC CAU3A\_THEMGIATIEN'GIATIEN005', '7:00','7:00', 200000 --KHÓA CHÌNH BỊ RỖNG HOẶC TRÙNG

DELETE FROM GIATIENTHUESAN WHERE MAGIATIEN = 'GIATIEN006'

EXEC CAU3A\_THEMGIATIEN'GIATIEN006', '7:00','7:00', 200000 --GIỜ VÀO PHẢI NHỎ HƠN GIỜ RA

EXEC CAU3A\_THEMGIATIEN'GIATIEN006', '7:00','8:00', 200000 --THÀNH CÔNG

DELETE FROM GIATIENTHUESAN WHERE MAGIATIEN = 'GIATIEN006'

--b. Cập nhật đơn giá cho dịch vụ thuê áo thêm 10.000 (0.25 điểm).

GO

CREATE OR ALTER PROC CAU3B\_UPDATE\_DONGIA

AS

BEGIN

UPDATE DICHVU

SET DONGIA = DONGIA + 10000

WHERE MADICHVU = 'DICHVU004'

END

GO

SELECT \* FROM DICHVU

EXEC CAU3B\_UPDATE\_DONGIA

SELECT \* FROM DICHVU

Kết quả của code:

**3a.**

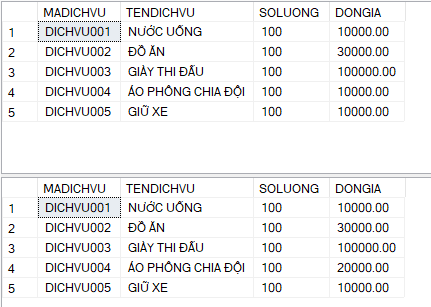
EXEC CAU3A\_THEMGIATIEN'GIATIEN005', '7:00','7:00', 200000 --KHÓA CHÌNH BỊ RỖNG HOẶC TRÙNG

DELETE FROM GIATIENTHUESAN WHERE MAGIATIEN = 'GIATIEN006'

EXEC CAU3A\_THEMGIATIEN'GIATIEN006', '7:00','7:00', 200000 --GIỜ VÀO PHẢI NHỎ HƠN GIỜ RA

EXEC CAU3A\_THEMGIATIEN'GIATIEN006', '7:00','8:00', 200000 --THÀNH CÔNG

**3b. Trước và sau khi dịch vụ áo đấu tăng thêm 10 nghìn:**

****

1. Viết 1 trigger sau: (1 điểm)

Khi thêm, xoá, sửa dữ liệu trong bảng Chi tiết thuê sân:

* Cập nhật lại Tổng tiền trong bảng thuê sân. Trong đó, tổng tiền = tiền thuê sân + tiền sử dụng dịch vụ của lần thuê sân đó (0.5 điểm)
* Đơn giá trong bảng Chi tiết thuê sân phải bằng với đơn giá trong bảng dịch vụ (0.5 điểm)

--4. Viết 1 trigger sau: (1 điểm)

--Khi thêm, xoá, sửa dữ liệu trong bảng Chi tiết thuê sân:

--- Cập nhật lại Tổng tiền trong bảng thuê sân. Trong đó, tổng tiền = tiền thuê sân + tiền sử dụng dịch vụ của lần thuê sân đó (0.5 điểm)

--- Đơn giá trong bảng Chi tiết thuê sân phải bằng với đơn giá trong bảng dịch vụ (0.5 điểm)s

GO

CREATE OR ALTER TRIGGER TRIG\_CAU4

ON CHITIETTHUESAN

FOR INSERT,UPDATE, DELETE

AS

DECLARE @GIOVAO TIME, @GIORA TIME

IF EXISTS (SELECT 1 FROM inserted)

BEGIN

--INSERT

DECLARE @MATHUE\_INSERTED CHAR(15)

SELECT @MATHUE\_INSERTED = MATHUESAN FROM INSERTED

SELECT @GIOVAO = GIOBATDAU, @GIORA = GIOKETTHUC FROM THUESAN WHERE MATHUESAN = @MATHUE\_INSERTED

UPDATE THUESAN

SET TONGTIEN = DBO.TIENTHUESAN(@GIOVAO, @GIORA) + (SELECT SUM(CT.SOLUONG \* CT.DONGIA)

FROM CHITIETTHUESAN CT

WHERE CT.MATHUESAN = @MATHUE\_INSERTED)

WHERE MATHUESAN = @MATHUE\_INSERTED

--KIỂM TRA DƠN GIÁ

IF(SELECT DONGIA FROM inserted) != (SELECT DICHVU.DONGIA FROM DICHVU,inserted WHERE inserted.MADICHVU = DICHVU.MADICHVU)

BEGIN

PRINT(N'Đơn giá trong bảng Chi tiết thuê sân phải bằng với đơn giá trong bảng dịch vụ')

ROLLBACK TRAN

END

END

ELSE IF EXISTS(SELECT 1 FROM deleted)

IF NOT EXISTS(SELECT 1 FROM deleted)

BEGIN

RETURN

END

ELSE

BEGIN

--DELETE dựa trên mã thuê sân và mã dịch vụ

DECLARE @TIEN\_DELETE MONEY

SET @TIEN\_DELETE = (SELECT SUM(SOLUONG \* DONGIA)

FROM deleted

GROUP BY deleted.MATHUESAN)

UPDATE THUESAN

SET TONGTIEN = TONGTIEN - @TIEN\_DELETE

--WHERE ThueSan.MATHUESAN = deleted.MATHUESAN)

--FROM THUESAN, deleted

WHERE THUESAN.MATHUESAN = (select MATHUESAN from deleted)

END

ELSE

BEGIN

--UPDATE

---UPDATE DUA TREN MADICHVU VA MATHUESAN KHI XET WHERE

IF UPDATE(SOLUONG)

BEGIN

UPDATE THUESAN

SET TONGTIEN = dbo.TIENTHUESAN(GIOBATDAU,GIOKETTHUC) + (SELECT SUM(CT.SOLUONG \* CT.DONGIA)

FROM DICHVU DV,CHITIETTHUESAN CT

WHERE DV.MADICHVU = CT.MADICHVU AND CT.MATHUESAN = THUESAN.MATHUESAN)

END

--KIỂM TRA DƠN GIÁ

IF UPDATE(DONGIA)

BEGIN

PRINT(N'Đơn giá không được cập nhật')

ROLLBACK TRAN

END

END

GO

--test

drop trigger TRIG\_CAU4

SELECT \* FROM THUESAN

SELECT \* FROM CHITIETTHUESAN

DELETE FROM THUESAN

DELETE FROM CHITIETTHUESAN

SET DATEFORMAT DMY

INSERT INTO THUESAN VALUES('20012022001', '0343XXXXXX', 'SANBONG002', '20-01-2022', '17:00', '19:00',dbo.TIENTHUESAN('17:00','19:00'))

INSERT INTO THUESAN VALUES('20012022002', '0344XXXXXX', 'SANBONG001', '20-01-2022', '18:00', '19:00',dbo.TIENTHUESAN('18:00','19:00'))

INSERT INTO THUESAN VALUES('20012022003', '0344XXXXXX', 'SANBONG003', '20-01-2022', '18:00', '19:00',dbo.TIENTHUESAN('18:00','19:00'))

INSERT INTO CHITIETTHUESAN VALUES('20012022001', 'DICHVU001', 1, 1000000)

INSERT INTO CHITIETTHUESAN VALUES('20012022001', 'DICHVU005', 12, 10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU005', 14, 10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU002', 12, 30000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU001', 12, 10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022003', 'DICHVU005', 15, 10000)

DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022001'

DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU001'

DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU002'

DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU005'

DELETE FROM CHITIETTHUESAN WHERE MATHUESAN = '20012022003'

SELECT \* FROM THUESAN

SELECT \* FROM CHITIETTHUESAN

UPDATE CHITIETTHUESAN

SET SOLUONG = 10

WHERE MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU002'

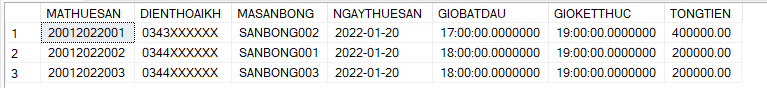
UPDATE CHITIETTHUESAN

SET DONGIA = 30000

WHERE MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU002'

Kết quả của code:

Thuê sân lúc đầu:



Sau khi add thêm chi tiết thuê sân:

* Với

INSERT INTO CHITIETTHUESAN VALUES('20012022001', 'DICHVU001', 1, 1000000)

Đơn giá trong bảng Chi tiết thuê sân phải bằng với đơn giá trong bảng dịch vụ

* Với

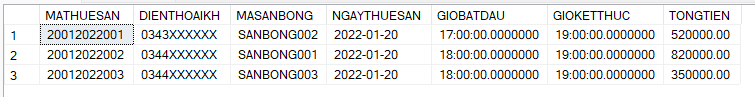
INSERT INTO CHITIETTHUESAN VALUES('20012022001', 'DICHVU005', 12, 10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU005', 14, 10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU002', 12, 30000)

INSERT INTO CHITIETTHUESAN VALUES('20012022002', 'DICHVU001', 12, 10000)

INSERT INTO CHITIETTHUESAN VALUES('20012022003', 'DICHVU005', 15, 10000)



* Với

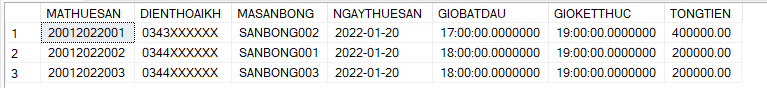
DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022001'

DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU001'

DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU002'

DELETE FROM CHITIETTHUESAN where MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU005'

DELETE FROM CHITIETTHUESAN WHERE MATHUESAN = '20012022003'

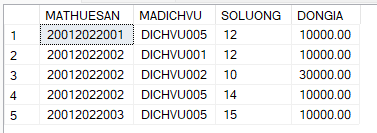


* Với

UPDATE CHITIETTHUESAN

SET SOLUONG = 10

WHERE MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU002'



* Với:

UPDATE CHITIETTHUESAN

SET DONGIA = 20000

WHERE MATHUESAN = '20012022002' AND MADICHVU = 'DICHVU002'

Đơn giá trong bảng Chi tiết thuê sân phải bằng với đơn giá trong bảng dịch vụ

* Do trong bảng dịch vụ thì dịch vụ 002 là 30000

Phần 2: (2 điểm)

Đề:

Cho lược đồ quan hệ R(U, F), U = MNOPQRS

F = {S🡪MR; NS🡪QM; PQ🡪RS; MO🡪NR; N🡪R}

1. Tìm tất cả các khóa (khóa dự tuyển) của lược đồ quan hệ (0.5 điểm)
2. Tìm phủ tối thiểu của tập phụ thuộc hàm (0.5 điểm)
3. Xác định dạng chuẩn cao nhất của lược đồ quan hệ (0.5 điểm)
4. Nếu lược đồ quan hệ chưa đạt dạng chuẩn BC, đưa lược đồ về dạng chuẩn BC (0.5 điểm)

Câu Trả lời:

a) Tìm tất cả các khóa của lược đồ quan hệ:

L = {MNOPQS}

R = {MNQRS}

Tập nguồn: N = Q\R = {OP}

Tập đích: D = R\L = {R}

Tập lững: L = {MNQS}

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| L |  | |  | |  |  | | Xi | | (Xi)F+ | | Super key | |
| M | | N | | Q | S | OP |  | |  | |  | |
| 0 | | 0 | | 0 | 0 | 1 | OP | | OP | |  | |
| 1 | | 0 | | 0 | 0 | 1 | MOP | | MOPNR | |  | |
| 0 | | 1 | | 0 | 0 | 1 | NOP | | NOPR | |  | |
| 1 | | 1 | | 0 | 0 | 1 | MNOP | | MNOPR | |  | |
| 0 | | 0 | | 1 | 0 | 1 | QOP | | QOPRSMN | | Q+ | |
| 1 | | 0 | | 1 | 0 | 1 | MQOP | | MQOPNRS | | Q+ | |
| 0 | | 1 | | 1 | 0 | 1 | NQOP | | NQOPRSM | | Q+ | |
| 1 | | 1 | | 1 | 0 | 1 | MNQOP | | MNQOPSR | | Q+ | |
| 0 | | 0 | | 0 | 1 | 1 | SOP | | SOPMRNQ | | Q+ | |
| 1 | | 0 | | 0 | 1 | 1 | MSOP | | MSOPRNQ | | Q+ | |
| 0 | | 1 | | 0 | 1 | 1 | NSOP | | NSOPMRQ | | Q+ | |
| 1 | | 1 | | 0 | 1 | 1 | MNSOP | | MNSOPRQ | | Q+ | |
| 0 | | 0 | | 1 | 1 | 1 | QSOP | | QSOPMRN | | Q+ | |
| 1 | | 0 | | 1 | 1 | 1 | MQSOP | | MQSOPNQ | | Q+ | |
| 0 | | 1 | | 1 | 1 | 1 | NQSOP | | NQSOPMR | | Q+ | |
| 1 | | 1 | | 1 | 1 | 1 | MNQSOP | | MNQSOPR | | Q+ | |

**Vậy ta có 2 khóa** {QOP VÀ SOP} và các siêu khóa {QOP, MQOP, NQOP, MNQOP, SOP, MSOP, NSOP, MNSOP, QSOP, MQSOP, NQSOP, MNQSOP}

b. Tìm phủ tối thiểu của tập phụ thuộc hàm (0.5 điểm)

Bước 1: Phân rã vế phải của lược đồ quan hệ ta có:

F = {S🡪MR;

NS🡪QM;

PQ🡪RS;

MO🡪NR;

N🡪R}

* F1 = {S 🡪 M, S 🡪R, NS 🡪 Q, NS 🡪 M, PQ 🡪 R, PQ 🡪 S, MO 🡪 N, MO 🡪 R, N 🡪 R}

Bước 2: Loại bỏ các thuộc tính dư thừa ở vế trái:

* Xét NS 🡪 Q, không dư thừa:
  + NF+ = NR không chứa Q.
  + SF+ = SMR không chứa Q.
* Xét NS 🡪 M, N **dư thừa**:
  + NF+ = NR không chứa M.
  + SF+ = SMR chứa M.
* Xét PQ 🡪 R, không dư thừa:
  + PF+ = P không chứa R
  + QF+ = Q không chứa R
* Xét PQ 🡪 S, không dư thừa:
  + PF+ = P không chứa S
  + QF+ = Q không chứa S
* Xét MO 🡪 N, không dư thừa:
  + MF+ = M không chứa N.
  + OF+ = O không chứa N.
* F2 = {S 🡪 M, S 🡪R, NS 🡪 Q, PQ 🡪 R, PQ 🡪 S, MO 🡪 N, MO 🡪 R, N 🡪 R}

Bước 3: Loại bỏ phụ thuộc hàm dư thừa trên F2:

* Xét S 🡪 M, không dư thừa:
  + SF+(F2-{S🡪M}) = SR không chứa M
* Xét S 🡪 R, không dư thừa vì:
  + S+(F2-{S🡪R}) = SM không chứa R
* Xét NS 🡪 Q, không dư thừa:
  + NS+(F2-{NS🡪Q}) = NSMR không chứa Q
* Xét PQ 🡪 R, **dư thừa:**
  + PQ+(F2-{PQ🡪R}) = PQSMR chứa R (1)
* Xét PQ 🡪 S, không dư thừa:
  + PQ+(F2-(1)-{PQ🡪S}) = PQR không chứa S
* Xét MO 🡪 N, không dư thừa:
  + MO+(F2-(1)-{MO🡪N}) = MOR không chứa N
* Xét MO 🡪 R, **dư thừa:**
  + MO+(F2-(1)-{MO🡪R}) = MONR chứa R (2)
* Xét N 🡪 R, không dư thừa:
  + N+(F2-(1)-(2)-{N🡪R}) = N không chứa R
* F3 = {S 🡪 M, S 🡪R, NS 🡪 Q, PQ 🡪 S, MO 🡪 N, N 🡪 R} là phủ tối thiểu

Vậy phủ tối thiểu của lược đồ là: F3 = {S 🡪 M, S 🡪R, NS 🡪 Q, PQ 🡪 S, MO 🡪 N, N 🡪 R}

* 1. Xác định dạng chuẩn cao nhất của lược đồ quan hệ (0.5 điểm)
* Tập thuộc tính khóa {QSOP}
* Tập thuộc tính không khóa: {M,N,R}

Xét dạng 2NF, ta có:

* Xét QSOP 🡪 M, thuộc tính không khóa không phụ thuộc đầy đủ vào thuộc tính khóa:
  + Bao đóng Q = Q
  + Bao đóng S = SMR chứa M
  + Bao đóng O = O
  + Bao đóng P = P
* Lược đồ không thuộc dạng 2NF

Vậy dạng chuẩn cao nhất của lược đồ quan hệ là 1NF

* 1. Nếu lược đồ quan hệ chưa đạt dạng chuẩn BC, đưa lược đồ về dạng chuẩn BC (0.5 điểm)
* Lược đồ quan hệ chưa đạt dạng chuẩn BC do dạng chuẩn cao nhất của lược đồ quan hệ là 1NF

Ta có U = {MNOPQRS}

Phủ tối thiểu: F3 = {S 🡪 M, S 🡪R, NS 🡪 Q, PQ 🡪 S, MO 🡪 N, N 🡪 R}

Bước 1: Kiểm tra tất cả các thuộc tính trong U có tồn tại trong vế phải và vế trài của phụ thuộc hàm phủ tối thiểu hay không?

Tất cả thuộc tính trong U đều có trong VT, VP của PTH hàm phủ tối thiểu

Bước 2: gộp các phụ thuộc hàm có cùng vế trái, ta có:

* {S 🡪 MR, NS 🡪 Q, PQ 🡪 S, MO 🡪 N, N🡪 R}

Bước 3: phân rã:

* R1(SMR); K1 = S, F1 = {S 🡪 MR} đạt dạng 1NF, 2NF, 3NF, BC
* R2(NSQ); K2 = NS, F2 = {NS 🡪 Q} đạt dạng 1NF, 2NF, 3NF, BC
* R3(PQS); K3 = PQ, F3 = {PQ 🡪 S} đạt dạng 1NF, 2NF, 3NF, BC
* R4(MON); K4 = MO, F4 = {MO 🡪 N} đạt dạng 1NF, 2NF, 3NF, BC
* R5(NR); K5 = R, F5 = {N 🡪 R} đạt dạng 1NF, 2NF, 3NF, BC

Vậy R1, R2, R3, R4, R5 thỏa dạng BCNF

Phần 3: (3 điểm)

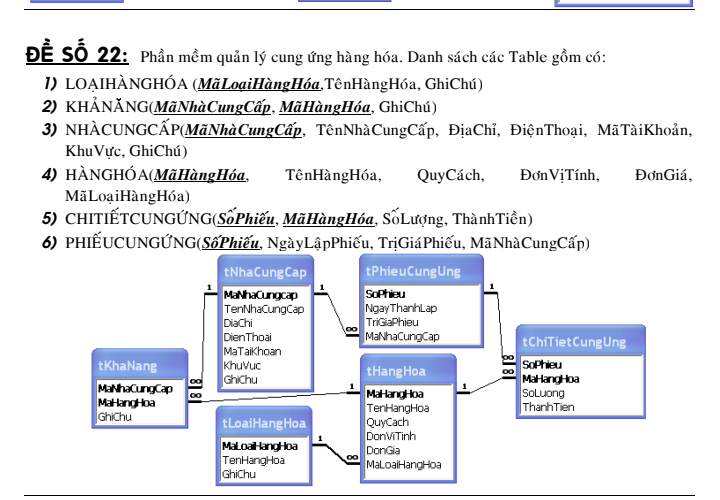
Đề:

Sinh viên tự viết đặc tả về một nghiệp vụ quản lý nào đó. Đặc tả phải thể hiện được có các mối quan hệ 1-1, 1-n, n-n, kế thừa, thực thể mạnh, thực thể yếu, … Sau đó, sinh viên vẽ mô hình ERD cho đặc tả đó, chuyển mô hình ERD sang mô hình quan hệ.

1. Viết đặc tả (1.5 điểm)
2. Vẽ ERD (0.75 điểm)
3. Chuyển ERD sang mô hình quan hệ (0.75 điểm)

Câu trả lời:

1. Viết đặc tả:



Đặt tả (có chỉnh sửa để đầy đủ các yêu cầu đề bài):

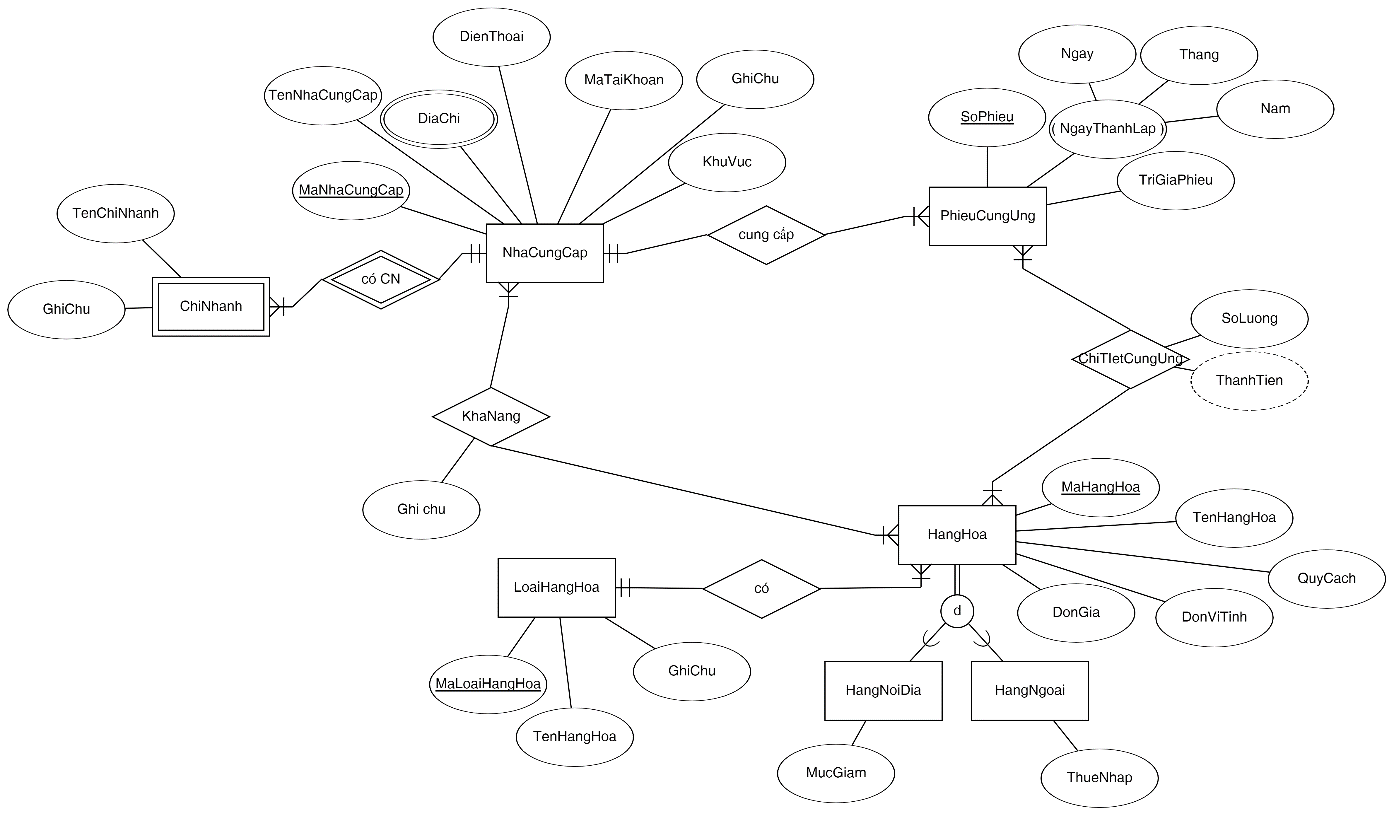
Một chuỗi cung ứng hàng hóa chuyên cung ứng hàng hóa như thực phẩm, đồ dùng gia đình, vật dụng nói chung. . Hàng hóa gồm hàng hóa nội địa và hàng hóa ngoại. **Hàng hóa** sẽ gồm các thông tin là Mã hàng hóa, tên hàng hóa, quy cách, đơn vị tính, đơn giá, mã loại hàng hoá. ***Hàng hóa nội địa*** sẽ có thêm thông tin mức giảm, ***hàng hóa ngoại*** sẽ lưu thêm thông tin thuế. Một hàng hóa thuộc một loại hàng hóa mà một loại hàng hóa có nhiều hàng hóa**. Mỗi loại hàng hóa** sẽ có thông tin gồm mã loại hàng hóa, tên hàng hóa và ghi chú.

Mỗi khi có một nhà cung cấp cung cấp hàng hóa hệ thống sẽ tạo ra phiếu cung ứng và mỗi nhà cũng cấp sẽ có nhiều phiếu cung ứng. Và thông tin của **nhà cung cấp** bao gồm Mã nhà cung cấp, Tên nhà cung cấp, địa chỉ, điện thoại, mã tài khoản, khu vực, ghi chú; một nhà cung cấp có thể định vị ở nhiều địa chỉ khác nhau. Bên cạnh đó các nhà cung cấp có thêm nhân thân. Thông tin **Phiếu cung ứng** sẽ chứa số phiếu, ngày thành lập (gồm ngày, tháng, năm), trị giá phiếu, mã nhà cung cấp. Để quản lí độ tin cậy cho nhà cung cấp khách hàng cần biết thêm các chi nhánh của nhà cung cấp, thông tin **chi nhánh** gồm tên chi nhánh, ghi chú.

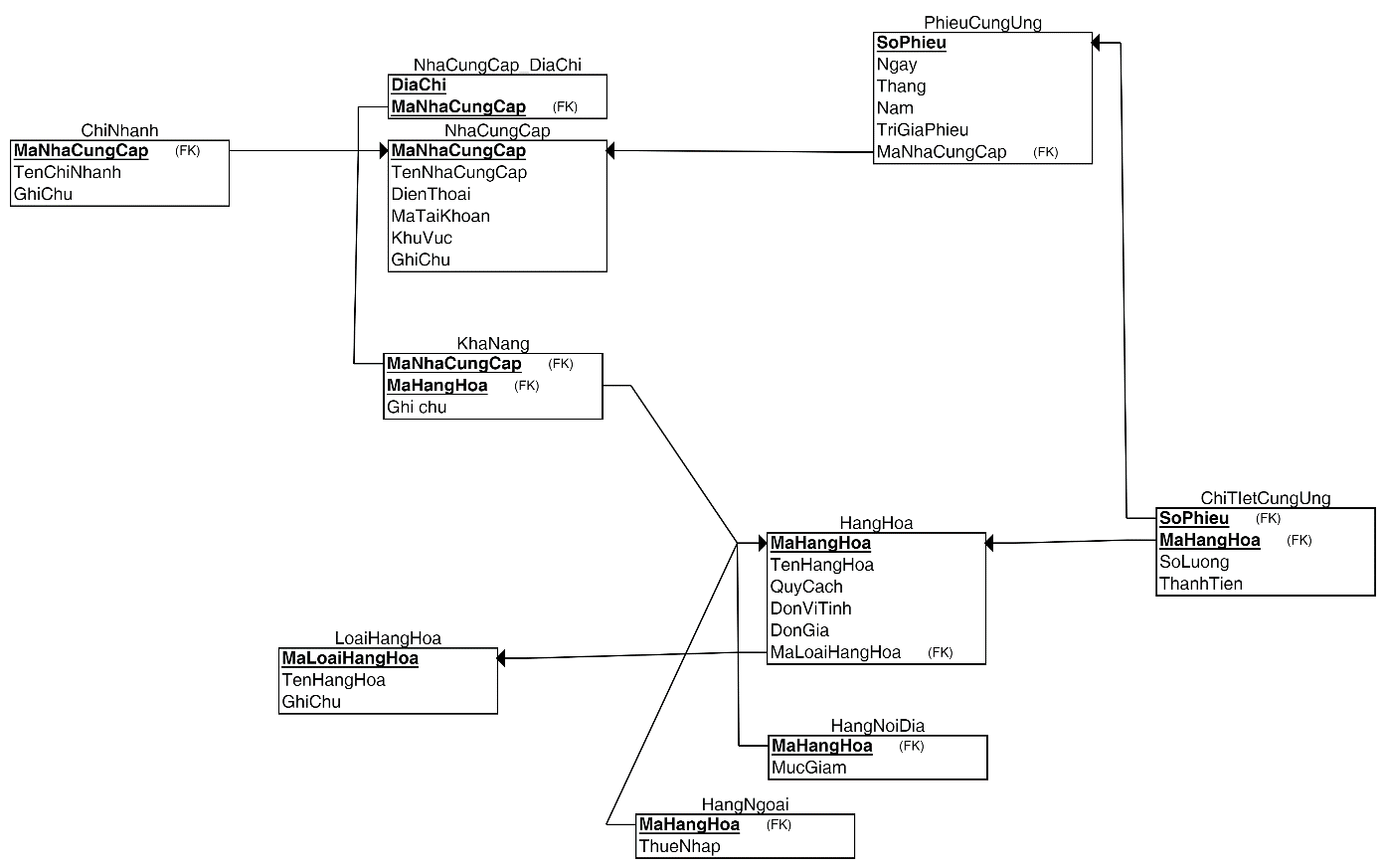
Mỗi Phiếu cung ứng chứa một hoặc nhiều hàng hóa, mỗi hàng hóa nằm trong một hoặc nhiều phiếu cung ứng**,** khi phiếu cung ứng cung ứng hàng hóa hệ thống sẽ ghi lại thông tin số lượng và thành tiền với thành tiền được tính bằng đơn giá của hàng hóa nhân với số lương. Thông tin hệ thống ghi nhận (**chi tiết cung ứng**) gồm số phiếu, mã hàng hóa, số lượng và thành tiền

Mỗi nhà cung cấp có cung cấp một hoặc nhiều hàng hóa và hàng hóa có ít nhất một nhà cung cấp hệ thống sẽ lưu lại ghi chú của nhà cung cấp. Thông tin ghi nhận (**khả năng**) gồm Mã nhà cung cấp, Mã hàng hóa, ghi chú.

1. Vẽ ERD:



1. Mô hình quan hệ:



Phần 4: (2 điểm)

Đề:

Sử dụng ngôn ngữ C/Java/Python để cài đặt các chức năng sau:

1. Cho lược đồ cơ sở dữ liệu (LĐCSDL), cho biết các khoá của lược đồ LĐCSDL (0.5 điểm)
2. Cho LĐCSDL, cho biết phủ tối thiểu của LĐCSDL (0.75 điểm)
3. Cho LĐCSDL, cho biết dạng chuẩn cao nhất của LĐCSDL. Giải thích tại sao. (0.75 điểm)

Câu trả lời (python):

1 Cho lược đồ quan cho biết các khóa của lược đồ quan hệ:

**Ý tưởng code:**

* Đầu tiên thực hiện tìm bao đóng của thuật toán:

Sau đó tiến hành các bước:

**Bước 1 :**

Tìm tập thuộc tính nguồn TN và Tập thuộc tính trung gian TG

**Bước 2 :**

Nếu TG = 0

Thì K(Key) = TN, và kết thúc thuật toán, xuất ra K của tập cơ sở dữ liệu <Q,F>

Ngược lại, nếu TG # 0

Thì qua bước 3

**Bước 3 :**

Tìm tất cả các tập con Xicủa TG

**Bước 4 :**

Tìm Siêu khóa(Si) bằng cách với mọi Xi, nếu (TN U Xi)+= Q thì khi đó Si = TN U Xi

**Bước 5 :**

Tìm Khóa(Ki) bằng cách loại bỏ các siêu khóa không tối thiểu

Với mọi SiSj thuộc S

**Source Code và comment code (được viết bằng python):**

import itertools

# this class will used to store relation: their attr, functionalDepend

class Relation:

    def \_\_init\_\_(*self*):

*self*.attr\_set = set()  # store attr set

*self*.functional\_dependency = dict()  # store functional dependency

    def setAttrSet(*self*, *attribute*):

*self*.attr\_set = *attribute*

    def getAttrSet(*self*):

        return *self*.attr\_set

    def setFunctionalDependency(*self*, *x*):

*self*.functional\_dependency = *x*

    def getFunctionalDependency(*self*):

        return *self*.functional\_dependency

    def findClosureAttributeSet(*self*, *attrX*):

        attrX\_set = set(*attrX*)

        functionalDependencies = *self*.getFunctionalDependency()

        # Left is the side of Left of functional dependency

        Left = list()

        for item in functionalDependencies.keys():

            Left.append(set(item))

        # Right is the side of Right of functional dependency

        Right = list()

        for item in functionalDependencies.values():

            Right.append(set(item))

        # Find Closesure OF X

        closureOfX = attrX\_set

        numIndex = 0

        while True:

            numIndex = 0

            for item in Left:

                if item.issubset(closureOfX):

                    break

                else:

                    numIndex += 1

            if numIndex != len(Left):

                closureOfX = closureOfX.union(Right[numIndex])

                Left.remove(Left[numIndex])

                Right.remove(Right[numIndex])

            else:

                break

        return closureOfX

    def findAllSubsetOfX(*self*, *Xi*):

        AllSubsetOfX = list()

        for i in range(0, len(*Xi*)+1):

            for sub in itertools.combinations(*Xi*, i):

                AllSubsetOfX.append(sub)

        return AllSubsetOfX

    def findAllKey(*self*):

        Relational\_set = *self*.getAttrSet()

        # Left is the side of Left of functional dependency

        Left = list()

        functionalDependencies = *self*.getFunctionalDependency()

        for item in functionalDependencies.keys():

            Left.append(item)

        temp = list()

        for item in Left:

            for attr in item:

                temp.append(attr)

        # Left\_2 will be stored every item in Left

        Left\_2 = set(temp)

        # Right is the side of Right of functional dependency

        Right = list()

        for item in functionalDependencies.values():

            Right.append(set(item))

        temp = list()

        for item in Right:

            for attr in item:

                temp.append(attr)

        # Right\_2 will be stored every item in Right

        Right\_2 = set(temp)

        # SourceSet

        SourceSet = Relational\_set.difference(Right\_2)

        # IntermediateSet

        IntermediateSet = Left\_2.intersection(Right\_2)

        # superKeyList store list of super key set

        superKeyList = list()

        for subset in *self*.findAllSubsetOfX(IntermediateSet):

            if *self*.findClosureAttributeSet(SourceSet.union(subset)) == *self*.getAttrSet():

                superKeyList.append(SourceSet.union(subset))

        # notKey will be store super key and not key

        notKey = list()

        for i in range(0, len(superKeyList) - 1):

            for j in range(i + 1, len(superKeyList)):

                if superKeyList[i].issubset(superKeyList[j]):

                    notKey.append(superKeyList[j])

        # KeyList is solution

        KeyList = list()

        for superKey in superKeyList:

            if superKey not in notKey:

                KeyList.append(superKey)

        return KeyList

    def findAllSuperKey(*self*):

        Relational\_set = *self*.getAttrSet()

        # Left is the side of Left of functional dependency

        Left = list()

        functionalDependencies = *self*.getFunctionalDependency()

        for item in functionalDependencies.keys():

            Left.append(item)

        temp = list()

        for item in Left:

            for attr in item:

                temp.append(attr)

        # Left\_2 will be stored every item in Left

        Left\_2 = set(temp)

        # Right is the side of Right of functional dependency

        Right = list()

        for item in functionalDependencies.values():

            Right.append(set(item))

        temp = list()

        for item in Right:

            for attr in item:

                temp.append(attr)

        # Right\_2 will be stored every item in Right

        Right\_2 = set(temp)

        # SourceSet

        SourceSet = Relational\_set.difference(Right\_2)

        # IntermediateSet

        IntermediateSet = Left\_2.intersection(Right\_2)

        # superKeyList store list of super key set

        superKeyList = list()

        for subset in *self*.findAllSubsetOfX(IntermediateSet):

            if *self*.findClosureAttributeSet(SourceSet.union(subset)) == *self*.getAttrSet():

                superKeyList.append(SourceSet.union(subset))

        return superKeyList

relation = Relation()

relation.setAttrSet({'A', 'B', 'C'})

relation.setFunctionalDependency(

    {"AB": "C", "C": "A"})

print(relation.findAllKey())

print(relation.findAllSuperKey())

#

# relation = Relation()

# relation.setAttrSet({'M', 'N', 'O', 'P', 'Q', 'R', 'S'})

# relation.setFunctionalDependency(

#     {'S': 'MR', 'NS': 'QM', 'PQ': 'RS', 'MO': 'NR', 'N': 'R'})

# print(relation.findAllKey())

# print(relation.findAllSuperKey())

**Kết quả chạy code dựa trên bài mẫu ở phần 2: xuất ra tập khóa và siêu khóa**



* Một ví dụ khác:

relation = Relation()

relation.setAttrSet({'A', 'B', 'C'})

relation.setFunctionalDependency(

    {"AB": "C", "C": "A"})

print(relation.findAllKey())

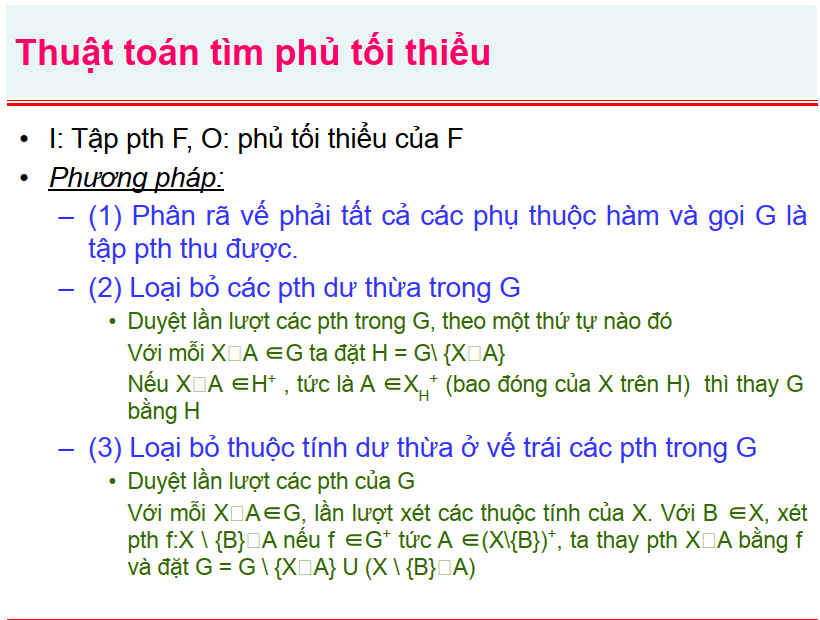
print(relation.findAllSuperKey())

Kết quả:



1. Cho LĐCSDL, cho biết phủ tối thiểu của LĐCSDL (0.75 điểm)

**Ý tưởng code:**



**Source Code:**

class Relation:

    def \_\_init\_\_(*self*):

*self*.attr\_set = set()  # store attr set

*self*.functional\_dependency = dict()  # store functional dependency

    def setAttrSet(*self*, *attribute*):

*self*.attr\_set = *attribute*

    def getAttrSet(*self*):

        return *self*.attr\_set

    def setFunctionalDependency(*self*, *x*):

*self*.functional\_dependency = *x*

    def getFunctionalDependency(*self*):

        return *self*.functional\_dependency

    def findClosureAttributeSet(*self*, *attrX*):

        attrX\_set = set(*attrX*)

        functionalDependencies = *self*.getFunctionalDependency()

        # Left is the side of Left of functional dependency

        Left = list()

        for item in functionalDependencies.keys():

            Left.append(set(item))

        # Right is the side of Right of functional dependency

        Right = list()

        for item in functionalDependencies.values():

            Right.append(set(item))

        # Find Closesure OF X

        closureOfX = attrX\_set

        numIndex = 0

        while True:

            numIndex = 0

            for item in Left:

                if item.issubset(closureOfX):

                    break

                else:

                    numIndex += 1

            if numIndex != len(Left):

                closureOfX = closureOfX.union(Right[numIndex])

                Left.remove(Left[numIndex])

                Right.remove(Right[numIndex])

            else:

                break

        resultString = ""

        for item in closureOfX:

            resultString += str(item)

        return resultString

    def step1(*self*):

        # right-hand decomposition

        dict\_temp = dict()

        dict\_temp = *self*.getFunctionalDependency().copy()

        for key, values in dict\_temp.items():

            dict\_temp[key] = list(c for c in values)

        return dict\_temp

    def resolve(*self*):

        F1\_temp = *self*.step1()

        F2\_temp = *self*.getFunctionalDependency()

        for key, value in F1\_temp.items():  # S, ['M','R']

            list\_temp = value.copy()

            F2\_temp.pop(key)

            for item in value:

                list\_temp.remove(item)

                F2\_temp[key] = list\_temp

                if item in *self*.findClosureAttributeSet(key):

                    continue

                else:

                    F2\_temp[key].append(item)

        return F2\_temp

relation = Relation()

relation.setAttrSet({'M', 'N', 'O', 'P', 'Q', 'R', 'S'})

relation.setFunctionalDependency(

    {'S': 'MR', 'NS': 'QM', 'PQ': 'RS', 'MO': 'NR', 'N': 'R'})

print(relation.resolve())

**Kết quả chạy code:**

Dựa vào ví dụ mấy của bài 2:

relation = Relation()

relation.setAttrSet({'M', 'N', 'O', 'P', 'Q', 'R', 'S'})

relation.setFunctionalDependency(

    {'S': 'MR', 'NS': 'QM', 'PQ': 'RS', 'MO': 'NR', 'N': 'R'})

print(relation.resolve())

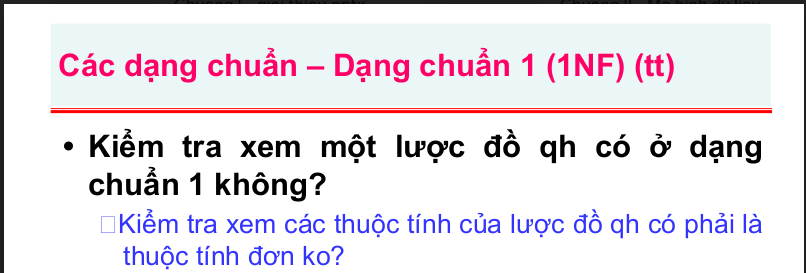
Kết quả:

****

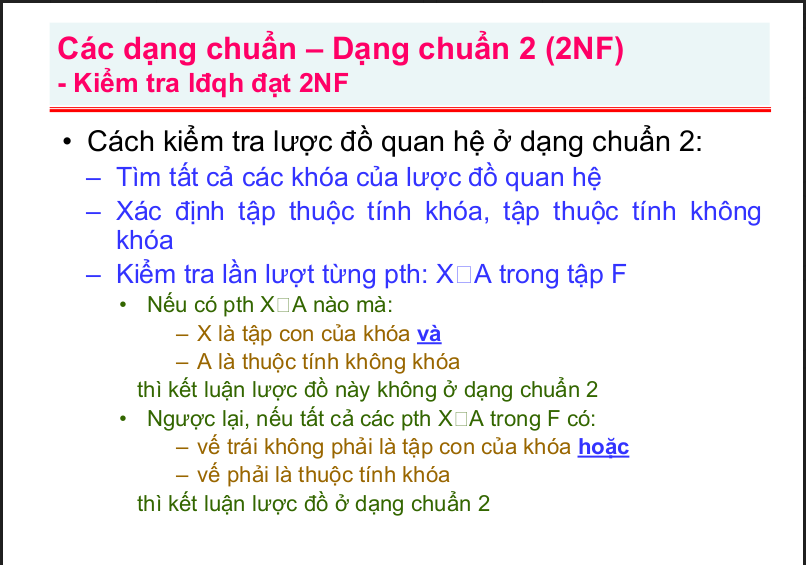
1. Cho LĐCSDL, cho biết dạng chuẩn cao nhất của LĐCSDL. Giải thích tại sao. (0.75 điểm)

**Ý tưởng code:**

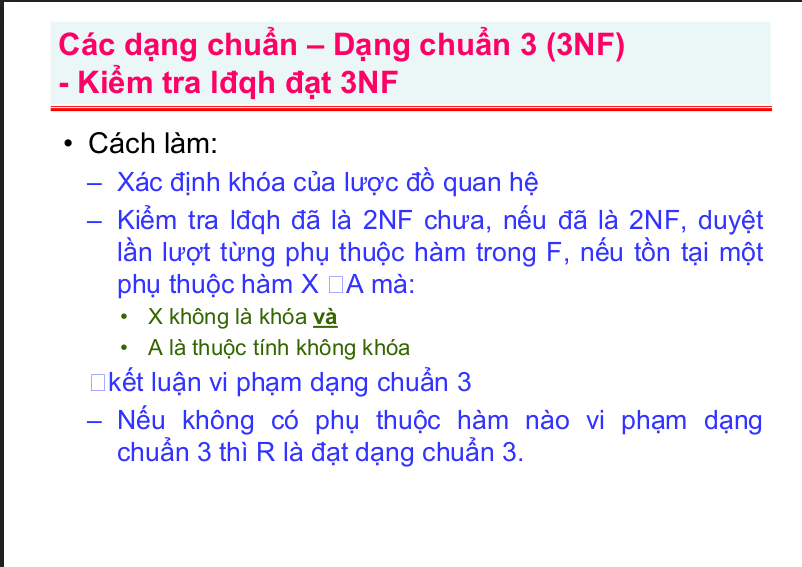
* Dạng chuẩn 1NF:



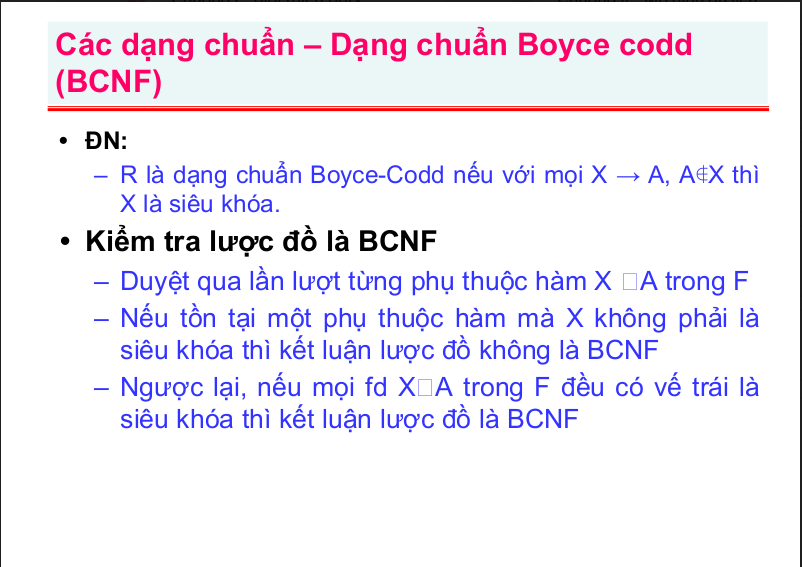
* Dạng chuẩn 2NF:



* Dạng chuẩn 3NF:



* Dạng chuẩn BCNF:



**Source Code:**

import itertools

# this class will used to store relation: their attr, functionalDepend

class Relation:

    def \_\_init\_\_(*self*):

*self*.attr\_set = set()  # store attr set

*self*.functional\_dependency = dict()  # store functional dependency

    def setAttrSet(*self*, *attribute*):

*self*.attr\_set = *attribute*

    def getAttrSet(*self*):

        return *self*.attr\_set

    def setFunctionalDependency(*self*, *x*):

*self*.functional\_dependency = *x*

    def getFunctionalDependency(*self*):

        return *self*.functional\_dependency

    def findClosureAttributeSet(*self*, *attrX*):

        attrX\_set = set(*attrX*)

        functionalDependencies = *self*.getFunctionalDependency()

        # Left is the side of Left of functional dependency

        Left = list()

        for item in functionalDependencies.keys():

            Left.append(set(item))

        # Right is the side of Right of functional dependency

        Right = list()

        for item in functionalDependencies.values():

            Right.append(set(item))

        # Find Closesure OF X

        closureOfX = attrX\_set

        numIndex = 0

        while True:

            numIndex = 0

            for item in Left:

                if item.issubset(closureOfX):

                    break

                else:

                    numIndex += 1

            if numIndex != len(Left):

                closureOfX = closureOfX.union(Right[numIndex])

                Left.remove(Left[numIndex])

                Right.remove(Right[numIndex])

            else:

                break

        return closureOfX

    def findAllSubsetOfX(*self*, *Xi*):

        AllSubsetOfX = list()

        for i in range(0, len(*Xi*)+1):

            for sub in itertools.combinations(*Xi*, i):

                AllSubsetOfX.append(sub)

        return AllSubsetOfX

    def findAllKey(*self*):

        Relational\_set = *self*.getAttrSet()

        # Left is the side of Left of functional dependency

        Left = list()

        functionalDependencies = *self*.getFunctionalDependency()

        for item in functionalDependencies.keys():

            Left.append(item)

        temp = list()

        for item in Left:

            for attr in item:

                temp.append(attr)

        # Left\_2 will be stored every item in Left

        Left\_2 = set(temp)

        # Right is the side of Right of functional dependency

        Right = list()

        for item in functionalDependencies.values():

            Right.append(set(item))

        temp = list()

        for item in Right:

            for attr in item:

                temp.append(attr)

        # Right\_2 will be stored every item in Right

        Right\_2 = set(temp)

        # SourceSet

        SourceSet = Relational\_set.difference(Right\_2)

        # IntermediateSet

        IntermediateSet = Left\_2.intersection(Right\_2)

        # superKeyList store list of super key set

        superKeyList = list()

        for subset in *self*.findAllSubsetOfX(IntermediateSet):

            if *self*.findClosureAttributeSet(SourceSet.union(subset)) == *self*.getAttrSet():

                superKeyList.append(SourceSet.union(subset))

        # notKey will be store super key and not key

        notKey = list()

        for i in range(0, len(superKeyList) - 1):

            for j in range(i + 1, len(superKeyList)):

                if superKeyList[i].issubset(superKeyList[j]):

                    notKey.append(superKeyList[j])

        # KeyList is solution

        KeyList = list()

        for superKey in superKeyList:

            if superKey not in notKey:

                KeyList.append(superKey)

        return KeyList

    def findAllSuperKey(*self*):

        Relational\_set = *self*.getAttrSet()

        # Left is the side of Left of functional dependency

        Left = list()

        functionalDependencies = *self*.getFunctionalDependency()

        for item in functionalDependencies.keys():

            Left.append(item)

        temp = list()

        for item in Left:

            for attr in item:

                temp.append(attr)

        # Left\_2 will be stored every item in Left

        Left\_2 = set(temp)

        # Right is the side of Right of functional dependency

        Right = list()

        for item in functionalDependencies.values():

            Right.append(set(item))

        temp = list()

        for item in Right:

            for attr in item:

                temp.append(attr)

        # Right\_2 will be stored every item in Right

        Right\_2 = set(temp)

        # SourceSet

        SourceSet = Relational\_set.difference(Right\_2)

        # IntermediateSet

        IntermediateSet = Left\_2.intersection(Right\_2)

        # superKeyList store list of super key set

        superKeyList = list()

        for subset in *self*.findAllSubsetOfX(IntermediateSet):

            if *self*.findClosureAttributeSet(SourceSet.union(subset)) == *self*.getAttrSet():

                superKeyList.append(SourceSet.union(subset))

        return superKeyList

    def findNormalization(*self*):

        keyList = *self*.findAllKey()

        functionalDependency = *self*.getFunctionalDependency()

        # KeyAttrSet

        keyAttrSet = set().union(\*keyList)

        KeyNotAttrSet = *self*.getAttrSet().difference(keyAttrSet)

        print("Attr key: ", keyAttrSet)

        print("Attr not key: ", KeyNotAttrSet)

        # 2NF Non-key attributes must be fully functionally dependent on the primary key

        # Method:

        # - Check each functional dependency X -> A:

        # - if X is subset of key and A is in attribute not key then isn't 2NF.

        # - else is 2NF

        # Notes: if X has fully key of key set then continue

        # End.

        flag = bool()  # to check 2NF

        LeftOfFD = list()

        RightOfFD = list()

        for key, value in functionalDependency.items():

            LeftOfFD.append(set(key))

            RightOfFD.append(set(value))

        for itemL, itemR in zip(LeftOfFD, RightOfFD):

            if (itemL.issubset(keyAttrSet)) and (itemR.issubset(KeyNotAttrSet)):

                if len(itemL) == len(keyAttrSet):

                    flag = True

                    continue

                flag = False

                break

            else:

                flag = True

        # 3NF Non-key attributes must directly depend on the primary key

        # Method:

        # - Find key

        # - Check: Relation is 2NF?;

        # - Check each functional dependency of F if exist X -> A:

        # -     X is not key and A is attribute non-key

        # -         => not is 3NF

        # - else 3NF

        # End.

        flag\_2 = bool()  # to check#3NF

        if (flag):

            LeftOfFD = list()

            RightOfFD = list()

            for key, value in functionalDependency.items():

                LeftOfFD.append(set(key))

                RightOfFD.append(set(value))

            for itemL, itemR in zip(LeftOfFD, RightOfFD):

                if itemL.issubset(KeyNotAttrSet) and itemR.issubset(KeyNotAttrSet):

                    flag\_2 = False

                    break

                else:

                    flag\_2 = True

        # BCNF There is no key attribute but a functional dependency on a non-key attribute.

        flag\_3 = bool()  # to check BCNF

        if (flag == True and flag\_2 == True):

            LeftOfFD = list()

            RightOfFD = list()

            for key, value in functionalDependency.items():

                LeftOfFD.append(set(key))

                RightOfFD.append(set(value))

            for itemL, itemR in zip(LeftOfFD, RightOfFD):

                if not (itemL in *self*.findAllSuperKey()):

                    flag\_3 = False

                    break

                else:

                    flag\_3 = True

        # 1NF

        if (flag):

            print(

                "2NF is the highest standard form because Non-key attributes is fully functionally dependent on the primary key")

        if (flag\_2):

            print(

                "3NF is the highest standard form because Non-key attributes directly depend on the primary key")

        if (flag\_3):

            print('BCNF is the highest standard form because There is no key attribute but a functional dependency on a non-key attribute.')

        if (flag == False and flag\_2 == False and flag\_3 == False):

            print('1NF is the highest standard form because it isn\'t 2NF, 3NF or BCNF')

        return

        #

# 1NF

relation = Relation()

relation.setAttrSet({'M', 'N', 'O', 'P', 'Q', 'R', 'S'})

relation.setFunctionalDependency(

    {'S': 'MR', 'NS': 'QM', 'PQ': 'RS', 'MO': 'NR', 'N': 'R'})

relation.findNormalization()

# 1NF

# relation = Relation()

# relation.setAttrSet({'M', 'N', 'O', 'P', 'Q', 'R', 'S'})

# relation.setFunctionalDependency(

#     {'S': 'MR', 'NS': 'QM', 'PQ': 'RS', 'MO': 'NR', 'N': 'R'})

# relation.findNormalization()

# 1NF

# relation = Relation()

# relation.setAttrSet({'A', 'S', 'P', 'I'})

# relation.setFunctionalDependency(

#     {'SI': 'P', 'S': 'A'})

# relation.findNormalization()

# 2NF

# relation = Relation()

# relation.setAttrSet({'A', 'B', 'C', 'D'})

# relation.setFunctionalDependency(

#     {'A': 'BC', 'B': 'D'})

# relation.findNormalization()

# 3NF

# relation = Relation()

# relation.setAttrSet({'C', 'S', 'Z'})

# relation.setFunctionalDependency(

#     {'CS': 'Z', 'Z': 'C'})

# relation.findNormalization()

# BCNF

# relation = Relation()

# relation.setAttrSet({'C', 'S', 'Z'})

# relation.setFunctionalDependency(

#     {'CS': 'Z'})

# relation.findNormalization()

**Kết quả chạy code:**

\_\_\_Với ví dụ ở phần 2:

# 1NF

relation = Relation()

relation.setAttrSet({'M', 'N', 'O', 'P', 'Q', 'R', 'S'})

relation.setFunctionalDependency(

    {'S': 'MR', 'NS': 'QM', 'PQ': 'RS', 'MO': 'NR', 'N': 'R'})

relation.findNormalization()

Kết quả:



\_\_Với ví dụ 1NF:

# 1NF

relation = Relation()

relation.setAttrSet({'A', 'S', 'P', 'I'})

relation.setFunctionalDependency(

    {'SI': 'P', 'S': 'A'})

relation.findNormalization()

Kết quả:



\_\_Với ví dụ 2NF:

# 2NF

relation = Relation()

relation.setAttrSet({'A', 'B', 'C', 'D'})

relation.setFunctionalDependency(

    {'A': 'BC', 'B': 'D'})

relation.findNormalization()

Kết quả:



\_\_Với ví dụ 3NF:

# 3NF

relation = Relation()

relation.setAttrSet({'C', 'S', 'Z'})

relation.setFunctionalDependency(

    {'CS': 'Z', 'Z': 'C'})

relation.findNormalization()

Kết quả:



\_\_Với ví dụ BCNF:

# BCNF

relation = Relation()

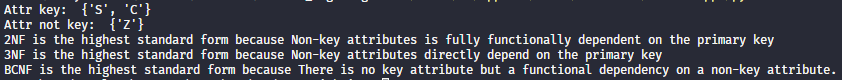
relation.setAttrSet({'C', 'S', 'Z'})

relation.setFunctionalDependency(

    {'CS': 'Z'})

relation.findNormalization()

Kết quả:



**TÀI LIỆU THAM KHẢO**

**Tiếng Việt**

Tìm bao đóng của tập thuộc tính, <https://sites.google.com/site/nhatnamcpt/thu-thuat-tin-hoc-1/lap-trinh/csdl-nang-cao/tim-bao-dhong-cua-tap-thuoc-tinh-phu-toi-thieu>

Tìm khóa và các khóa của lược đồ, <https://tltiendat.wordpress.com/2011/12/05/thu%E1%BA%ADt-ton-v-v-d%E1%BB%A5-tm-t%E1%BA%A5t-c%E1%BA%A3-cc-kha-trong-l%C6%B0%E1%BB%A3c-d%E1%BB%93-quan-h%E1%BB%87/>

**Tiếng Anh**