**CZ2007 Assignment 3: SQL**

SSP3 Group 1: Alice Chua Qin Hui, Foo Chuan Sheng, Tanay Bharadwaja, Vincent Ribli

|  |
| --- |
| **CREATE.SQL** |

CREATE TABLE Person\_R2(

addressP varchar(50),

zip char(6) NOT NULL,

stateP varchar(20) NOT NULL,

city varchar(20) NOT NULL,

PRIMARY KEY (addressP)

)

CREATE TABLE Person\_R1(

personID char(9),

name varchar(20) NOT NULL,

school varchar(100) NOT NULL,

addressP varchar(50) NOT NULL,

phone char(8) NOT NULL,

email varchar(50) NOT NULL UNIQUE

CHECK (email LIKE '%@%.%'),

PRIMARY KEY (personID),

FOREIGN KEY (addressP) REFERENCES Person\_R2

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Stakeholder(

personID char(9),

domain varchar(40) NOT NULL,

PRIMARY KEY (personID),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Professor(

personID char(9),

expertiseField varchar(40) NOT NULL,

PRIMARY KEY (personID),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Student(

personID char(9),

studentID char(9) NOT NULL UNIQUE,

admissionDate date NOT NULL,

major varchar(50) NOT NULL,

minor varchar(50),

PRIMARY KEY (personID),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Staff(

personID char(9) NOT NULL,

staffID char(9) NOT NULL,

dateHired datetime NOT NULL,

position varchar(20) NOT NULL,

PRIMARY KEY (personID),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Undergraduate(

personID char(9) NOT NULL,

PRIMARY KEY (personID),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Graduate(

personID char(9) NOT NULL,

PRIMARY KEY (personID),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Lab(

nameL varchar(20),

school varchar(50),

location varchar(50),

PRIMARY KEY (nameL,school)

)

CREATE TABLE Research\_Lab(

nameRL varchar(20),

school varchar(50),

researchArea varchar(50),

PRIMARY KEY (nameRL, school),

FOREIGN KEY (nameRL, school) REFERENCES Lab

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Teaching\_Lab(

name varchar(20),

school varchar(50),

labType varchar(20),

PRIMARY KEY (name, school),

FOREIGN KEY (name,school) REFERENCES Lab

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Technical\_Staff(

personID char(9) NOT NULL,

labName varchar(20) NOT NULL,

labSchool varchar(50) NOT NULL,

PRIMARY KEY (personID, labName, labSchool),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (labName, labSchool) REFERENCES Lab(nameL, school)

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Admin\_Staff(

personID char(9) NOT NULL,

office varchar(20) NOT NULL,

officePhone char(8) NOT NULL,

portfolio varchar(20) NOT NULL,

PRIMARY KEY (personID),

FOREIGN KEY (personID) REFERENCES Person\_R1

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Timetable(

dayT varchar(20) NOT NULL

CHECK(dayT <> 'Saturday' AND dayT <> 'Sunday'),

timeTstart TIME (0) NOT NULL

CHECK(timeTstart >= '08:00:00'),

timeTend TIME (0) NOT NULL

CHECK(timeTend <= '18:00:00'),

venue varchar(20) NOT NULL,

classType varchar(20) NOT NULL,

CHECK(timeTstart < timeTend),

PRIMARY KEY (dayT,timeTstart,timeTend,venue)

)

CREATE TABLE Course(

courseCode char(6) NOT NULL UNIQUE,

courseName varchar(50) NOT NULL,

PRIMARY KEY (courseCode)

)

CREATE TABLE CommentSuggestion(

topic varchar(20),

dateCS date,

stakeholderPersonID char(9),

statusCS varchar(10) NOT NULL,

PRIMARY KEY (topic, dateCS, stakeholderPersonID),

FOREIGN KEY (stakeholderPersonID) REFERENCES Stakeholder(personID)

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Experiment(

dayE varchar(20) NOT NULL

CHECK(dayE <> 'Saturday' AND dayE <> 'Sunday'),

timeEstart TIME (0) NOT NULL

CHECK(timeEstart >= '08:00:00'),

timeEend TIME (0) NOT NULL

CHECK(timeEend <= '18:00:00'),

undergraduatePersonID char(9) NOT NULL,

labName varchar(20) NOT NULL,

labSchool varchar(50) NOT NULL,

CHECK(timeEstart < timeEend),

PRIMARY KEY (dayE, timeEstart, timeEend, undergraduatePersonID, labName,  
labSchool),

FOREIGN KEY (undergraduatePersonID) REFERENCES Undergraduate(personID)

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (labName, labSchool) REFERENCES Lab(nameL, school)

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE Equipment(

ID varchar(9),

nameE varchar(20),

labName varchar(20),

labSchool varchar(50),

modelNo varchar(10) NOT NULL,

datePurchased date NOT NULL,

PRIMARY KEY (ID, nameE, labName, labSchool),

FOREIGN KEY (labName, labSchool) REFERENCES Lab(nameL, school)

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE teach(

dayT varchar(20) NOT NULL

CHECK(dayT <> 'Saturday' AND dayT <> 'Sunday'),

timeTstart TIME (0) NOT NULL

CHECK(timeTstart >= '08:00:00'),

timeTend TIME (0) NOT NULL

CHECK(timeTend <= '18:00:00'),

venue varchar(20),

courseCode char(6),

professorPersonID char(9),

CHECK(timeTstart < timeTend),

PRIMARY KEY (dayT, timeTstart, timeTend, venue, courseCode, professorPersonID),

FOREIGN KEY (professorPersonID) REFERENCES Professor(personID)

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (dayT, timeTstart, timeTend, venue) REFERENCES

Timetable(dayT,timeTstart,timeTend,venue)

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (courseCode) REFERENCES Course(courseCode)

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE takes(

personID char(9),

courseCode char(6),

PRIMARY KEY (courseCode, personID),

FOREIGN KEY (courseCode) REFERENCES Course

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (personID) REFERENCES Student

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE partOf(

courseCode char(6),

dayPO varchar(20) NOT NULL

CHECK(dayPO <> 'Saturday' AND dayPO <> 'Sunday'),

timePOstart TIME (0) NOT NULL

CHECK (timePOstart >= '08:00:00'),

timePOend TIME (0) NOT NULL

CHECK(timePOend <= '18:00:00'),

personID char(9),

labName varchar(20),

labSchool varchar(50),

CHECK(timePOstart < timePOend),

PRIMARY KEY (courseCode, dayPO, timePOstart, timePOend, personID, labName,

labSchool),

FOREIGN KEY (courseCode) REFERENCES Course

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (dayPO, timePOstart, timePOend, personID, labName, labSchool) REFERENCES Experiment(dayE, timeEstart, timeEend, undergraduatePersonID, labName, labSchool)

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE assignGraduateResearchLab(

personID char(9),

labName varchar(20),

labSchool varchar(50),

PRIMARY KEY (personID, labName, labSchool),

FOREIGN KEY (personID) REFERENCES Graduate(personID)

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (labName, labSchool) REFERENCES Research\_Lab(nameRL, school)

ON DELETE cascade

ON UPDATE cascade

)

CREATE TABLE assignCommentSuggestionAdminStaff(

AdminStaffPersonID char(9),

StakeholderPersonID char(9),

topic varchar(20),

dateA date,

PRIMARY KEY (AdminStaffPersonID, StakeholderPersonID),

FOREIGN KEY (AdminStaffPersonID) REFERENCES Admin\_Staff(personID)

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (StakeholderPersonID) REFERENCES Stakeholder(personID)

ON DELETE NO ACTION

ON UPDATE NO ACTION

)

CREATE TABLE supervisedBy(

ProfessorPersonID char(9),

GraduatePersonID char(9),

topic varchar(50),

PRIMARY KEY (ProfessorPersonID, GraduatePersonID),

FOREIGN KEY (ProfessorPersonID) REFERENCES Professor(personID)

ON DELETE cascade

ON UPDATE cascade,

FOREIGN KEY (GraduatePersonID) REFERENCES Graduate(personID)

ON DELETE NO ACTION

ON UPDATE NO ACTION

)

|  |
| --- |
| **TRIGGER\_student.SQL** |

CREATE TRIGGER studentMaxCourse

ON takes

FOR INSERT

AS DECLARE @NewPersonID CHAR(9)

SELECT @NewPersonID = ins.personID FROM inserted ins;

IF (@NewPersonID IN (

SELECT personID

FROM takes

GROUP BY personID

HAVING COUNT(\*) >= 6

))

BEGIN

RAISERROR('Maximum number of Courses reached!', 16, 1)

ROLLBACK TRANSACTION

END

|  |
| --- |
| **TRIGGER\_professor.SQL** |

CREATE TRIGGER professorMaxCourse

ON teach

FOR INSERT

AS DECLARE @NewPersonID CHAR(9)

SELECT @NewPersonID = ins.professorPersonID FROM inserted ins;

IF (@NewPersonID IN (

SELECT professorPersonID

FROM teach

GROUP BY professorPersonID

HAVING COUNT(\*) >= 3

))

BEGIN

RAISERROR('Maximum number of Courses reached!', 16, 1)

ROLLBACK TRANSACTION

END

|  |
| --- |
| **QUERIES.SQL** |

**1** Find all Stakeholders who belong to the public domain.

SELECT P.name, P.personID

FROM Stakeholder S, Person\_R1 P

WHERE S.personID = P.personID

AND S.domain = 'Public';

|  |  |
| --- | --- |
| **name** | **personID** |
| Tamatha Billingsly | S6048755N |
| Abel Harbor | S6255118D |
| Mafalda Pavon | S6293407X |
| Shakia Oliverio | S6299846Z |
| Lady Aultman | S6818581R |
| Clara Saulnier | S7116670Q |
| Karl Lex | S7138455N |
| Nila Combes | S7487820E |
| Candis Carnley | S7994678N |
| Adina Kiley | S8019609S |
| Hubert Eacret | S8746763I |
| Deanne Shultz | S8870503O |

**2** Find all Stakeholders who have provided at least five comments or suggestions.

SELECT P.name, P.personID

FROM Stakeholder S, CommentSuggestion C, Person\_R1 P

WHERE S.personID = C.stakeholderPersonID

AND S.personID = P.personID

GROUP BY P.name, P.personID

HAVING COUNT(\*) >= 5;

|  |  |
| --- | --- |
| **name** | **personID** |
| Clara Saulnier | S7116670Q |
| Candis Carnley | S7994678N |
| Adina Kiley | S8019609S |

**3** Find Graduates who are supervised by more than one professor and assigned to more than one research laboratory.

SELECT P.name, P.personID

FROM supervisedBy S, Person\_R1 P

WHERE S.GraduatePersonID = P.personID

GROUP BY P.name, P.personID

HAVING COUNT(\*) > 1

INTERSECT

SELECT P.name, P.personID

FROM assignGraduateResearchLab A, Person\_R1 P

WHERE A.personID = P.personID

GROUP BY P.name, P.personID

HAVING COUNT(\*) > 1;

|  |  |
| --- | --- |
| **name** | **personID** |
| Wendy Moser | S9664494X |
| Bernard Frisch | S9666826Y |
| Vonda Onstad | S9899788Y |

**4** Find all Professors who teach more than one courses in the semester.

WITH Temp(courseCode, professorPersonID)

AS (SELECT DISTINCT courseCode, professorPersonID

FROM teach

AS temp)

SELECT P.name, P.personID

FROM temp T, Person\_R1 P

WHERE T.professorPersonID = P.personID

GROUP BY P.name, P.personID

HAVING COUNT(\*) > 1;

|  |  |
| --- | --- |
| **name** | **personID** |
| Fabian Deardorff | S7690922X |
| Efrain Spurrier | S8468939S |
| Hubert Borelli | S8972332B |

**5** List all the Equipment belonging to a particular Laboratory.

SELECT labName, ID, nameE

FROM Equipment

ORDER BY labName;

|  |  |  |
| --- | --- | --- |
| **labName** | **ID** | **nameE** |
| Computer Lab 1 | E1134547 | Computer |
| Computer Lab 1 | E1134548 | Arduino |
| Computer Lab 1 | E1134549 | Robotics Kit |
| Computer Lab 1 | E1134550 | Lego Mindstorms |
| Computer Lab 1 | E1134551 | Sensehat |
| Computer Lab 2 | E1134554 | Computer |
| Computer Lab 2 | E1134555 | Arduino |
| Computer Lab 2 | E1134556 | Robotics Kit |
| Computer Lab 2 | E1134557 | Lego Mindstorms |
| Hardware Lab 3 | E1134542 | Raspberry Pi |
| Hardware Lab 3 | E1134543 | Computer |
| Hardware Lab 3 | E1134544 | Arduino |
| Hardware Lab 3 | E1134545 | Robotics Kit |
| Hardware Lab 3 | E1134546 | Lego Mindstorms |
| Physics Lab | E1134558 | Computer |
| Physics Lab | E1134559 | Arduino |
| Physics Lab | E1134560 | Robotics Kit |
| Physics Lab | E1134561 | Lego Mindstorms |
| Research Lab 1 | E1134562 | Computer |
| Research Lab 2 | E1134565 | Computer |
| Research Lab 3 | E1134568 | Computer |
| Research Lab 4 | E1134571 | Computer |
| Research Lab 5 | E1134574 | Computer |
| Research Lab 6 | E1134578 | Computer |
| Software Lab 1 | E1134535 | Raspberry Pi |
| Software Lab 1 | E1134536 | Computer |
| Software Lab 1 | E1134537 | Arduino |
| Software Lab 1 | E1134538 | Robotics Kit |
| Software Lab 2 | E1134539 | Lego Mindstorms |
| Software Lab 2 | E1134540 | Sensehat |
| Software Lab 2 | E1134541 | Computer |

**6** Find all Undergraduates who have not attended at least one laboratory experiments.

WITH Temp(personID)

AS (SELECT U.personID

FROM Undergraduate U

EXCEPT

SELECT DISTINCT E.undergraduatePersonID

FROM Experiment E)

SELECT P.name, P.personID

FROM Temp T, Person\_R1 P

WHERE P.personID = T.personID;

|  |  |
| --- | --- |
| **name** | **personID** |
| Glynis Dumont | S9628046W |
| Sterling Kamer | S9637726Z |
| Katlyn Coache | S9682691N |
| Genny Mckeon | S9719396U |
| Coral Emberton | S9726844G |
| Loris Trumbo | S9731179S |
| Mack Dinwiddie | S9750630U |
| Regine Obrien | S9770748L |
| Hwa Diana | S9775721E |
| Noe Whitely | S9786239J |
| Sharyl Cutsforth | S9796567Y |
| Dalila Snedden | S9817088U |
| Brooks Despres | S9828865E |
| Branda Nakamura | S9867108D |
| Romana Tardif | S9883595K |
| Cheryle Foote | S9915654K |
| Shayne Sedgwick | S9930318Z |
| Rozanne Starr | S9945680P |
| Lakeesha Mcarthur | S9963538Q |

**7** List all Graduates who are doing research and taking courses in the semester

SELECT P.name, P.personID

FROM Person\_R1 P

WHERE P.personID IN (

SELECT G.personID

FROM Graduate G, takes t

WHERE G.personID = t.personID);

|  |  |
| --- | --- |
| **name** | **personID** |
| Kaleigh Bjerke | S9628927W |
| Elana Bull | S9639826W |
| Shawanda Oestreich | S9662439O |
| Bernard Frisch | S9666826Y |
| Daisey Haugen | S9677057V |
| Fumiko Yost | S9683984N |
| Palmira Bavaro | S9844485R |
| Rubye Fray | S9878146A |
| Shaquita Fury | S9881948R |
| Vonda Onstad | S9899788Y |
| Vita Carrizales | S9925377M |
| Hobert Banks | S9990896A |