COMS 4111-HW3
HAORAN GUO
UNI: hg2461

PART1-DDL, VIEW, PROCEDURE, FUNCTION and sample data

```
/*CREATE DATABASE HW3;*/

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `Student`
--
DROP TABLE IF EXISTS `student`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `student` (
  `UNI` varchar(12) NOT NULL,
  `School` varchar(32) NOT NULL,
  `Year` int(4) NOT NULL,
  PRIMARY KEY (`UNI`),
  KEY `student_fk`(`UNI`),
  CONSTRAINT `student_fk` FOREIGN KEY (`UNI`) REFERENCES `Person` (`UNI`) ON
DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `Student`
--
LOCK TABLES `student` WRITE;
/*!40000 ALTER TABLE `Student` DISABLE KEYS */;
/*!40000 ALTER TABLE `Student` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `Faculty`
--
DROP TABLE IF EXISTS `faculty`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `faculty` (
  `UNI` varchar(12) NOT NULL,
  `Pay_grade` varchar(20) NOT NULL,
  `title` varchar(15) NOT NULL,
  `department_code` varchar(14) NOT NULL,
```

```
  PRIMARY KEY (`UNI`),
  KEY `faculty_fk`(`UNI`),
  KEY `fac_department_fk`(`department_code`),
  CONSTRAINT `faculty_fk` FOREIGN KEY (`UNI`) REFERENCES `Person` (`UNI`) ON
DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `fac_department_fk` FOREIGN KEY (`department_code`) REFERENCES
`department` (`code`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `Faculty`
--
LOCK TABLES `Faculty` WRITE;
/*!40000 ALTER TABLE `Faculty` DISABLE KEYS */;
/*!40000 ALTER TABLE `Faculty` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `Person`
--
DROP TABLE IF EXISTS `Person`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `Person` (
  `UNI` varchar(12) NOT NULL,
  `last_name` varchar(20) NOT NULL,
  `first_name` varchar(20) NOT NULL,
  PRIMARY KEY (`UNI`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `Person`
--
LOCK TABLES `Person` WRITE;
/*!40000 ALTER TABLE `Person` DISABLE KEYS */;
/*!40000 ALTER TABLE `Person` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `course_participant`
--
DROP TABLE IF EXISTS `course_participant`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `course_participant` (
  `UNI` varchar(12) NOT NULL,
  `section_call_no` char(5) NOT NULL,
  `role` varchar(20) NOT NULL,
  PRIMARY KEY (`UNI`,`section_call_no`),
  KEY `cp_section_fk` (`section_call_no`),
  CONSTRAINT `cp_participant_fk` FOREIGN KEY (`UNI`) REFERENCES `Person` (`UNI`)
ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `cp_section_fk` FOREIGN KEY (`section_call_no`) REFERENCES `sections`
(`call_no`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `course_participant`
--
LOCK TABLES `course_participant` WRITE;
/*!40000 ALTER TABLE `course_participant` DISABLE KEYS */;
INSERT INTO `course_participant`
VALUES ('AMTO1','00001','student'),
('AMTO1','00002','student'),
('AMTO1','00003','student'),
('AMTO1','00004','student'),
('AMTO1','00005','student'),
('AMTO1','00006','student'),
('AMTO1','00007','student'),
('AMTO1','00008','student'),
('GUHA1','00014','student'),
('GUHA1','00002','student'),
('GUHA1','00003','student'),
('GUHA1','00004','student'),
('GUHA1','00006','student'),
('GUHA2','00005','student'),
('GUHA2','00006','student'),
('GUHA2','00007','student'),
('GUHA2','00008','student'),
('GUHA2','00015','student'),
('FEDO1','00001','Instructor'),
('FEDO1','00002','Instructor'),
('FEDO1','00003','Instructor'),
('FEDO1','00006','Instructor'),
('FEDO1','00014','Instructor'),
('FEDO1','00008','Instructor'),
('HASO1','00004','Instructor')
;
/*!40000 ALTER TABLE `course_participant` ENABLE KEYS */;
UNLOCK TABLES;
/*SELECT * FROM `course_participant`;*/

--
-- Table structure for table `course_prereqs`
--
DROP TABLE IF EXISTS `course_prereqs`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `course_prereqs` (
  `course_id` varchar(12) NOT NULL,
  `prereq_id` varchar(12) NOT NULL,
  PRIMARY KEY (`course_id`,`prereq_id`),
  KEY `prereq_prereq_fk` (`prereq_id`),
  CONSTRAINT `prereq_course_fk` FOREIGN KEY (`course_id`) REFERENCES `courses`
(`course_id`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT `prereq_prereq_fk` FOREIGN KEY (`prereq_id`) REFERENCES `courses`
(`course_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
```

```
-- Dumping data for table `course_prereqs`
--
LOCK TABLES `course_prereqs` WRITE;
/*!40000 ALTER TABLE `course_prereqs` DISABLE KEYS */;
INSERT INTO `course_prereqs`
VALUES ('COMSW4111','COMSE1006'),
('COMSW4111','COMSW3270'),
('COMSW4119','COMSW3270'),
('EENG6893','COMSW3270'),
('COMSW4231','COMSW3270'),
('EENG6991','COMSW3270'),
('COMSW4119','COMSE1006'),
('EENG6893','COMSE1006'),
('COMSW4231','COMSE1006'),
('EENG6991','COMSE1006')
;
/*!40000 ALTER TABLE `course_prereqs` ENABLE KEYS */;
UNLOCK TABLES;
/*SELECT * FROM `course_prereqs`;*/

--
-- Table structure for table `courses`
--
DROP TABLE IF EXISTS `courses`;
/*!40101 SET @saved_cs_client         = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `courses` (
   `dept_code` char(4) NOT NULL,
   `faculty_code` enum('BC','C','E','F','G','V','W','X') NOT NULL,
   `level` enum('0','1','2','3','4','6','8','9') NOT NULL,
   `number` char(3) NOT NULL,
   `title` varchar(32) NOT NULL,
   `description` varchar(128) NOT NULL,
   `course_id` varchar(12) GENERATED ALWAYS AS
(concat(`dept_code`,`faculty_code`,`level`,`number`)) STORED,
   `full_number` char(4) GENERATED ALWAYS AS (concat(`level`,`number`)) VIRTUAL,
   PRIMARY KEY (`dept_code`,`faculty_code`,`level`,`number`),
   UNIQUE KEY `course_id` (`course_id`),
   FULLTEXT KEY `keywords` (`title`,`description`),
   CONSTRAINT `course2_dept_fk` FOREIGN KEY (`dept_code`) REFERENCES `department`
(`code`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `courses`
--
LOCK TABLES `courses` WRITE;
/*!40000 ALTER TABLE `courses` DISABLE KEYS */;
INSERT INTO `courses` (`dept_code`, `faculty_code`, `level`, `number`, `title`, `description`)
VALUES ('COMS','E','1','006','Intro. to Program for Eng.','Graduate students do not need to
take.'),
('COMS','W','3','270','Data Structures','Seems safe to take.'),
('EENG','E','6','893','Big Data Analysis','Would have a heavy workload'),
('EENG','E','6','991','Reinforcement Learning','We shall see'),
('COMS','W','4','119','Computer Networks','Midterm grades would not be curved'),
('COMS','W','4','231','Analysis of Algorithm','Testing your IQ'),
```

```
('COMS','W','4','111','Intro. to Databases','Might be interesting');
/*!40000 ALTER TABLE `courses` ENABLE KEYS */;
UNLOCK TABLES;
/*SELECT * FROM courses;*/

--
-- Table structure for table `department`
--
DROP TABLE IF EXISTS `department`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `department` (
  `code` char(4) NOT NULL,
  `name` varchar(32) DEFAULT NULL,
  PRIMARY KEY (`code`),
  UNIQUE KEY `name_UNIQUE` (`name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `department`
--
LOCK TABLES `department` WRITE;
/*!40000 ALTER TABLE `department` DISABLE KEYS */;
INSERT INTO `department` VALUES ('COMS','Computer Science'),('EENG','Electrical
Engineering');
/*!40000 ALTER TABLE `department` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `sections`
--
DROP TABLE IF EXISTS `sections`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `sections` (
  `call_no` char(5) NOT NULL,
  `course_id` varchar(12) NOT NULL,
  `section_no` varchar(45) NOT NULL,
  `limit` int NOT NULL,
  `year` int(11) NOT NULL,
  `semester` varchar(45) NOT NULL,
  `section_key` varchar(45) GENERATED ALWAYS AS
(concat(`year`,`semester`,`course_id`,`section_no`)) STORED,
  PRIMARY KEY (`call_no`),
  UNIQUE KEY `unique` (`course_id`,`section_no`,`year`,`semester`),
  CONSTRAINT `section_course_fk` FOREIGN KEY (`course_id`) REFERENCES `courses`
(`course_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `sections`
--
LOCK TABLES `sections` WRITE;
/*!40000 ALTER TABLE `sections` DISABLE KEYS */;
INSERT INTO `sections` (`call_no`, `course_id`,`limit`, `section_no`, `year`, `semester`)
```

```
VALUES ('00001','COMSW4111',5, '1',2017,'1'),
('00002','COMSW4111',6,'2',2014,'3'),
('00003','COMSW4111',4,'3',2014,'3'),
('00004','COMSE1006',50,'2',2015,'3'),
('00005','COMSE1006',150,'3',2015,'3'),
('00006','COMSW3270',55,'1',2017,'3'),
('00007','COMSW3270',80,'3',2016,'4'),
('00008','COMSW4119',55,'1',2016,'4'),
('00009','COMSW4119',60,'3',2017,'1'),
('00010','COMSW4231',20,'2',2017,'1'),
('00011','COMSW4231',10,'3',2017,'1'),
('00012','COMSW6893',15,'4',2017,'1'),
('00013','COMSW6893',15,'3',2017,'1'),
('00014','COMSW6991',2,'2',2017,'1'),
('00015','COMSW6991',17,'3',2017,'1');
/*!40000 ALTER TABLE `sections` ENABLE KEYS */;
UNLOCK TABLES;
/*SELECT * FROM sections;*/

--
-- VIEW faculty_all_info
--
DROP VIEW IF EXISTS `faculty_all_info`;
CREATE ALGORITHM = UNDEFINED
DEFINER = `root`@`localhost`
SQL SECURITY DEFINER
VIEW `faculty_all_info` AS
	SELECT
		person.UNI, person.last_name, person.first_name,
		faculty.title, faculty.Pay_grade, faculty.department_code
	FROM person join faculty
	ON person.UNI = faculty.UNI;

--
-- VIEW student_all_info
--
DROP VIEW IF EXISTS `student_all_info`;
CREATE ALGORITHM = UNDEFINED
DEFINER = `root`@`localhost`
SQL SECURITY DEFINER
VIEW `student_all_info` AS
	SELECT
		person.UNI, person.last_name, person.first_name,
		student.School, student.`Year`
	FROM person join student
	ON person.UNI = student.UNI;

--
-- FUCTION TO JUDGE WHETHER THIS SEMESTER IS A CURRENT ONE
--
DELIMITER    ;
DROP FUNCTION IF EXISTS `is_current_section`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` FUNCTION `is_current_section`(`year` INT(11),
`semester` VARCHAR(45))
RETURNS int(1)
BEGIN
```

```
    DECLARE `time` CHAR(25);
    DECLARE `now_year` CHAR(4);
    DECLARE `now_month` CHAR(2);
    DECLARE `now_semester` CHAR(45);
    DECLARE `is_current` int(1);

    SET `time`= DATE(now());
    SET `now_year`= substr(`time`, 1);
    SET `now_month`= substr(`time`, 6);
    SET `is_current` = 0;

    IF `now_month` in ('1','2','3','4','5') THEN SET `now_semester`='2';
    END IF;
    IF `now_month` in ('6','7') THEN SET `now_semester`='3';
    END IF;
    IF `now_month` in ('8') THEN SET `now_semester`='4';
    END IF;
    IF `now_month` in ('9','10','11','12') THEN SET `now_semester`='1';
    END IF;

    IF `now_year` = `year` AND `semester` = `now_semester` THEN SET `is_current` = 1 ;
    END IF;

    RETURN `is_current`;

END;;

--
-- CHECK FUNCTION `is_current_section`
--
/*DELIMITER    ;
DROP PROCEDURE IF EXISTS `checktime`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `checktime`()
BEGIN
    SELECT `is_current_section`(2017,'1');
END;;
CALL `checktime`;
*/

--
-- VIEW COURSE_TAKEN
--
DELIMITER    ;
DROP VIEW IF EXISTS `course_taken`;
DELIMITER    ;;
CREATE ALGORITHM = UNDEFINED
DEFINER = `root`@`localhost`
SQL SECURITY DEFINER
VIEW `course_taken` AS
    SELECT
        UNI, courses.course_id AS course_id, courses.title AS course_name,
        participant_section.`year` as `year`, participant_section.semester as semester
    FROM
        courses join
    (SELECT
```

```
        UNI, sections.course_id as course_id,
        sections.`year` as `year`, sections.semester as semester
    FROM
        sections join
    (SELECT
        course_participant.UNI as UNI, course_participant.section_call_no as section_call_no
    FROM
        course_participant join
    (SELECT
        person.UNI as person_uni, student.UNI as student_uni
    FROM
        person join student
    ON person.UNI=student.UNI) AS student_person
    ON course_participant.UNI=person_uni) AS person_participant
    ON sections.call_no = section_call_no) AS participant_section
    ON participant_section.course_id = courses.course_id
    WHERE `is_current_section`(`year`, `semester`)=0;; /*To test this function, set 0 to 1*/

--
-- VIEW COURSE_TAUGHT
--
DROP VIEW IF EXISTS `course_taught`;
CREATE ALGORITHM = UNDEFINED
DEFINER = `root`@`localhost`
SQL SECURITY DEFINER
VIEW `course_taught` AS
    SELECT
        UNI, courses.course_id AS course_id, courses.title AS course_name,
        participant_setion.`year` as `year`, participant_setion.semester as semester
    FROM
        courses join
    (SELECT
        UNI, sections.course_id as course_id,
        sections.`year` as `year`, sections.semester as semester
    FROM
        sections join
    (SELECT
        course_participant.UNI as UNI, course_participant.section_call_no as section_call_no
    FROM
        course_participant join
    (SELECT
        person.UNI as person_uni, faculty.UNI as faculty_uni
    FROM
        person join faculty
    ON person.UNI=faculty.UNI) AS faculty_person
    ON UNI=person_uni) AS person_participant
    ON sections.call_no = section_call_no) AS participant_setion
    ON participant_setion.course_id = courses.course_id;
/*SELECT * FROM course_taught;*/

--
-- Functions
--
DELIMITER    ;
DROP FUNCTION IF EXISTS `generate_uni`;
DELIMITER    ;;
```

```
CREATE DEFINER=`root`@`localhost` FUNCTION `generate_uni`(last_name VARCHAR(20),
first_name VARCHAR(20))
RETURNS VARCHAR(8) CHARSET utf8 deterministic
BEGIN

	DECLARE c1 CHAR(2);
	DECLARE c2 CHAR(2);
	DECLARE cc CHAR(4);
	DECLARE prefix CHAR(10);
	DECLARE uniCount INT(20);
	DECLARE newuniCount INT(20);
	DECLARE newUni VARCHAR(8);

	SET c1 = UPPER(SUBSTR(last_name, 1));
	SET c2 = UPPER(SUBSTR(first_name, 1));
	SET cc = CONCAT(c1, c2);
	SET prefix = CONCAT(cc,'%');

	SELECT COUNT(*) INTO uniCount FROM person WHERE (UNI LIKE prefix);
	SET newuniCount = uniCount+1;
	SET newUni = CONCAT(cc, newuniCount);
RETURN newUni;
END;;

--
-- Trigger-faculty_BERORE_INSERT
--
DELIMITER    ;
DROP TRIGGER IF EXISTS `faculty_BERORE_INSERT`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` TRIGGER `faculty_BERORE_INSERT` BEFORE
INSERT ON `faculty` FOR EACH ROW
BEGIN
	IF New.pay_grade NOT IN (1, 2, 3, 4, 5 ,6, 7)
	THEN SIGNAL SQLSTATE '45001'
		SET MESSAGE_TEXT = 'INVALID PAY GRADE';
	END IF;
END;;

--
-- Trigger-student_BERORE_INSERT
--
DELIMITER    ;
DROP TRIGGER IF EXISTS `student_BERORE_INSERT`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` TRIGGER `student_BERORE_INSERT` BEFORE
INSERT ON `student` FOR EACH ROW
BEGIN
	IF New.`year` NOT BETWEEN 1700 AND 2018
	THEN SIGNAL SQLSTATE '45002'
		SET MESSAGE_TEXT = 'INVALID year';
	END IF;
END;;

--
-- Trigger-faculty_BERORE_UPDATE
--
```

```
DELIMITER    ;
DROP TRIGGER IF EXISTS `faculty_BERORE_UPDATE`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` TRIGGER `faculty_BERORE_UPDATE`
BEFORE UPDATE ON `faculty` FOR EACH ROW
BEGIN
      IF NOT EXISTS
      (SELECT * FROM faculty
            WHERE faculty.UNI=New.UNI) THEN
      SIGNAL SQLSTATE '45002'
            SET MESSAGE_TEXT = 'INVALID UNI';
      END IF;

      IF NOT EXISTS
      (SELECT * FROM person
            WHERE person.UNI=New.UNI) THEN
      SIGNAL SQLSTATE '45002'
            SET MESSAGE_TEXT = 'INVALID UNI';
      END IF;

      IF New.pay_grade NOT IN (1, 2, 3, 4, 5 ,6, 7)
      THEN SIGNAL SQLSTATE '45001'
            SET MESSAGE_TEXT = 'INVALID PAY GRADE';
      END IF;
END;;

--
-- Trigger-student_BERORE_UPDATE
--
DELIMITER    ;
DROP TRIGGER IF EXISTS `student_BERORE_UPDATE`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` TRIGGER `student_BERORE_UPDATE`
BEFORE UPDATE ON `student` FOR EACH ROW
BEGIN
      IF NOT EXISTS
      (SELECT * FROM student
            WHERE student.UNI=New.UNI) THEN
      SIGNAL SQLSTATE '45002'
            SET MESSAGE_TEXT = 'INVALID UNI';
      END IF;

      IF NOT EXISTS
      (SELECT * FROM person
            WHERE person.UNI=New.UNI) THEN
      SIGNAL SQLSTATE '45002'
            SET MESSAGE_TEXT = 'INVALID UNI';
      END IF;

      IF New.`year` NOT BETWEEN 1700 AND 2018
      THEN SIGNAL SQLSTATE '45002'
            SET MESSAGE_TEXT = 'INVALID year';
      END IF;
END;;

--
-- Procedure-insert_faculty
```

```
--
DELIMITER    ;
DROP PROCEDURE IF EXISTS `insert_faculty`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `insert_faculty`(last_name
VARCHAR(20),
first_name VARCHAR(20), pay_grade varchar(20), title varchar(15), department_code char(4))
BEGIN
	DECLARE UNI varchar(12);
	SET UNI = `generate_uni`(last_name, first_name);
	INSERT INTO faculty(UNI, pay_grade, title, department_code)
	VALUES(UNI, pay_grade, title, department_code);
	INSERT INTO person(UNI, last_name, first_name)
	VALUES(UNI, last_name, first_name);
END;;

--
-- Procedure-insert_student
--
DELIMITER    ;
DROP PROCEDURE IF EXISTS `insert_student`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `insert_student`(last_name
VARCHAR(20),
first_name VARCHAR(20), school varchar(32), `year` INT(4))
BEGIN
	DECLARE UNI varchar(12);
	SET UNI = `generate_uni`(last_name, first_name);
	INSERT INTO student(UNI, school, `year`)
	VALUES(UNI, school, `year`);
	INSERT INTO person(UNI, last_name, first_name)
	VALUES(UNI, last_name, first_name);
END;;

--
-- Procedure-update_faculty
--
DELIMITER    ;
DROP PROCEDURE IF EXISTS `update_faculty`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `update_faculty`(UNI varchar(12),
last_name VARCHAR(20), first_name VARCHAR(20), pay_grade varchar(20), title varchar(15),
department_code char(4))
BEGIN
	SET FOREIGN_KEY_CHECKS=0;

	UPDATE faculty
	SET faculty.department_code=department_code, faculty.Pay_grade = pay_grade,
		faculty.title = title
	WHERE faculty.UNI = UNI;

	UPDATE person
	SET person.first_name = first_name, person.last_name=last_name
	WHERE person.UNI = UNI;
	SET FOREIGN_KEY_CHECKS=1;
END;;
```

```
--
-- Procedure-update_student
--
DELIMITER    ;
DROP PROCEDURE IF EXISTS `update_student`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `update_student`(UNI varchar(12),
last_name VARCHAR(20), first_name VARCHAR(20), school varchar(32), `year` INT)
BEGIN
	SET FOREIGN_KEY_CHECKS=0;

	UPDATE student
	SET student.school = school, student.`year` = `year`
	WHERE student.UNI = UNI;

	UPDATE person
	SET person.first_name = first_name, person.last_name=last_name
	WHERE person.UNI = UNI;

	SET FOREIGN_KEY_CHECKS=1;
END;;

--
-- Procedure- delete_faculty
--
DELIMITER    ;
DROP PROCEDURE IF EXISTS `delete_faculty`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `delete_faculty`(UNI varchar(12))
BEGIN
	SET FOREIGN_KEY_CHECKS=0;

	DELETE FROM faculty
	WHERE faculty.UNI = UNI;

	DELETE FROM person
	WHERE person.UNI = UNI;

	 SET FOREIGN_KEY_CHECKS=1;
END;;

--
-- Procedure- delete_student
--
DELIMITER    ;
DROP PROCEDURE IF EXISTS `delete_student`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `delete_student`(UNI varchar(12))
BEGIN
	SET FOREIGN_KEY_CHECKS=0;

	DELETE FROM student
	WHERE student.UNI = UNI;

	DELETE FROM person
	WHERE person.UNI = UNI;
```

```
    SET FOREIGN_KEY_CHECKS=1;
END;;

--
-- Insert students and faculties, using procedures
--
CALL `insert_faculty`('Ferguson', 'Donald', '5', 'professor', 'COMS');;
CALL `insert_faculty`('Ferguson', 'Donale', '4', 'professor', 'EENG');;
CALL `insert_faculty`('Hana', 'Song', '6', 'lecturer', 'COMS');;
CALL `insert_faculty`('Zenyatta', 'Mondatta', '6', 'associate professor', 'COMS');;
CALL `insert_faculty`('Shimada', 'Genji', '1', 'professor', 'EENG');;
CALL `insert_faculty`('Shimada', 'Hanzo', '2', 'professor', 'COMS');;
/*SELECT * from faculty_all_info;*/
CALL `insert_student`('Guo', 'Haoran', 'SEAS', '2017');;
CALL `insert_student`('Guo', 'Haofan', 'GSAS', '2016');;
CALL `insert_student`('Wu', 'Victor', 'GSAS', '2015');;
CALL `insert_student`('Guo', 'Haotian', 'GSAS', '2017');;
CALL `insert_student`('Korol', 'Lucio', 'SEAS', '2014');;
CALL `insert_student`('Amari', 'Tom', 'SEAS', '2017');;
CALL `insert_student`('Guo', 'Jerry', 'SEAS', '2017');;
CALL `insert_student`('Lin', 'Bill', 'SEAS', '2016');;
CALL `insert_student`('Lu', 'Mingfei', 'GSAS', '2016');;
CALL `insert_student`('Mao', 'Zero', 'GSAS', '2016');;
CALL `insert_student`('Hogo', 'Symmetra', 'SEAS', '2016');;
CALL `insert_student`('Lu', 'Mingze', 'SEAS', '2016');;
/*SELECT * FROM student; */

--
-- TRIGGER FOR FACULTY'S ENROLLMENT LIMITS
--
DELIMITER    ;
DROP TRIGGER IF EXISTS `course_faculty_BEFORE_INSERT`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` TRIGGER `course_faculty_BEFORE_INSERT`
BEFORE INSERT ON `course_participant` FOR EACH ROW
BEGIN
    DECLARE faculty_teaching_sections INT;
    IF NEW.role = 'Instructor' THEN
    SELECT COUNT(*) INTO faculty_teaching_sections
    FROM
        course_taught join
        (SELECT
            sections.call_no as `that_section_no`,
            sections.`year` as `that_year`,
            sections.semester as `that_semester`
        FROM
            sections
        WHERE
            sections.call_no = New.section_call_no) AS that_semester
    ON
        (that_semester.`that_year` = course_taught.`year`
        AND
        that_semester.`that_semester`= course_taught.semester);


    IF faculty_teaching_sections>=3 THEN
        SIGNAL SQLSTATE '45100'
```

```
            SET MESSAGE_TEXT = 'OVER 3 SECTIONS TEACHING THIS SEMESTER';
        END IF;
        END IF;
END;;

--
-- TRIGGER FOR STUDENT ENROLLMENT LIMITS- SECTION FULL
--
DELIMITER    ;
DROP TRIGGER IF EXISTS `course_participant_limit_BEFORE_INSERT`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` TRIGGER
`course_participant_limit_BEFORE_INSERT`
BEFORE INSERT ON `course_participant` FOR EACH ROW
BEGIN
        DECLARE enroll_num INT;
        DECLARE enroll_limit INT;

        IF NEW.role != 'Instructor' THEN
        SELECT COUNT(*) INTO enroll_num FROM course_participant
            WHERE course_participant.section_call_no = New.section_call_no;

        SELECT `limit` INTO enroll_limit FROM sections
            WHERE sections.call_no = New.section_call_no;

        IF enroll_num>=enroll_limit THEN
            SIGNAL SQLSTATE '45003'
            SET MESSAGE_TEXT = 'SECTION FULL';
        END IF;
        END IF;
END;;

--
-- TRIGGER FOR STUDENT ENROLLMENT LIMITS- PREREQUESTS
--
DELIMITER    ;
DROP TRIGGER IF EXISTS `course_participant_prereqs_BEFORE_INSERT`;
DELIMITER    ;;
CREATE DEFINER=`root`@`localhost` TRIGGER
`course_participant_prereqs_BEFORE_INSERT`
BEFORE INSERT ON `course_participant` FOR EACH ROW
BEGIN

        DECLARE fulfill tinyint(1);

        IF NEW.role != 'Instructor' THEN
        SET fulfill =
        (SELECT EXISTS(
        SELECT pre_id
        FROM
        (
        SELECT
            sections.course_id as pre_id
        FROM
            sections
        WHERE
            sections.call_no = New.section_call_no
```

```
        ) AS pre_need
        WHERE
            pre_id NOT IN
        (SELECT course_id as pre_id FROM
        (SELECT
            course_taken.course_id as course_id
        FROM
            course_taken
        WHERE
            New.UNI = course_taken.UNI) AS uni_has_taken
        )));

        IF fulfill = 1 THEN
            SIGNAL SQLSTATE '45004'
            SET MESSAGE_TEXT = 'PREREQUESTS NOT FULFILLED';
        END IF;
        END IF;
END;;

--
--SOME TEST DATA
--
/*SELECT * FROM `course_prereqs`;;*/
/*INSERT INTO `course_participant` VALUES('FEDO1','00015','Instructor');; *//*This is a
current section*/
/*SELECT * FROM `course_taught`;;*/
/*INSERT INTO `course_participant` VALUES ('GUHA3','00014','student');;*//*This is a section
which student has not fulfilled the prerequests*/
/*INSERT INTO `course_participant` VALUES ('FEDO1','00003','Instructor');;*//*This professor
has been teaching 3 sections at that semester*/
/*SELECT * FROM `course_taken`;;*/
```

PART2- SOME TESTS

1.VIEW-`course_taken`
A. `course_taken` would correctly select all the courses students have taken **before**.

UNI	course_id	course_name	year	semester
AMTO1	COMSE1006	Intro. to Program for Eng.	2015	3
GUHA1	COMSE1006	Intro. to Program for Eng.	2015	3
AMTO1	COMSE1006	Intro. to Program for Eng.	2015	3
GUHA2	COMSE1006	Intro. to Program for Eng.	2015	3
AMTO1	COMSW3270	Data Structures	2017	3
GUHA1	COMSW3270	Data Structures	2017	3
GUHA2	COMSW3270	Data Structures	2017	3
AMTO1	COMSW3270	Data Structures	2016	4
GUHA2	COMSW3270	Data Structures	2016	4
AMTO1	COMSW4111	Intro. to Databases	2014	3
GUHA1	COMSW4111	Intro. to Databases	2014	3
AMTO1	COMSW4111	Intro. to Databases	2014	3
GUHA1	COMSW4111	Intro. to Databases	2014	3
AMTO1	COMSW4119	Computer Networks	2016	4
GUHA2	COMSW4119	Computer Networks	2016	4

B. After resetting the codes, `course_taken` could also select all the courses students are taking **this semester (2017 semester1)**. Shows that the function `is_current_section` works correctly.

UNI	course_id	course_name	year	semester
AMTO1	COMSW4111	Intro. to Databases	2017	1

2.Select * FROM `course_taught`;

UNI	course_id	course_name	year	semester
HASO1	COMSE1006	Intro. to Program for Eng.	2015	3
FEDO1	COMSW3270	Data Structures	2017	3
FEDO1	COMSW4111	Intro. to Databases	2017	1
FEDO1	COMSW4111	Intro. to Databases	2014	3
FEDO1	COMSW4111	Intro. to Databases	2014	3
FEDO1	COMSW4119	Computer Networks	2016	4

3.SELECT * FROM `faculty_all_info`;

UNI	last_name	first_name	title	Pay_grade	department_code
FEDO1	Ferguson	Donald	professor	5	COMS
FEDO2	Ferguson	Donale	professor	4	EENG
HASO1	Hana	Song	lecturer	6	COMS
SHGE1	Shimada	Genji	professor	1	EENG
SHHA1	Shimada	Hanzo	professor	2	COMS
ZEMO1	Zenyatta	Mondatta	associate profe	6	COMS

4.SELECT * FROM `student_all_info`;

UNI	last_name	first_name	School	Year
GUHA1	Guo	Haoran	SEAS	2017
GUHA2	Guo	Haofan	GSAS	2016
GUHA3	Guo	Haotian	GSAS	2017
GUJE1	Guo	Jerry	SEAS	2017
HOSY1	Hogo	Symmetra	SEAS	2016
KOLU1	Korol	Lucio	SEAS	2014
LIBI1	Lin	Bill	SEAS	2016
LUMI1	Lu	Mingfei	GSAS	2016
LUMI2	Lu	Mingze	SEAS	2016
MAZE1	Mao	Zero	GSAS	2016
WUVI1	Wu	Victor	GSAS	2015

5.CALL `insert_faculty`('Ferguson', 'Donald', '6', 'professor', 'EENG');
SELECT * FROM faculty_all_info;

UNI	last_name	first_name	title	Pay_grade	department_code
FEDO1	Ferguson	Donald	professor	5	COMS
FEDO2	Ferguson	Donale	professor	4	EENG
FEDO3	Ferguson	Donald	professor	6	EENG
HASO1	Hana	Song	lecturer	6	COMS
SHGE1	Shimada	Genji	professor	1	EENG
SHHA1	Shimada	Hanzo	professor	2	COMS
ZEMO1	Zenyatta	Mondatta	associate profe	6	COMS

(Function `generate_uni` works well)

6.CALL `insert_student`('Guo', 'Haoran', 'SEAS', '2017');
SELECT * FROM student_all_info;

UNI	last_name	first_name	School	Year
AMTO1	Amari	Tom	SEAS	2017
GUHA1	Guo	Haoran	SEAS	2017
GUHA2	Guo	Haofan	GSAS	2016
GUHA3	Guo	Haotian	GSAS	2017
GUHA4	Guo	Haoran	SEAS	2017
GUJE1	Guo	Jerry	SEAS	2017
HOSY1	Hogo	Symmetra	SEAS	2016
KOLU1	Korol	Lucio	SEAS	2014
LIBI1	Lin	Bill	SEAS	2016
LUMI1	Lu	Mingfei	GSAS	2016
LUMI2	Lu	Mingze	SEAS	2016
MAZE1	Mao	Zero	GSAS	2016
WUVI1	Wu	Victor	GSAS	2015

7.INSERT INTO `course_participant` VALUES ('FEDO1','00003','Instructor');
/*This professor has been teaching 3 sections at that semester*/

```
1008  10:57:53  INSERT INTO `course_participant` VALUES ('FEDO1','00003','Instructor')  Error Code: 1644. OVER 3 SECTIONS TEACHING THIS SEMESTER
```

8.INSERT INTO `course_participant` VALUES ('GUHA3','00014','student');
/*A full section*/

```
1009  10:59:19  INSERT INTO `course_participant` VALUES ('GUHA3','00014','student')  Error Code: 1644. SECTION FULL
```

9.INSERT INTO `course_participant` VALUES ('GUHA3','00014','student');
/*This is a section which student has not fulfilled the prerequests*/ (Some sample data has been changed for testing)

```
1151  11:01:23  INSERT INTO `course_participant` VALUES ('GUHA3','00014','student')  Error Code: 1644. PREREQUESTS NOT FULFILLED
```

10.CALL `update_faculty`('HASO1', 'FER', 'DON', 7, 'associate professor', 'COMS');
SELECT * FROM `faculty_all_info`;

UNI	last_name	first_name	title	Pay_grade	department_code
FEDO1	Ferguson	Donald	professor	5	COMS
FEDO2	Ferguson	Donale	professor	4	EENG
HASO1	FER	DON	associate profe	7	COMS
SHGE1	Shimada	Genii	professor	1	EENG
SHHA1	Shimada	Hanzo	professor	2	COMS
ZEMO1	Zenvatta	Mondatta	associate profe	6	COMS

11.CALL `update_student`('AMTO1', 'GUO', 'VICTOR', 'GASA', '2017');
SELECT * FROM `student_all_info`;

UNI	last_name	first_name	School	Year
AMTO1	GUO	VICTOR	GASA	2017
GUHA1	Guo	Haoran	SEAS	2017
GUHA2	Guo	Haofan	GSAS	2016
GUHA3	Guo	Haotian	GSAS	2017
GUJE1	Guo	Jerry	SEAS	2017
HOSY1	Hogo	Symmetra	SEAS	2016
KOLU1	Korol	Lucio	SEAS	2014

12.CALL `delete_faculty`('FEDO2');
SELECT * FROM `faculty_all_info`;

UNI	last_name	first_name	title	Pay_grade	department_code
FEDO1	Ferguson	Donald	professor	5	COMS
HASO1	Hana	Song	lecturer	6	COMS
SHGE1	Shimada	Genii	professor	1	EENG
SHHA1	Shimada	Hanzo	professor	2	COMS
ZEMO1	Zenvatta	Mondatta	associate profe	6	COMS

13.CALL `delete_student`('GUHA2');
SELECT * FROM `student_all_info`;

UNI	last_name	first_name	School	Year
AMTO1	Amari	Tom	SEAS	2017
GUHA1	Guo	Haoran	SEAS	2017
GUHA3	Guo	Haotian	GSAS	2017
GUJE1	Guo	Jerry	SEAS	2017
HOSY1	Hogo	Symmetra	SEAS	2016
KOLU1	Korol	Lucio	SEAS	2014
LIBI1	Lin	Bill	SEAS	2016

14.(Test function `is_current_section`)

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `checktime`()
BEGIN
    SELECT `is_current_section`(2017,'1');
END;
```

CALL `checktime`;

`is_current_section`(2017,'1')
1

`is_current_section`(2016,'4')
0

`is_current_section`(2017,'3')
0

`is_current_section`(2016,'1')
0