Title of My Project

University Management System

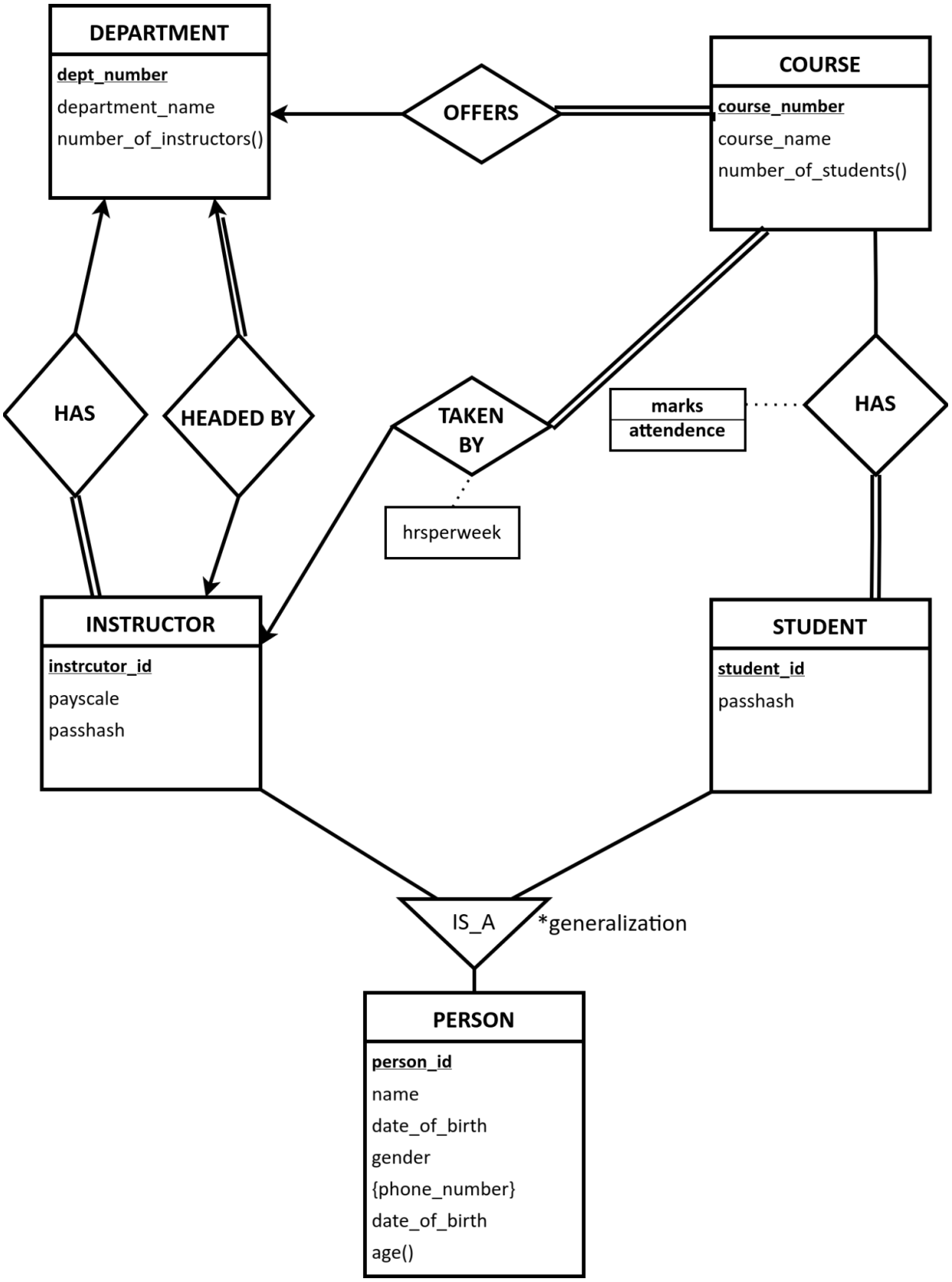
Prepared by:
Dhrumilkumar Patel
21CEUOS031
CE028
Computer Engineering,
Dharmsinh Desai University

table of Content

Brief information about the system

1. College has several departments and instructors. Several instructors can work in one department.
2. An instructor can work in only one department.
3. Every department has a head who is an instructor. An instructor can head only one department.
4. Each department can offer any number of courses. An instructor can only take a course offered by his department.
5. Each instructor can take any number of courses. A course can be taken by only one instructor.
6. College has several students. A student can enroll for any number of courses. Each course can have any number of students.
7. Every department has a unique department_number and dept_name .
8. Every course has a globally unique course_number and course_name . A person in general (instructor or student) has a First Name , Middle Name (optional), Last Name (optional), Date of Birth , and Gender .
9. A person may have one or more Phone Numbers .
10. Every instructor, in addition, has a unique instructor_id and payscale . Every student, in addition, has a unique student_id .
11. Both instructors and students have a saved Password Hash required for login.
12. Every course has a count of the hours_per_week it is taken by its instructor.
13. Every course associated with a student has a record of the Attendance and Marks obtained by the student in that course.

ER Diagram

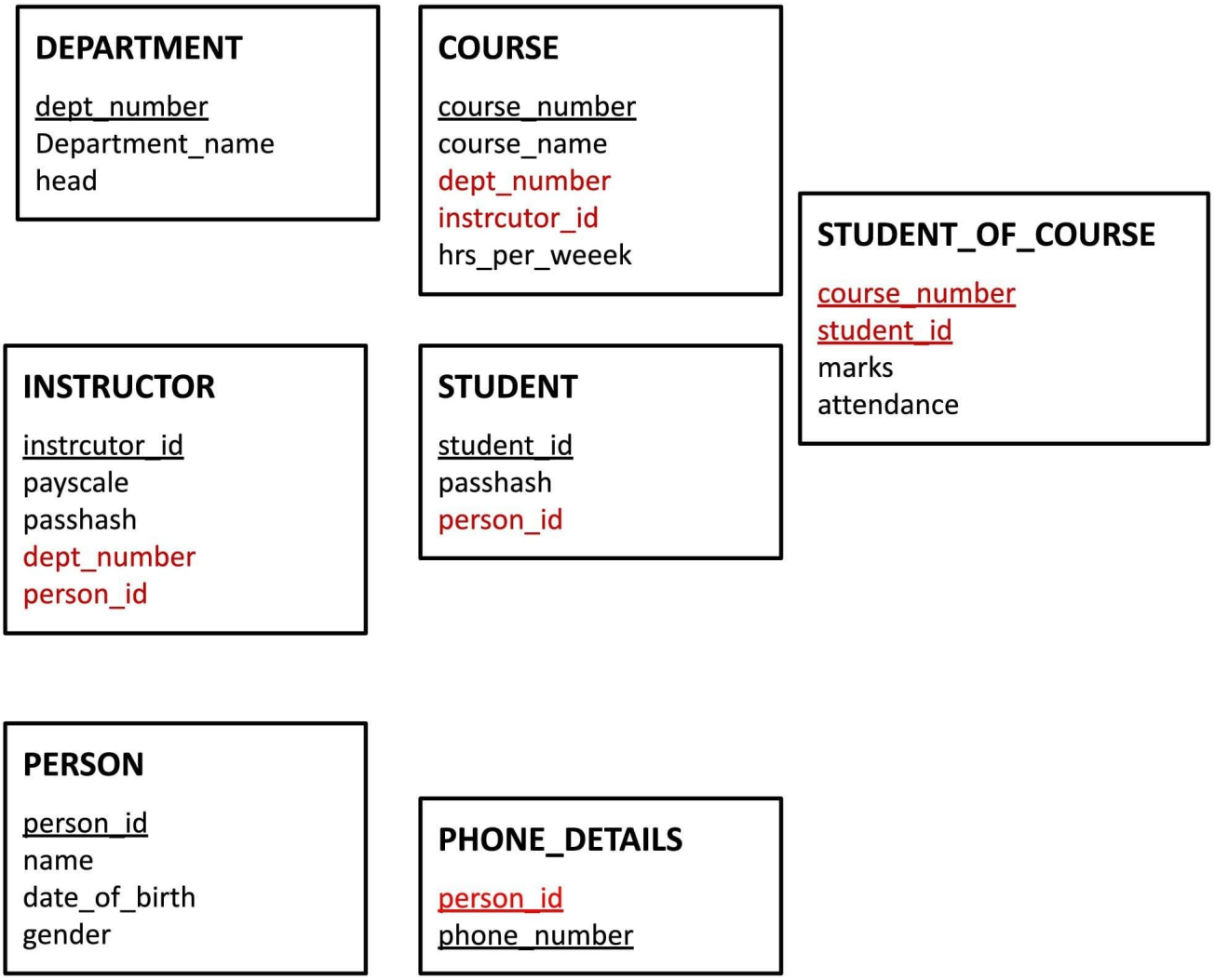

DEPARTMENT
dept_number
department_name
number_of_instructors()
OFFERS
COURSE
course_number
course_name
number_of_students()
HAS
HEADED BY
TAKEN BY
marks
attendence
HAS
hrsperweek
INSTRUCTOR
instrcutor_id
payscale
passhash
STUDENT
student_id
passhash
IS_A
*generalization
PERSON
person_id
name
date_of_birth
gender
{phone_number}
date_of_birth
age()

Schema before Normalization

First of all we are removing all three derivable attributes.

And now preparing the schema as follows, all the points regarding conversion of ER diagram to schema are listed after this page.

underlined attributes are representing foreign keys, red attributes are representing foreign keys.



- **Some notes About schema formation**

1. we are having following relationships,

RELATIONSHIP	CARDINALITY	SEPARATE RELATION REQUIRED?
DEPARTMENT HAS INSTRUCTOR	ONE TO MANY	NO
DEPARTMENT HEADED BY INSTRUCTOR	ONE TO ONE	NO
DEPARTMENT OFFERS COURSE	ONE TO MANY	NO
COURSE TAKEN BY INSTRUCTOR	MANY TO ONE	NO
COURSE HAS STUDENT	MANY TO MANY	**YES**

2. For conversion of the last relationship we will need a relation having a primary key as a combination of both COURSE and STUDENT.

3. We have one multivalued attribute for which we have to make different relations in which both person_id and phone_number combined will form the primary key.

Normalization comments (optional)

- ## First Normal Form

    1. In relation person ***name*** attribute is not atomic so,we will divide it into three subparts: ***first_name***,***middle_name*** and ***last_name***.
    2. In relation person relation ***phone_number*** can be multivalued so we will make a separate schema for that attribute.

    Now in all relations, every attribute is atomic and takes only one value from its domain. Hence all are in their First Normal Form (1NF).

- ## Second Normal Form

    Super keys of the relations are as:

relation	**super key**
DEPARTMENT	*dept_number*
PERSON	*person_id*
PHONE_DETAILS	*person_id,phone_number*
INSTRUCTOR	*instructor_id*
COURSE	*course_number*
STUDENT	*student_id*
STUDENT_OF_COURSE	*student_id,course_number*

In relation **STUDENT_OF_COURSE** , the functional dependencies are:

student_id, course_number → *Attendance*

student_id, course_number → *Marks*

Removing either student_id or course_number cannot identify the Attendance and Marks uniquely. Hence, the super key is the candidate key. Therefore, the relation is in its **Second Normal Form (2NF).**

same for relation **PHONE_DETAILS.**

Now in all other relations there is only one attribute forming the super key. Hence, in all such relations, the super key is the candidate key. Therefore, the relations are in their **Second Normal Form (2NF).**

- ## Third Normal Form for

The functional dependencies are:

DEPARTMENT

dept_number → *dept_number*

dept_number → *department_name*

dept_number → *head*

department_name →*head*

We can remove the transitive dependency

dept_number→ *department_name* → *head*

Hence, new FDs are:

dept_number → *dept_number*

dept_number → *department_name*

dept_number → *head*

To convert relation into Third Normal Form (3NF), it should not have any transitive functional dependencies. As there is no transitive dependency we can say that the above relation is in Third Normal Form (3NF).

*we will make another relation HEAD for this.

PERSON

person_id → *person_id*

person_id → *first_name*

person_id → *middle_name*

person_id → *last_name*

person_id → *date_of_birth*

person_id → *gender*

To convert relation into Third Normal Form (3NF), it should not have any transitive functional dependencies. As there is no transitive dependency we can say that the above relation is in Third Normal Form (3NF).

PHONE_DETAILS

person_id,phone_number → *person_id*

person_id ,phone_number → *phone_number*

To convert relation into Third Normal Form (3NF), it should not have any transitive functional dependencies. As there is no transitive dependency we can say that the above relation is in Third Normal Form (3NF).

INSTRUCTOR

instructor_id → *instructor_id*

instructor_id → *passhash*

instructor_id → *person_id*

instructor_id → *payscale*

instructor_id → *dept_number*

To convert relation into Third Normal Form (3NF), it should not have any transitive functional dependencies. As there is no transitive dependency we can say that the above relation is in Third Normal Form (3NF).

COURSE

course_number → *course_number*

course_number → *course_name*

course_number → dept_number

course_number → instructor_id

course_number → hrs_per_week

To convert a relation into Third Normal Form (3NF), it should not have any transitive functional dependencies. As there is no transitive dependency we can say that the above relation is in Third Normal Form (3NF).

STUDENT

student_id → student_id

student_id → passhash

student_id → person_id

To convert relation into Third Normal Form (3NF), it should not have any transitive functional dependencies. As there is no transitive dependency we can say that the above relation is in Third Normal Form (3NF).

STUDENT_OF_COURSE

student_id,course_number → student_id, course_number

student_id,course_number → attendance

student_id,course_number → marks

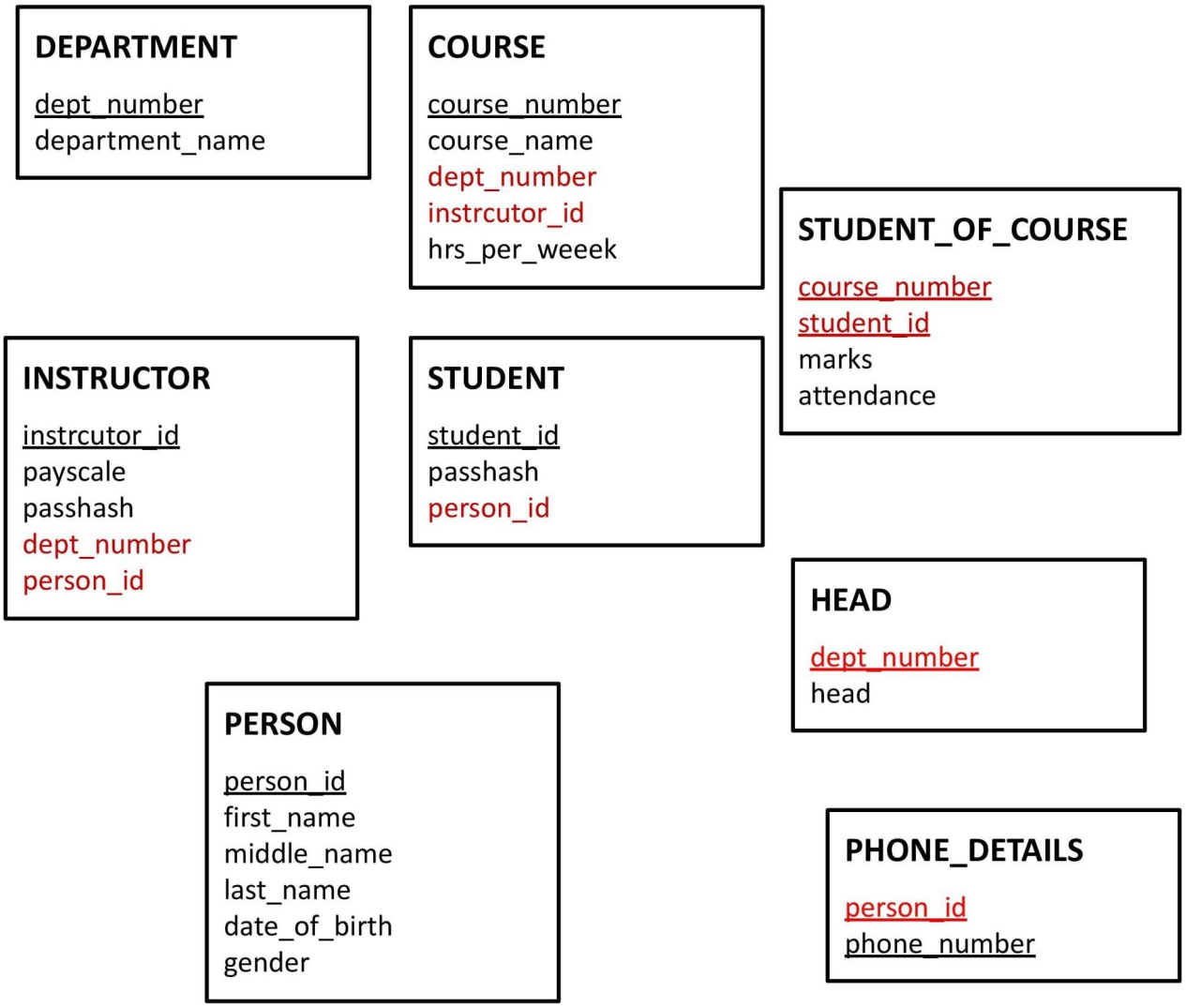
To convert relation into Third Normal Form (3NF), it should not have any transitive functional dependencies. As there is no transitive dependency we can say that the above relation is in Third Normal Form (3NF).

- ## Boyce-Codd Normal Form

In all relations, for all functional dependencies, the L.H.S. is a super key. Hence, all

relations are in their Boyce-Codd Normal Form (BCNF).

schema after applying normalization



CRUD

1. create

   INSERT INTO `STUDENT` (`student_id`, `passhash`, `person_id`) VALUES ('1', 'hell@1', '1');

   INSERT INTO `DEPARTMENT` (`dept_number`, `department_name`) VALUES ('1', 'computer_engineering');

   INSERT INTO `COURSE` (`course_number`, `course_name`, `dept_number`, `instructor_id`, `hrs_per_week`) VALUES ('1', 'dbms', '1', '1', '6');

   INSERT INTO `INSTRUCTOR` (`instructor_id`, `payscale`, `passhash`, `dept_number`, `person_id`) VALUES ('1', '100000', 'abc@123', '1', '1');

   INSERT INTO `PHONE_DETAILS` (`person_id`, `phone_number`) VALUES ('1', '9999955555');

   INSERT INTO `PERSON` (`person_id`, `first_name`, `middle_name`, `last_name`, `date_of_birth`, `gender`) VALUES ('1', 'dhrumil', 'bharatbhai', 'patel', '2004-07-30', 'male');

   INSERT INTO `HEAD` (`dept_number`, `head`) VALUES ('1', '1');

2. read

   SELECT `student_id`, `passhash`, `person_id` FROM `STUDENT` WHERE 1

   SELECT `dept_number`, `department_name` FROM `DEPARTMENT` WHERE 1

   SELECT `course_number`, `course_name`, `dept_number`, `instructor_id`, `hrs_per_week` FROM `COURSE` WHERE 1

   SELECT `instructor_id`, `payscale`, `passhash`, `dept_number`, `person_id` FROM `INSTRUCTOR` WHERE 1

   SELECT `person_id`, `first_name`, `middle_name`, `last_name`, `date_of_birth`, `gender` FROM `PERSON` WHERE 1

3. update

   UPDATE `INSTRUCTOR` SET `person_id`=2 WHERE `person_id`=1;

   UPDATE `HEAD` SET `head`=2 WHERE `dept_number`= 1;

   UPDATE `INSTRUCTOR` SET `payscale`=200000,`passhash`='adada1' WHERE `instructor_id`=2;

UPDATE `PERSON` SET `date_of_birth` = '2003-04-20' WHERE `PERSON`.`person_id` = 2;

UPDATE `PHONE_DETAILS` SET `phone_number`='9999955554' WHERE `person_id`=1;

4. delete

DELETE FROM INSTRUCTOR WHERE `INSTRUCTOR`.`instructor_id` = 2

DELETE FROM COURSE WHERE `COURSE`.`course_number` = 1

DELETE FROM INSTRUCTOR WHERE `INSTRUCTOR`.`instructor_id` = 1

SQL to create database

```
CREATE TABLE COURSE (
  course_number int(11) NOT NULL,
  course_name varchar(20) NOT NULL,
  dept_number int(11) NOT NULL,
  instructor_id int(11) NOT NULL,
  hrs_per_week int(11) NOT NULL
)
CREATE TABLE DEPARTMENT (
  dept_number int(11) NOT NULL,
  department_name varchar(20) NOT NULL
)
INSERT INTO DEPARTMENT (dept_number, department_name) VALUES
(1, 'computer_engineering');

CREATE TABLE HEAD (
  dept_number int(11) NOT NULL,
```

```
  head int(11) NOT NULL
)
INSERT INTO HEAD (dept_number, head) VALUES
(1, 2);

CREATE TABLE INSTRUCTOR (
  instructor_id int(11) NOT NULL,
  payscale int(11) NOT NULL,
  passhash varchar(15) NOT NULL,
  dept_number int(11) NOT NULL,
  person_id int(11) NOT NULL
)
INSERT INTO INSTRUCTOR (instructor_id, payscale, passhash, dept_number, person_id) VALUES
(1, 100000, 'abc@123', 1, 2);

CREATE TABLE PERSON (
  person_id int(11) NOT NULL,
  first_name varchar(20) NOT NULL,
  middle_name varchar(20) NOT NULL,
  last_name varchar(20) NOT NULL,
  date_of_birth date NOT NULL,
  gender varchar(10) NOT NULL
)
INSERT INTO PERSON (person_id, first_name, middle_name, last_name, date_of_birth, gender)
VALUES
(1, 'dhrumil', 'bharatbhai', 'patel', '2004-07-30', 'male'),
```

```
(2, 'harsh', '', 'gajera', '2003-04-20', 'male'),
(3, 'abc', 'pqr', 'xyz', '1999-12-12', 'male');

CREATE TABLE PHONE_DETAILS (
  person_id int(11) NOT NULL,
  phone_number varchar(20) NOT NULL
)
INSERT INTO PHONE_DETAILS (person_id, phone_number) VALUES
(1, '9999955554');

CREATE TABLE STUDENT (
  student_id int(11) NOT NULL,
  passhash varchar(15) NOT NULL,
  person_id int(11) NOT NULL
)
INSERT INTO STUDENT (student_id, passhash, person_id) VALUES
(1, 'hell@1', 1);

CREATE TABLE STUDENT_OF_COURSE (
  course_number int(11) NOT NULL,
  student_id int(11) NOT NULL,
  marks int(11) NOT NULL,
  attendance int(11) NOT NULL
)
```

```
ALTER TABLE COURSE
  ADD PRIMARY KEY (course_number),
  ADD UNIQUE KEY dept_number (dept_number,instructor_id),
  ADD KEY instructor_id (instructor_id);

ALTER TABLE DEPARTMENT
  ADD PRIMARY KEY (dept_number);

ALTER TABLE HEAD
  ADD PRIMARY KEY (dept_number);

ALTER TABLE INSTRUCTOR
  ADD PRIMARY KEY (instructor_id),
  ADD KEY dept_number (dept_number),
  ADD KEY person_id (person_id);

ALTER TABLE PERSON
  ADD PRIMARY KEY (person_id);

ALTER TABLE PHONE_DETAILS
  ADD KEY person_id (person_id);

ALTER TABLE STUDENT
  ADD PRIMARY KEY (student_id),
  ADD KEY person_id (person_id);
```

```
ALTER TABLE STUDENT_OF_COURSE
  ADD KEY course_number (course_number),
  ADD KEY student_id (student_id);

ALTER TABLE COURSE
  ADD CONSTRAINT COURSE_ibfk_1 FOREIGN KEY (dept_number) REFERENCES DEPARTMENT
(dept_number),
  ADD CONSTRAINT COURSE_ibfk_2 FOREIGN KEY (instructor_id) REFERENCES INSTRUCTOR
(instructor_id);

ALTER TABLE HEAD
  ADD CONSTRAINT HEAD_ibfk_1 FOREIGN KEY (dept_number) REFERENCES DEPARTMENT
(dept_number) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE INSTRUCTOR
  ADD CONSTRAINT INSTRUCTOR_ibfk_1 FOREIGN KEY (dept_number) REFERENCES
DEPARTMENT (dept_number),
  ADD CONSTRAINT INSTRUCTOR_ibfk_2 FOREIGN KEY (person_id) REFERENCES PERSON
(person_id);

ALTER TABLE PHONE_DETAILS
  ADD CONSTRAINT PHONE_DETAILS_ibfk_1 FOREIGN KEY (person_id) REFERENCES PERSON
(person_id);

ALTER TABLE STUDENT
```

```
  ADD CONSTRAINT STUDENT_ibfk_1 FOREIGN KEY (person_id) REFERENCES PERSON
(person_id);


ALTER TABLE STUDENT_OF_COURSE

  ADD CONSTRAINT STUDENT_OF_COURSE_ibfk_1 FOREIGN KEY (course_number) REFERENCES
COURSE (course_number),

  ADD CONSTRAINT STUDENT_OF_COURSE_ibfk_2 FOREIGN KEY (student_id) REFERENCES
STUDENT (student_id);

COMMIT;
```

References

1. “Data Base System Concepts”, Henry F. Korth and A. slberschatz 2 nd Edition, McGraw-Hill 1991.
2. https://www.geeksforgeeks.org/normal-forms-in-dbms/
3. https://docs.oracle.com/database/121/COMSC/diagrams.htm
4. https://www.geeksforgeeks.org/introduction-of-er-model/