Will Richmond & Mason Shepherd

CS2243

Dr. Wang

22 April 2018

Database Project

Q1:

**Primary Keys:**

course: COURSE\_ID

course\_section: C\_SEC\_ID

enrollment: S\_ID, C\_SEC\_ID – both are needed as the primary key, see data.

faculty: F\_ID

location: LOC\_ID

student: S\_ID

term: TERM\_ID

**Foreign Keys:**

course\_section: COURSE\_ID(course), TERM\_ID(term), F\_ID(faculty), LOC\_ID(location)

enrollment: S\_ID(student), C\_SEC\_ID(course\_section) – these two FK make up the PK

Brief Justification: Upon update or deletion, we have set the action to “RESTRICT”; the decision was made to prevent foreign keys from being updated, as they are all primary keys of another table.

**Create Table Statements:**

CREATE TABLE `course` (

`COURSE\_ID` int(11) NOT NULL,

`COURSE\_NO` varchar(45) DEFAULT NULL,

`COURSE\_NAME` varchar(45) DEFAULT NULL,

`CREDITS` int(11) DEFAULT NULL,

PRIMARY KEY (`COURSE\_ID`),

UNIQUE KEY `COURSE\_ID\_UNIQUE` (`COURSE\_ID`)

)

CREATE TABLE `course\_section` (

`C\_SEC\_ID` int(11) NOT NULL,

`COURSE\_ID` int(11) DEFAULT NULL,

`TERM\_ID` int(11) DEFAULT NULL,

`SEC\_NUM` int(11) DEFAULT NULL,

`F\_ID` int(11) DEFAULT NULL,

`MTG\_DAYS` varchar(45) DEFAULT NULL,

`START\_TIME` varchar(45) DEFAULT NULL,

`END\_TIME` varchar(45) DEFAULT NULL,

`LOC\_ID` int(11) DEFAULT NULL,

`MAX\_ENRL` int(11) DEFAULT NULL,

PRIMARY KEY (`C\_SEC\_ID`)

)

CREATE TABLE `enrollment` (

`S\_ID` int(11) NOT NULL,

`C\_SEC\_ID` int(11) NOT NULL,

`GRADE` char(1) DEFAULT NULL,

PRIMARY KEY (`S\_ID`,`C\_SEC\_ID`)

)

CREATE TABLE `faculty` (

`F\_ID` int(11) NOT NULL,

`F\_LAST` varchar(45) DEFAULT NULL,

`F\_FIRST` varchar(45) DEFAULT NULL,

`F\_MI` char(1) DEFAULT NULL,

`LOC\_ID` int(11) DEFAULT NULL,

`F\_PHONE` varchar(10) DEFAULT NULL,

`F\_RANK` varchar(45) DEFAULT NULL,

`F\_SUPER` int(11) DEFAULT NULL,

`F\_PIN` varchar(4) DEFAULT NULL,

PRIMARY KEY (`F\_ID`),

UN

IQUE KEY `F\_ID\_UNIQUE` (`F\_ID`),

UNIQUE KEY `F\_PIN\_UNIQUE` (`F\_PIN`)

)

CREATE TABLE `location` (

`LOC\_ID` int(11) NOT NULL,

`BLDG\_CODE` varchar(45) DEFAULT NULL,

`ROOM` varchar(45) DEFAULT NULL,

`CAPACITY` int(11) DEFAULT NULL,

PRIMARY KEY (`LOC\_ID`)

)

CREATE TABLE `student` (

`S\_ID` int(11) NOT NULL,

`S\_LAST` varchar(45) DEFAULT NULL,

`S\_FIRST` varchar(45) DEFAULT NULL,

`S\_MI` char(1) DEFAULT NULL,

`S\_ADDRESS` varchar(45) DEFAULT NULL,

`S\_CITY` varchar(45) DEFAULT NULL,

`S\_STATE` varchar(2) DEFAULT NULL,

`S\_ZIP` varchar(5) DEFAULT NULL,

`S\_PHONE` varchar(10) DEFAULT NULL,

`S\_CLASS` varchar(2) DEFAULT NULL,

`S\_DOB` date DEFAULT NULL,

`S\_PIN` varchar(4) DEFAULT NULL,

`F\_ID` int(11) DEFAULT NULL,

`DATE\_ENROLLED` varchar(45) DEFAULT NULL,

PRIMARY KEY (`S\_ID`),

UNIQUE KEY `S\_ID\_UNIQUE` (`S\_ID`)

)

CREATE TABLE `term` (

`TERM\_ID` int(11) NOT NULL,

`TERM\_DESC` varchar(45) DEFAULT NULL,

`STATUS` varchar(45) DEFAULT NULL,

`START\_DATE` varchar(45) DEFAULT NULL,

PRIMARY KEY (`TERM\_ID`)

)

Q2: To import the data into our database, we used a built-in import wizard in Microsoft Excel to quickly append our relations with new data. After mapping the columns, MySQL took care of the rest, instead of needlessly writing INSERT INTO commands.

(Mason) I converted the excel files into CSVs to use a mechanism already in MySQL for import. Essentially the same thing Will did.

Q3:

a.

i. 19:37:16 INSERT INTO course\_section VALUES (12, 2, 6, 2, 2, “MTWRF”, 10:00 AM, 11:30 AM, 5, 35) Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ':00 AM, 11:30 AM, 5, 35)' at line 2 0.000 sec

ii. 19:38:46 INSERT INTO course\_section VALUES (12, 2, 6, 2, 2, “MTWRF”, 9:00 AM, 10:30 AM, 6, 35) Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ':00 AM, 10:30 AM, 6, 35)' at line 2 0.000 sec

iii. 19:39:31 INSERT INTO course\_section VALUES (2, 1, 4, 2, 3, “TR”, 9:30 AM, 10:45 AM, 4, 35) Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ':30 AM, 10:45 AM, 4, 35)' at line 2 0.000 sec

b.

i. 19:40:33 INSERT INTO faculty VALUES (4, “Brown”, “Colin”, “D”, 11, “3253456789”, “Assistant”, 4, 9871) Error Code: 1054. Unknown column '“Brown”' in 'field list' 0.031 sec //F\_ID 4 already exists – must be unique

ii. 19:41:19 INSERT INTO faculty VALUES (6, “Reeves”, “Bob”, “S”, 15, “3256789012”, “Full”, , 1234) Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ' 1234)' at line 2 0.000 sec //no LOC\_ID 15

iii. 19:42:02 INSERT INTO faculty VALUES (6, “Reeves”, “Bob”, “S”, 10, “3256789012”, “Assistant”, 7, 1234) Error Code: 1054. Unknown column '“Reeves”' in 'field list' 0.000 sec //no F\_SUPER = 7

iv. 19:42:41 INSERT INTO faculty VALUES (6, “Reeves”, “Bob”, “S”, 10, “3255678901”, “Assistant”, 2, 1234) Error Code: 1054. Unknown column '“Reeves”' in 'field list' 0.000 sec //SSN must be unique

c. 19:43:43 INSERT INTO course VALUES (4, “CS 120”, “Intro. to Programming in C++”, 3) Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '120”, “Intro. to Programming in C++”, 3)' at line 2 0.000 sec //course\_id 4 already exists

d. 19:49:43 DELETE FROM location WHERE (LOC\_ID=11 AND BLDG\_CODE=“BUS” AND ROOM=“222” AND CAPACITY=1) Error Code: 1054. Unknown column '“BUS”' in 'where clause' 0.015 sec //tuple does not exist

e. 19:54:06 DELETE FROM term WHERE (TERM\_ID=4 AND TERM\_DESC=“Fall 2007” AND STATUS=“CLOSED” AND START\_DATE=“28-AUG-07”) Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '2007” AND STATUS=“CLOSED” AND START\_DATE=“28-AUG-07”)' at line 2 0.000 sec //must use mm-dd-yyyy format

Q4:

a. SELECT S\_ID, S\_First, S\_Last

FROM STUDENT

WHERE S\_ID IN

(SELECT S\_ID

FROM ENROLLMENT

WHERE GRADE IN ("A", "B"));

OUTPUT:

S\_ID S\_FIRST S\_LAST

1 Tammy Jones

2 Jorge Perez

5 Liza Johnson

b. SELECT \*

FROM TERM

WHERE TERM\_DESC LIKE ('%07');

OUTPUT:

TERM\_ID TERM\_DESC STATUS START\_DATE

2 Spring 2007 CLOSED 01/09/08

3 Summer 2007 CLOSED 05/15/06

4 Fall 2007 CLOSED 08/28/07

c. SELECT BLDG\_CODE, ROOM, CAPACITY

FROM LOCATION

ORDER BY BLDG\_CODE, ROOM ASC;

OUTPUT:

BLDG\_CODE ROOM CAPACITY

BUS 105 42

BUS 211 55

BUS 402 1

BUS 404 35

BUS 421 35

BUS 424 1

BUS 433 1

CR 101 150

CR 103 35

CR 105 35

CR 202 40

LIB 217 2

LIB 222 1

d. SELECT COURSE\_NO, COURSE\_NAME, CREDITS \* 720

FROM COURSE;

OUTPUT:

COURSE\_NO COURSE\_NAME CREDITS \* 720

IT 101 Intro. to Info.Tech. 2160

IS 301 Systems Analysis 2160

IT 240 Intro. to Database Systems 2160

CS 120 Intro. To Programming in C++ 2160

IT 451 Web-Based Systems 2160

e.

SELECT COUNT(C\_SEC\_ID), AVG(C\_SEC\_ID), MAX(C\_SEC\_ID), MIN(C\_SEC\_ID)

FROM ENROLLMENT

WHERE C\_SEC\_ID IN (

SELECT C\_SEC\_ID

FROM COURSE\_SECTION

WHERE TERM\_ID IN (

SELECT TERM\_ID

FROM TERM

WHERE TERM\_DESC = 'Summer 2008'));

OUTPUT:

COUNT(C\_SEC\_ID) AVG(C\_SEC\_ID) MAX(C\_SEC\_ID) MIN(C\_SEC\_ID)

8 11.8750 13 11

f. SELECT COUNT(GRADE)

FROM ENROLLMENT

WHERE ((GRADE LIKE 'A') OR (GRADE LIKE 'B') OR (GRADE LIKE 'C') OR (GRADE LIKE 'D'))

AND (S\_ID IN(

SELECT S\_ID

FROM STUDENT

WHERE (S\_FIRST LIKE 'Lisa') AND (S\_LAST LIKE 'Johnson')));

OUTPUT:

COUNT(GRADE)

3

g. SELECT BLDG\_CODE, CAPACITY

FROM LOCATION

WHERE (CAPACITY > 100)

GROUP BY CAPACITY;

OUTPUT:

BLDG\_CODE CAPACITY

CR 150

h. SELECT STUDENT.S\_ID, STUDENT.S\_LAST, STUDENT.S\_FIRST, STUDENT.F\_ID, FACULTY.F\_LAST

FROM STUDENT

INNER JOIN FACULTY ON STUDENT.F\_ID = FACULTY.F\_ID;

OUTPUT:

S\_ID S\_LAST S\_FIRST F\_ID F\_LAST

1 Jones Tammy 1 Marx

2 Perez Jorge 1 Marx

3 Marsh John 1 Marx

4 Smith Mike 2 Zhulin

5 Johnson Lisa 4 Brown

6 Nguyen Ni 3 Langley

i. SELECT F\_LAST

FROM FACULTY

WHERE F\_ID IN (

SELECT F\_ID

FROM COURSE\_SECTION

WHERE TERM\_ID IN (

SELECT TERM\_ID

FROM TERM

WHERE TERM\_DESC IN ("Summer 2008")));

OUTPUT:

F\_LAST

Marx

Zhulin

Langley

j. SELECT lipCOURSE.COURSE\_NO, lipCOURSE.COURSE\_NAME, lipEnroll.GRADE

FROM ENROLLMENT AS lipEnroll

INNER JOIN COURSE\_SECTION AS COURSESection ON lipEnroll.C\_SEC\_ID = COURSESection.C\_SEC\_ID

INNER JOIN COURSE AS lipCOURSE ON COURSESection.COURSE\_ID = lipCOURSE.COURSE\_ID

WHERE S\_ID IN (

SELECT S\_ID

FROM STUDENT

WHERE (S\_FIRST LIKE 'Tammy') AND (S\_LAST LIKE 'Jones'));

OUTPUT:

COURSE\_NO COURSE\_NAME GRADE

IT 101 Intro. to Info.Tech. A

IS 301 Systems Analysis A

IT 240 Intro. to Database Systems B

IT 451 Web-Based Systems B

k. SELECT S\_LAST, S\_FIRST, S\_PHONE FROM STUDENT

UNION

SELECT F\_LAST, F\_FIRST, F\_PHONE FROM FACULTY;

OUTPUT:

S\_LAST S\_FIRST S\_PHONE

Jones Tammy 3250987654

Perez Jorge 3258765432

Marsh John 3257654321

Smith Mike 3256543210

Johnson Lisa 3255432109

Nguyen Ni 3254321098

Marx Teresa 3251234567

Zhulin Mark 3252345678

Langley Colin 3253456789

Brown Jonnel 3254567890

Sealy James 3255678901

Q5:

a. SELECT S\_FIRST, S\_LAST

FROM STUDENT

WHERE S\_CLASS IN (

SELECT S\_CLASS

FROM STUDENT

WHERE (S\_FIRST) IN ('Jorge') AND (S\_LAST) IN ('Perez'));

OUTPUT:

S\_FIRST S\_LAST

Tammy Jones

Jorge Perez

b. SELECT DISTINCT STUDENT.S\_FIRST, STUDENT.S\_LAST

FROM STUDENT

INNER JOIN ENROLLMENT ON STUDENT.S\_ID = ENROLLMENT.S\_ID

WHERE ENROLLMENT.C\_SEC\_ID = 1

OR ENROLLMENT.C\_SEC\_ID = 5

OR ENROLLMENT.C\_SEC\_ID = 6

OR ENROLLMENT.C\_SEC\_ID = 9;

OUTPUT:

S\_FIRST S\_LAST

Tammy Jones

Jorge Perez

John Marsh

Lisa Johnson

c. SELECT DISTINCT STUDENT.S\_FIRST, STUDENT.S\_LAST

FROM STUDENT

INNER JOIN ENROLLMENT ON STUDENT.S\_ID = ENROLLMENT.S\_ID

WHERE S\_CLASS IN (

SELECT S\_CLASS

FROM STUDENT

WHERE (S\_FIRST) IN ('Jorge') AND (S\_LAST) IN ('Perez')

AND (ENROLLMENT.C\_SEC\_ID = 1

OR ENROLLMENT.C\_SEC\_ID = 5

OR ENROLLMENT.C\_SEC\_ID = 6

OR ENROLLMENT.C\_SEC\_ID = 9));

OUTPUT:

S\_FIRST S\_LAST

Tammy Jones

Jorge Perez

d. SELECT STUDENT.S\_FIRST, STUDENT.S\_LAST

FROM STUDENT

INNER JOIN ENROLLMENT ON STUDENT.S\_ID = ENROLLMENT.S\_ID

INNER JOIN COURSE\_SECTION ON ENROLLMENT.C\_SEC\_ID = COURSE\_SECTION.C\_SEC\_ID

INNER JOIN LOCATION ON COURSE\_SECTION.LOC\_ID = LOCATION.LOC\_ID

WHERE LOCATION.LOC\_ID IN(

SELECT LOC\_ID

FROM LOCATION

WHERE BLDG\_CODE = ("CR")

AND STUDENT.S\_CLASS IN (

SELECT S\_CLASS

FROM STUDENT

WHERE (S\_FIRST) IN ('Jorge') AND (S\_LAST) IN ('Perez')));

OUTPUT:

S\_FIRST S\_LAST

Tammy Jones

Jorge Perez

e. SELECT COURSE\_NAME

FROM COURSE, COURSE\_SECTION

WHERE COURSE\_SECTION.TERM\_ID = 6

AND COURSE.COURSE\_ID = COURSE\_SECTION.COURSE\_ID

UNION

SELECT COURSE\_NAME

FROM COURSE AS A,

COURSE\_SECTION AS B,

STUDENT AS C,

ENROLLMENT AS D

WHERE A.COURSE\_ID = B.COURSE\_ID

AND B.C\_SEC\_ID = D.C\_SEC\_ID

AND C.S\_ID = D.S\_ID

AND C.S\_CLASS<>"SR";

OUTPUT:

COURSE\_NAME

Intro. to Info.Tech.

Systems Analysis

Intro. to Database Systems

Web-Based Systems

f. SELECT DISTINCT COURSE\_NAME

FROM COURSE AS A

JOIN COURSE\_SECTION AS B ON A.COURSE\_ID = B.COURSE\_ID

JOIN ENROLLMENT AS C ON B.C\_SEC\_ID = C.C\_SEC\_ID

JOIN STUDENT AS D ON C.S\_ID = D.S\_ID

WHERE S\_CLASS<>"SR" OR COURSE\_NAME IN(

SELECT COURSE\_NAME FROM COURSE AS A

JOIN COURSE\_SECTION AS B ON A.COURSE\_ID = B.COURSE\_ID

JOIN ENROLLMENT AS C ON B.C\_SEC\_ID = C.C\_SEC\_ID

JOIN STUDENT AS D ON C.S\_ID = D.S\_ID

WHERE TERM\_ID = 6);

OUTPUT:

COURSE\_NAME

Intro. to Info.Tech.

Systems Analysis

Intro. to Database Systems

Web-Based Systems

g. SELECT DISTINCT COURSE\_NAME

FROM COURSE AS A

JOIN COURSE\_SECTION AS B ON A.COURSE\_ID = B.COURSE\_ID

JOIN ENROLLMENT AS C ON B.C\_SEC\_ID = C.C\_SEC\_ID

JOIN STUDENT AS D ON C.S\_ID = D.S\_ID

WHERE S\_CLASS<>"SR" AND COURSE\_NAME NOT IN(

SELECT COURSE\_NAME FROM COURSE AS A

JOIN COURSE\_SECTION AS B ON A.COURSE\_ID = B.COURSE\_ID

JOIN ENROLLMENT AS C ON B.C\_SEC\_ID = C.C\_SEC\_ID

JOIN STUDENT AS D ON C.S\_ID = D.S\_ID

WHERE TERM\_ID = 6);

OUTPUT:

COURSE\_NAME

Web-Based Systems

h. SELECT F1.F\_FIRST AS JU\_FIRST, F1.F\_LAST AS JU\_LAST, F2.F\_FIRST AS SU\_FIRST, F2.F\_LAST AS SU\_LAST

FROM FACULTY AS F1

JOIN FACULTY AS F2

WHERE F2.F\_ID = F1.F\_SUPER;

OUTPUT:

JU\_FIRST JU\_LAST SU\_FIRST SU\_LAST

James Sealy Mark Zhulin

Teresa Marx Jonnel Brown

Colin Langley Jonnel Brown

Q6:

a. CREATE VIEW FACULTY\_view

AS SELECT F\_ID, F\_LAST, F\_FIRST, F\_MI, LOC\_ID, F\_PHONE, F\_RANK, F\_SUPER

FROM FACULTY;

b. INSERT INTO FACULTY\_view (F\_ID, F\_LAST, F\_FIRST, F\_MI, LOC\_ID, F\_PHONE, F\_RANK)

VALUES (6, 'May', 'Lisa', 'I', 11, '3256789012', 'Assistant');

c. SELECT \* FROM FACULTY\_view;

OUTPUT:

F\_ID F\_LAST F\_FIRST F\_MI LOC\_ID F\_PHONE F\_RANK F\_SUPER

1 Marx Teresa J 9 3251234567 Associate 4

2 Zhulin Mark M 10 3252345678 Full

3 Langley Colin A 12 3253456789 Assistant 4

4 Brown Jonnel D 11 3254567890 Full

5 Sealy James L 13 3255678901 Associate 2

6 May Lisa I 11 3256789012 Assistant

d. By inserting the tuple (6, 'May', 'Lisa', 'I', 11, '3256789012', 'Assistant') , we insert each piece of data into each column from left to right, arranged by attribute. Since there is no F\_SUPER listed, MySQL registers it as a NULL value.

e. SELECT FACULTY\_view.F\_LAST, FACULTY\_view.F\_FIRST, LOCATION.BLDG\_CODE, LOCATION.ROOM

FROM FACULTY\_view

INNER JOIN LOCATION ON FACULTY\_view.LOC\_ID=LOCATION.LOC\_ID;

OUTPUT:

F\_LAST F\_FIRST BLDG\_CODE ROOM

Marx Teresa BUS 424

Zhulin Mark BUS 402

Brown Jonnel BUS 433

May Lisa BUS 433

Langley Colin LIB 217

Sealy James LIB 222

f. DROP VIEW FACULTY\_view;

g. Views are an abstraction to visualize data in the database. They are usually temporary anyways, so deleting the view will have no effect on the database, or its data. Views are not part of the schema.

Q7:

a. UPDATE LOCATION

SET ROOM = 211

WHERE LOC\_ID IN(

SELECT LOC\_ID

FROM FACULTY

WHERE F\_LAST IN('Brown')

);

CHECK:

SELECT LOC\_ID, BLDG\_CODE, ROOM

FROM LOCATION

WHERE LOC\_ID = 11;

OUTPUT:

LOC\_ID BLDG\_CODE ROOM

11 BUS 211

b. CREATE TABLE ENROLLMENT\_numbers (

C\_SEC\_ID int,

NUM\_STUDENTS int

);

INSERTING DATA:

INSERT INTO ENROLLMENT\_numbers (C\_SEC\_ID, NUM\_STUDENTS)

VALUES (1, 4), (2, 0), (3, 0), (4, 1), (5, 2), (6, 2), (7, 0), (8, 0), (9, 3), (10, 0), (11, 3), (12, 3), (13,2);

OUTPUT:

C\_SEC\_ID NUM\_STUDENTS

1 4

2 0

3 0

4 1

5 2

6 2

7 0

8 0

9 3

10 0

11 3

12 3

13 2

Additional Queries:

Primary Key Setup:

ALTER TABLE `Group3`.`COURSE`

CHANGE COLUMN `COURSE\_ID` `COURSE\_ID` INT(11) NOT NULL DEFAULT 0 ,

ADD PRIMARY KEY (`COURSE\_ID`);