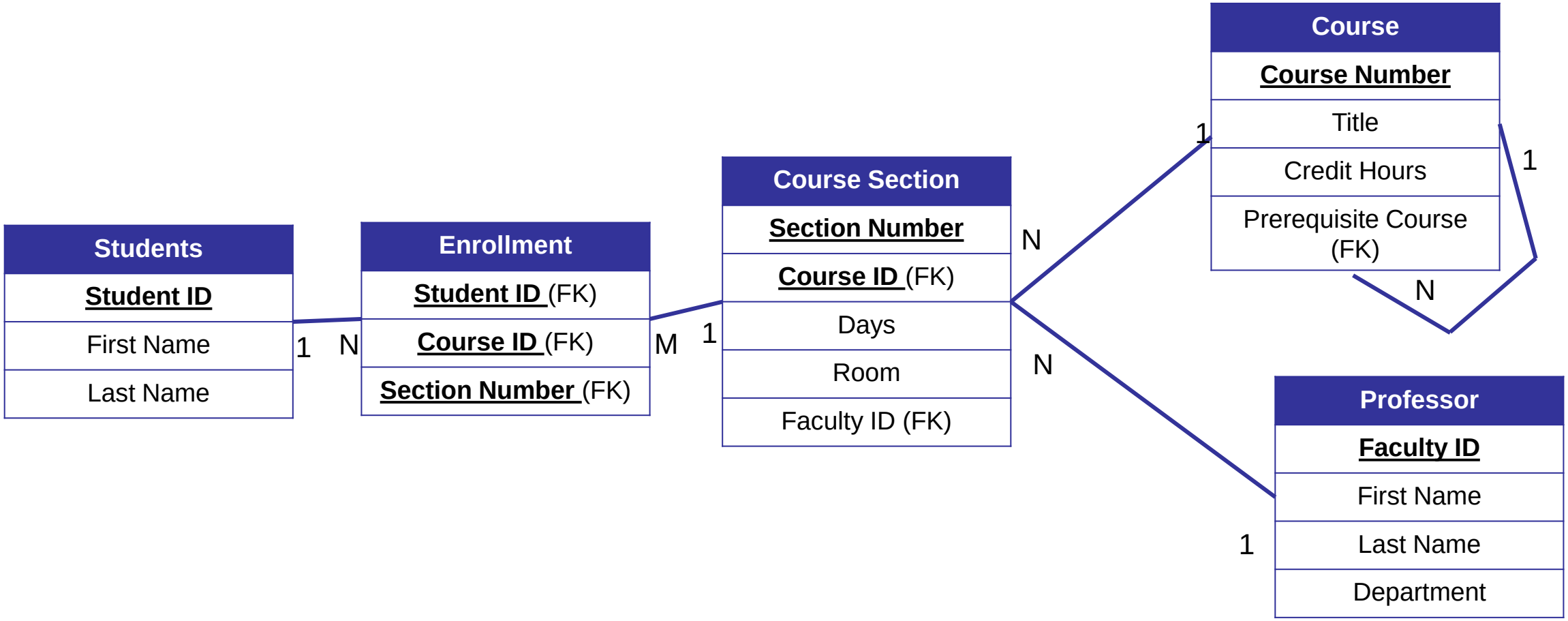
Database Creation in MySQL

- Chapter 11 in Murach text.
- EER diagrams in MySQL Workbench
- DDL in MySQL (CREATE, DROP, USE, ALTER, etc.)
- INSERT statements in MySQL

DDL Statements in MySQL

- Basic DDL SQL Keywords
  - CREATE
  - DROP
  - USE

University DB Example



Creating Databases in MySQL

Syntax

CREATE DATABASE [IF NOT EXISTS] db_name

DROP DATABASE [IF EXISTS] db_name

USE db_name

```
1 •  DROP DATABASE IF EXISTS university;
2 •  CREATE DATABASE university;
3
4 •  USE university;
5
```

Creating Tables in MySQL

```
CREATE TABLE [db_name.]table_name
(
        column_name_1           data_type           [column_attributes][,]
        [column_name_2          data_type           [column_attributes], ]
        [column_name_3          data_type           [column_attributes], ]
        [table_level_constraints]
);
```

Column Attributes

- Common column attributes
  - NOT NULL
  - UNIQUE
  - AUTO INCREMENT
  - DEFAULT default_value

```
 7  CREATE TABLE students
 8  (
 9      studentID               INT             PRIMARY KEY     NOT NULL,
10      studentFirstName        VARCHAR(40)     NOT NULL,
11      studentLastName         VARCHAR(40)     NOT NULL
12      );
13
14  CREATE TABLE courses
15  (
16      courseID            VARCHAR(10)     NOT NULL    PRIMARY KEY,
17      courseTitle         VARCHAR(60)     NOT NULL,
18      creditHours         INT             NOT NULL,
19      prereqCourse        VARCHAR(10)
20      );
21
22  CREATE TABLE sections
23  (
24      sectionNumber       INT             NOT NULL,
25      courseID            VARCHAR(10)     NOT NULL,
26      sectionDays         varchar(20)     NOT NULL,
27      room                VARCHAR(20)     NOT NULL,
28      facultyID           INT             NOT NULL,
29      CONSTRAINT  sections_pk PRIMARY KEY (courseID, sectionNumber)
30      );
```

Common Data Types in MySQL

String Data Types

Data Type Syntax	Maximum Size	Explanation
CHAR(*size*)	Maximum size of 255 characters.	Where ***size*** is the number of characters to store. Fixed-length strings. Space padded on right to equal ***size*** characters.
VARCHAR(*size*)	Maximum size of 255 characters.	Where ***size*** is the number of characters to store. Variable-length string.
TINYTEXT(*size*)	Maximum size of 255 characters.	Where ***size*** is the number of characters to store.
TEXT(*size*)	Maximum size of 65,535 characters.	Where ***size*** is the number of characters to store.
MEDIUMTEXT(*size*)	Maximum size of 16,777,215 characters.	Where ***size*** is the number of characters to store.
LONGTEXT(*size*)	Maximum size of 4GB or 4,294,967,295 characters.	Where ***size*** is the number of characters to store.
BINARY(*size*)	Maximum size of 255 characters.	Where ***size*** is the number of binary characters to store. Fixed-length strings. Space padded on right to equal ***size*** characters. (Introduced in MySQL 4.1.2)
VARBINARY(*size*)	Maximum size of 255 characters.	Where ***size*** is the number of characters to store. Variable-length string. (Introduced in MySQL 4.1.2)

Numeric Data Types

Data Type Syntax	Maximum Size	Explanation
BIT	Very small integer value that is equivalent to TINYINT(1). Signed values range from -128 to 127. Unsigned values range from 0 to 255.	
TINYINT(*m*)	Very small integer value. Signed values range from -128 to 127. Unsigned values range from 0 to 255.	
SMALLINT(*m*)	Small integer value. Signed values range from -32768 to 32767. Unsigned values range from 0 to 65535.	
MEDIUMINT(*m*)	Medium integer value. Signed values range from -8388608 to 8388607. Unsigned values range from 0 to 16777215.	
INT(*m*)	Standard integer value. Signed values range from -2147483648 to 2147483647. Unsigned values range from 0 to 4294967295.	
INTEGER(*m*)	Standard integer value. Signed values range from -2147483648 to 2147483647. Unsigned values range from 0 to 4294967295.	This is a synonym for the INT datatype.
BIGINT(*m*)	Big integer value. Signed values range from -9223372036854775808 to 9223372036854775807. Unsigned values range from 0 to 18446744073709551615.	
DECIMAL(*m*,*d*)	Unpacked fixed point number. ***m*** defaults to 10, if not specified. ***d*** defaults to 0, if not specified.	Where ***m*** is the total digits and ***d*** is the number of digits after the decimal.
DEC(*m*,*d*)	Unpacked fixed point number. ***m*** defaults to 10, if not specified. ***d*** defaults to 0, if not specified.	Where ***m*** is the total digits and ***d*** is the number of digits after the decimal. This is a synonym for the DECIMAL datatype.

Date/Time Data Types

Data Type Syntax	Maximum Size	Explanation
DATE	Values range from '1000-01-01' to '9999-12-31'.	Displayed as 'YYYY-MM-DD'.
DATETIME	Values range from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.	Displayed as 'YYYY-MM-DD HH:MM:SS'.
TIMESTAMP(*m*)	Values range from '1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' UTC.	Displayed as 'YYYY-MM-DD HH:MM:SS'.
TIME	Values range from '-838:59:59' to '838:59:59'.	Displayed as 'HH:MM:SS'.
YEAR[(2|4)]	Year value as 2 digits or 4 digits.	Default is 4 digits.

Column Attributes

Another statement that creates a table with column attributes

```
CREATE TABLE invoices
(
  invoice_id        INT             NOT NULL    UNIQUE,
  vendor_id         INT             NOT NULL,
  invoice_number    VARCHAR(50)     NOT NULL,
  invoice_date      DATE,
  invoice_total     DECIMAL(9,2)    NOT NULL,
  payment_total     DECIMAL(9,2)                DEFAULT 0
)
```

Add ‘professors’ Table in MySQL

Professor
Faculty ID
First Name
Last Name
Department

```
32 • CREATE TABLE professors
33   (
34       facultyID              INT             PRIMARY KEY     NOT NULL,
35       facultyFirstName       VARCHAR(40)     NOT NULL,
36       facultyLastName        VARCHAR(40)     NOT NULL,
37       department             VARCHAR(40)
38       );
```

Primary Key Constraint

- Primary Key = unique & not null
- Column-level primary key constraint

```
7   CREATE TABLE students
8   (
9       studentID               INT             PRIMARY KEY     NOT NULL,
10      studentFirstName        VARCHAR(40)     NOT NULL,
11      studentLastName         VARCHAR(40)     NOT NULL
12      );
```

- Table-level primary key constraint

```
22  CREATE TABLE sections
23  (
24      sectionNumber       INT             NOT NULL,
25      courseID            VARCHAR(10)     NOT NULL,
26      sectionDays         varchar(20)     NOT NULL,
27      room                VARCHAR(20)     NOT NULL,
28      facultyID           INT             NOT NULL,
29      CONSTRAINT  sections_pk PRIMARY KEY (courseID, sectionNumber)
30      );
```

Foreign Key Constraints

- Foreign Key = referential integrity
- Column-level foreign key constraint

```
CREATE TABLE invoices
(
  invoice_id        INT     PRIMARY KEY,
  vendor_id         INT     REFERENCES vendors (vendor_id),
  invoice_number    VARCHAR(50)     NOT NULL     UNIQUE
)
```

- Table-level foreign key constraint

```
CREATE TABLE invoices
(
  invoice_id        INT             PRIMARY KEY,
  vendor_id         INT             NOT NULL,
  invoice_number    VARCHAR(50)     NOT NULL     UNIQUE,
  CONSTRAINT invoices_fk_vendors
    FOREIGN KEY (vendor_id)
        REFERENCES vendors (vendor_id)
)
```

Foreign Key Constraints

A constraint that uses the ON DELETE clause

```
CONSTRAINT invoices_fk_vendors
  FOREIGN KEY (vendor_id) REFERENCES vendors (vendor_id)
  ON DELETE CASCADE
```

An INSERT statement that fails because a related row doesn't exist

```
INSERT INTO invoices
VALUES (1, 1, '1')
```

The response from the system

```
Error Code: 1452. Cannot add or update a child row: a
foreign key constraint fails ('ex'.'invoices', CONSTRAINT
'invoices_fk_vendors' FOREIGN KEY ('vendor_id')
REFERENCES 'vendors' ('vendor_id'))
```

ALTER Statements

A statement that adds a primary key constraint

```
ALTER TABLE vendors
ADD PRIMARY KEY (vendor_id)
```

A statement that adds a foreign key constraint

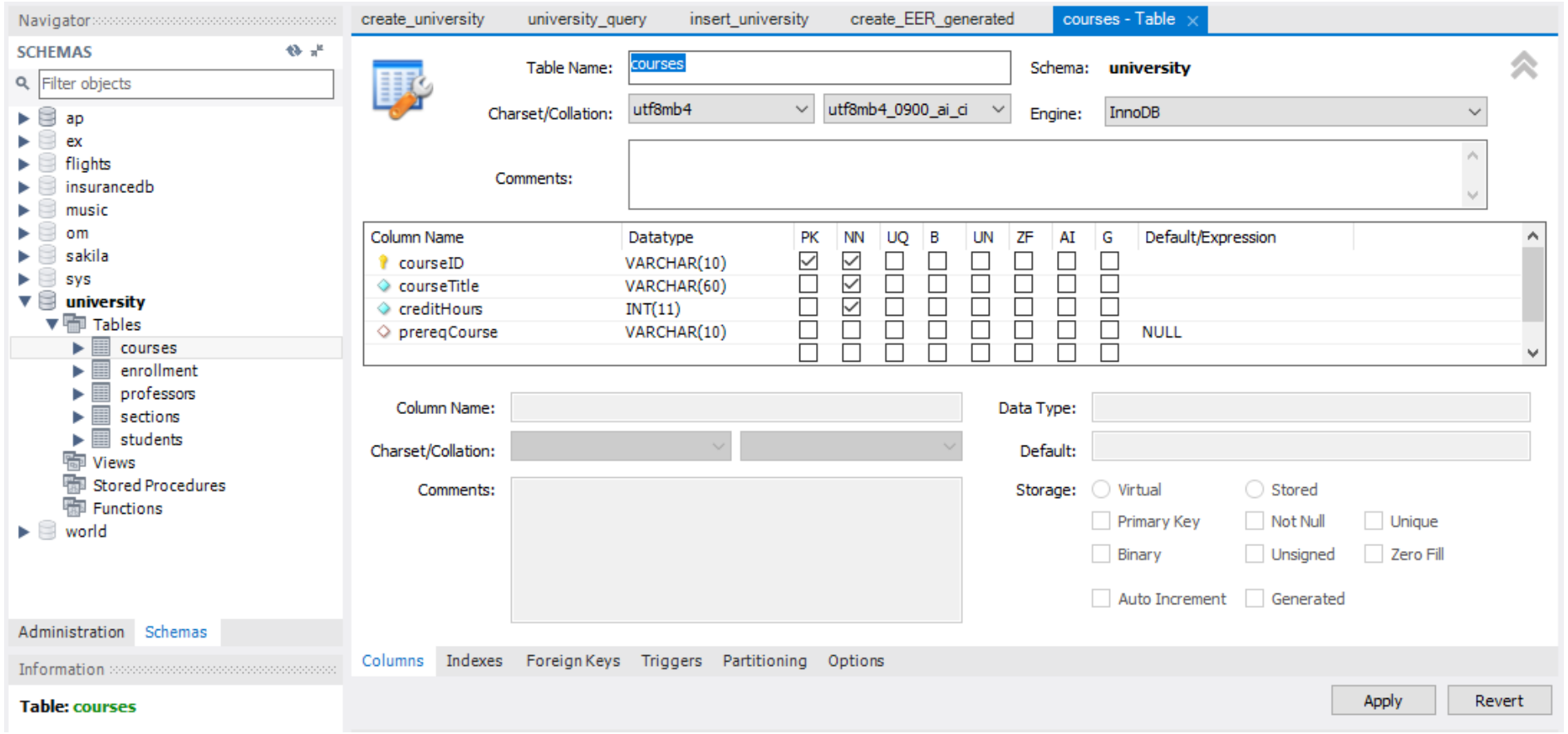
```
56 •  ALTER TABLE enrollment
57    ADD CONSTRAINT enrollment_fk_sections
58    FOREIGN KEY (courseID, sectionNumber) REFERENCES sections (courseID, sectionNumber);
59
60 •  ALTER TABLE enrollment
61    ADD CONSTRAINT enrollment_fk_students
62    FOREIGN KEY (studentID) REFERENCES students (studentID);
```

A statement that changes the type of a column

```
ALTER TABLE vendors
MODIFY vendor_name CHAR(100) NOT NULL UNIQUE
```

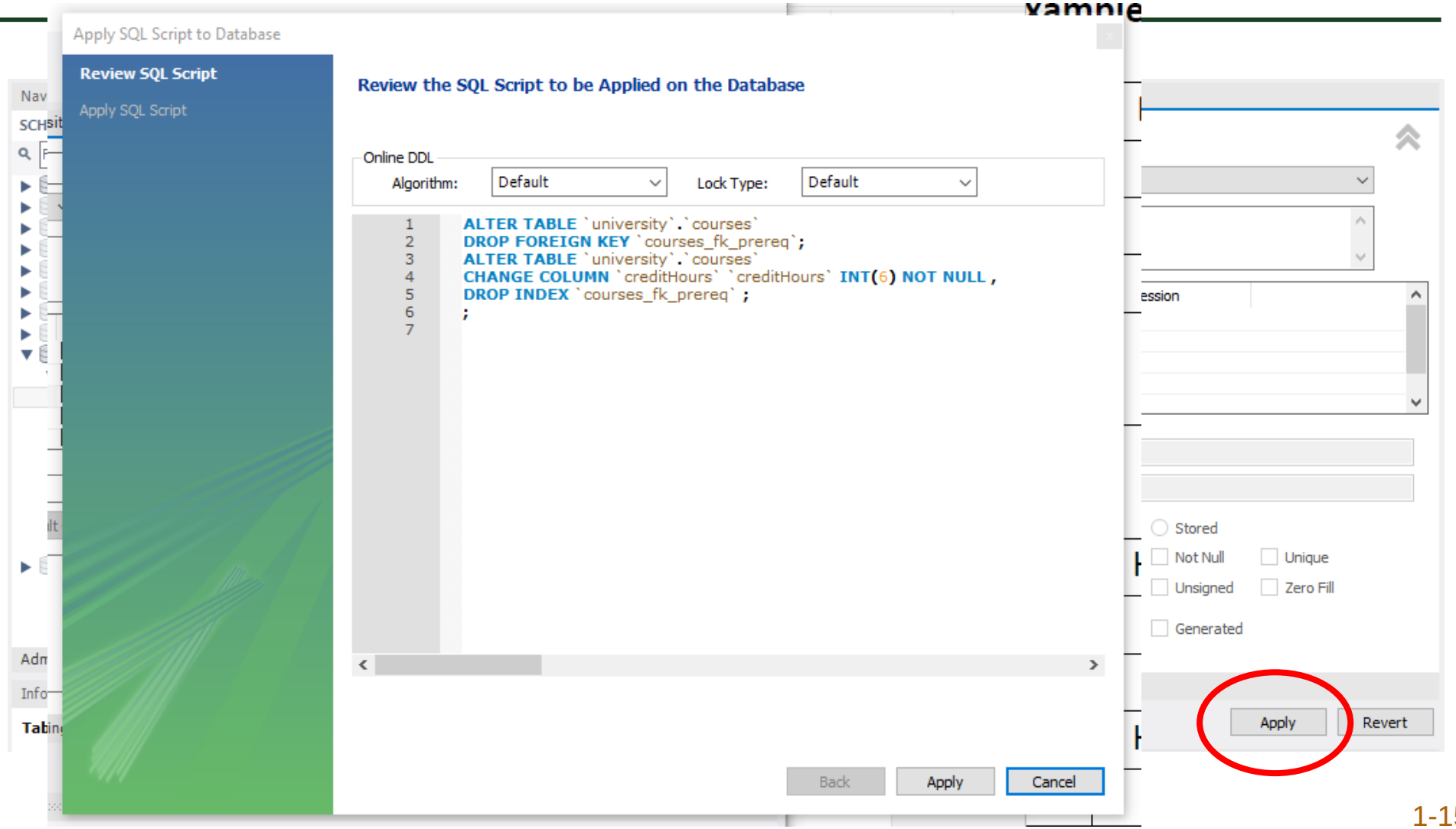
Using MySQL Workbench to ALTER tables

Right-click table name in Schemas window
> Alter Table…



Using MySQL Workbench to ALTER tables

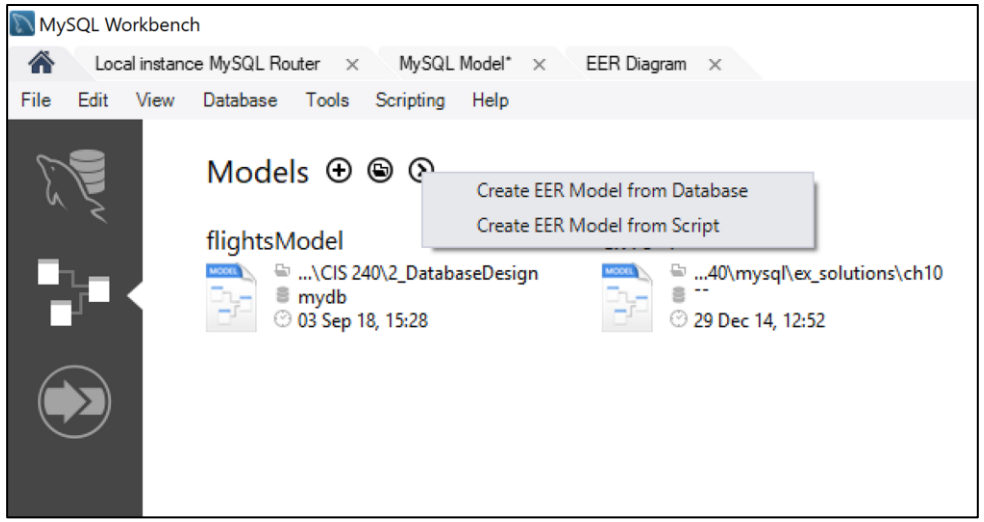
Right-click table name in Schemas window
> Alter Table…

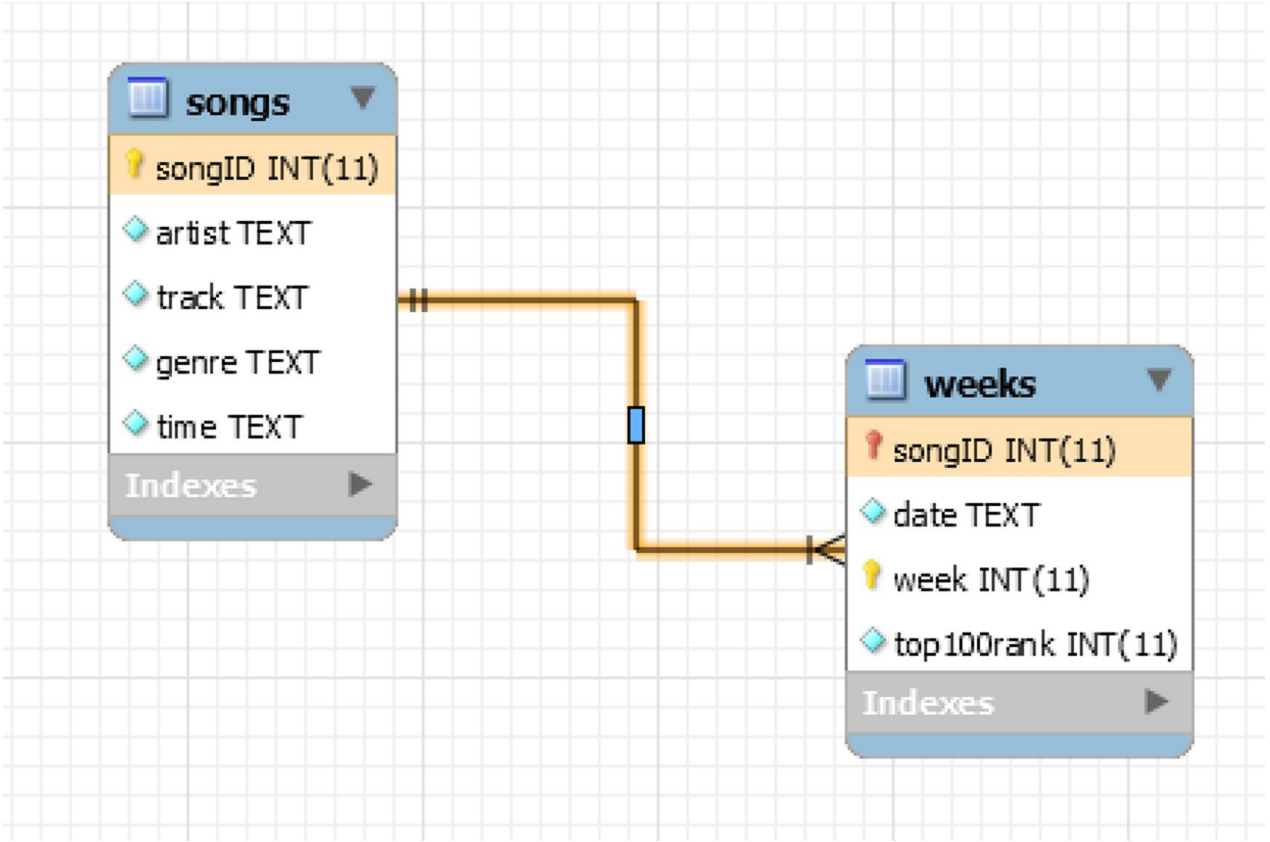


Add ‘studentID’ Foreign Key to ‘enrollment’ Table in MySQL

```
60 •  ALTER TABLE enrollment
61    ADD CONSTRAINT enrollment_fk_students
62    FOREIGN KEY (studentID) REFERENCES students (studentID);
63
```

EER Diagrams in MySQL Workbench





EER Diagrams in MySQL Workbench

Create EER diagrams

- From an existing database
- From an SQL creation script
- From scratch in MySQL Workbench

- EER model can also be used to create an SQL creation script
  - File -> Export -> Forward Engineer MySQL Create Script

CHARSET & COLLATION

- You can read more in Chapter 11 Murach if you are interested.
- One note: Using MySQL Workbench to generate "create_university.sql" could use a default charset or collation that is not available on another DB

```
87  CREATE TABLE IF NOT EXISTS `university`.`enrollment` (
88    `studentID` INT(11) NOT NULL,
89    `courseID` VARCHAR(10) NOT NULL,
90    `sectionNumber` INT(11) NOT NULL,
91    PRIMARY KEY (`studentID`, `courseID`, `sectionNumber`),
92    INDEX `enrollment_fk_sections` (`courseID` ASC, `sectionNumber` ASC)
93    CONSTRAINT `enrollment_fk_sections`
94      FOREIGN KEY (`courseID` , `sectionNumber`)
95      REFERENCES `university`.`sections` (`courseID` , `sectionNumber`),
96    CONSTRAINT `enrollment_fk_students`
97      FOREIGN KEY (`studentID`)
98      REFERENCES `university`.`students` (`studentID`))
99  ENGINE = InnoDB
100 DEFAULT CHARACTER SET = utf8mb4
101 COLLATE = utf8mb4_0900_ai_ci;
```

```
62  13:04:45  CREATE TABLE IF NOT EXISTS `university`.`sections` (  `sectionNumber` INT(11) NOT NULL,  ...  0 row(s) affected
63  13:04:45  CREATE TABLE IF NOT EXISTS `university`.`students` (  `studentID` INT(11) NOT NULL,  `stud...  Error Code: 1273. Unknown collation: 'utf8mb4_0900_ai_ci'
```

INSERT rows into tables

Insert a single row without using a column list

```
INSERT INTO invoices VALUES
(115, 97, '456789', '2014-08-01', 8344.50, 0, 0, 1,
'2014-08-31', NULL)

(1 row affected)
```

Insert a single row using a column list

```
INSERT INTO invoices
    (vendor_id, invoice_number, invoice_total, terms_id,
    invoice_date, invoice_due_date)
VALUES
    (97, '456789', 8344.50, 1, '2014-08-01',
    '2014-08-31')

(1 row affected)
```

INSERT rows into tables

Insert multiple rows

```
INSERT INTO invoices VALUES
    (116, 97, '456701', '2014-08-02', 270.50, 0, 0, 1,
    '2014-09-01', NULL),
    (117, 97, '456791', '2014-08-03', 4390.00, 0, 0, 1,
     '2014-09-02', NULL),
    (118, 97, '456792', '2014-08-03', 565.60, 0, 0, 1,
    '2014-09-02', NULL)
(3 rows affected)
```

insert_university.sql

```
1   USE university;
2
3   INSERT INTO students
4   VALUES (1003,'Phil','Kelly'),(1006,'Virginia','Jones'),(1016,'Diane','Simpson'),
5   (1031,'Joe','Robertson'),(1041,'Natalie','Lee'),(1050,'Donna','Chapman'),(1062,'Kevin','Pull
6   (1074,'Jonathan','Hart'),(1086,'Joshua','Newman'),(1098,'John','Peake'),(1107,'Theresa','Gil
7   (1122,'Steven','King'),(1137,'Amanda','Cornish'),(1146,'Leah','Mills'),(1156,'Maria','Piper'
8   (1161,'Joshua','Knox'),(1164,'Caroline','Poole'),(1178,'Liam','Mills'),(1192,'Joe','Terry'),
9   (1196,'Anna','Roberts'),(1199,'Peter','Marshall'),(1210,'Amelia','Hill'),(1224,'Michelle','R
10  (1230,'Jason','Bower'),(1236,'Jane','Grant'),(1245,'Bella','Gray'),(1258,'Nathan','Gibson'),
11  (1271,'Christopher','Taylor'),(1277,'William','Wright'),(1286,'Katherine','Parsons'),
```

```
265 INSERT INTO enrollment
266 VALUES
267 (1224,'BIOL129',1),(1544,'BIOL129',1),(1617,'BIOL129',1),(1622,'BIOL129',1),
268 (1122,'BIOL129',2),(1968,'BIOL129',2),(2121,'BIOL129',2),(2491,'BIOL129',2),
269 (1041,'BIOL141',1),(1286,'BIOL141',1),(1406,'BIOL141',1),(1585,'BIOL141',1),
```

Create_University Script

- Create_university.sql
- Insert_university.sql