SQL – EdX Part I

**Data Modeling Lab**

**edX Exercise Part I**

**Overview**

In this lab, you will build the data model from a dataset that contains users who took courses taught on the edX platform. These data are provided to you as a single Excel spreadsheet, almost exactly as they were made available from the original website. There are data for over 500,000 course registrations, from users around the world, to courses offered in 17 different sessions. We will look at a subset of the data, about 63,000 rows.

In order to make inferences from this dataset, we’ll need to clean up certain fields and also make some adjustments to the data model.

This is a common task to undertake for customers or clients. They’ll often hand you data for analysis in a giant Excel sheet, and it will be your job to understand the business rules underpinning the dataset.

**Reminder: What constitutes a data model?**

Entities: objects

e.g. distribution center

Attributes: properties of objects

e.g. distribution center address

Primary keys: unique key for one record

e.g. distribution center ID number (or name)

Foreign keys: column that references another column in a different table (often PK)

e.g. factory ID or name

note: the foreign key must exist in the table being referenced

Relationships: “How many of one entity are related to another entity”

e.g. 1 warehouse serves *n* stores

Referential integrity: every value of a foreign key must exist in another table

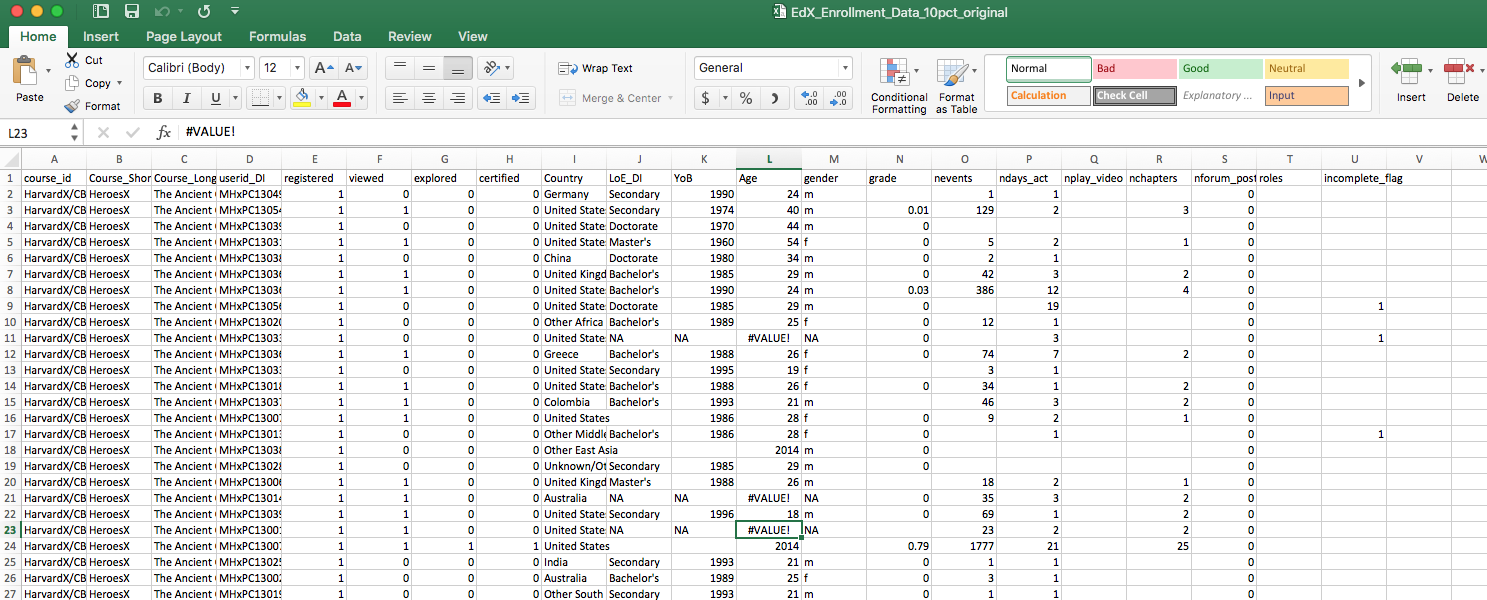
e.g. factory name must exist in factory table

**Exploring the dataset**

**DO THIS ON YOUR OWN, AND THEN WE WILL DISCUSS AS A CLASS**

It’s frequently helpful (especially if the dataset is small enough) to open it in Excel and manually inspect it. There are other tools you can use to inspect large files, and there are also tools that you can use to manipulate them if needed (you’ve already seen how to do some of this in R.)

If you open up the dataset in Excel, you can take a look at it quickly to see what it looks like and what data are included.



The data that are available as you build the database include:

|  |  |
| --- | --- |
| Column Name | Description |
| course\_id | three-part identifier for a course number |
| Course Short Title | Short title for the course |
| Course Long Title | Long title for the course |
| userid\_DI | Individual user ID |
| registered | Whether the user is registered (1/0) |
| viewed | Whether the user has viewed the contents (1/0) |
| explored | Whether the user has explored the course (1/0) |
| certified | Whether the user is certified (1/0) |
| Country | User’s country of origin |
| LoE\_DI | User’s level of education |
| YoB | User’s year of birth |
| Age | User’s age |
| gender | User’s gender |
| grade | User’s grade in the course |
| nevents | Number of events the user has done on the site |
| ndays\_act | Number of actions taken by the user |
| nplay\_video | Number of video plays done by the user |
| nchapters | Number of chapters read by the user |
| nforum\_posts | Number of forum posts made by the user |
| roles | Any roles the user has |
| incomplete\_flag | Whether the user has an incomplete for the course |

**Some preliminary questions:**

The first thing to think about is what each row in this dataset represents. What do you think each row uniquely represents if you had to say it in a phrase?

*Info about a certain number of users for the HeroesX course*

Are there any fields that have heavily replicated data? Why?

|  |  |  |
| --- | --- | --- |
| course\_id, | Course\_Short\_Title, | Course\_Long\_Title |

*Redundant because this info is from the EdX website for the HeroesX course and the long title for that course is the same*

**Brief note about column names:** Database queries can get tricky if there are spaces in any of the column names. Are there any columns that have spaces? In this example, columns B and C were originally called “Course Short Name” and “Course Long Name”. This makes the syntax harder to write out when you want to make queries. We have changed these fields to “course\_short\_name” and “course\_long\_name” for consistency with the other fields and also so that they will be easier to query later.

Let’s take a deeper look at the contents of the fields. There are a few things that look tricky about this data that might make it hard to put in a database correctly. Some fields are composites of several pieces of data. For example, the very first field, course\_id, seems to have three parts, which actually relate to three different pieces of data. What data do they represent and how would you take the information in this single column and convert it to be more ready for database analysis?

HarvardX/CB22x/2013\_Spring -- Institution/Course Number/Semester and Year

Split this 3 parts of the Course ID into separate fields/attributes

Do you see any columns which have weird values or no values? What do you think explains this and do we need to do anything to address this now?

YoB, Grade, nplay\_video, nchapters, roles, incomplete\_flag, LoE\_iD

Some reasons may be that the user hasn't entered the data, entered the data incorrectly or that the system hasn't registered a value

if the value is not binary, replace \N with unknown

if the value is binary, replace \N with 0

What do the 0 and 1 values in the registered and viewed columns represent?

1 : True, Yes

0: False, No

What is the range of the grade values? Is it traditional 0-100 scoring, and how is the data stored?

**Range: 0-1 as a percentage of traditional scoring**

**Data stored as a float/double**

What about data in the last seven columns? Are missing values okay here?

**Missing values are ok but better to replace \N with a more descriptive value: NotApplicable, Unknown, etc.**

Now that you’ve looked over the data, we can pick appropriate data types for each column. The choices from MySQL Workbench are int for integer data, double for decimal data, datetime for date/time data, binary for true/false data, and text for words and everything else.

**PAUSE HERE, CHECK ANSWERS WITH CLASS, BRING FORWARD ANY QUESTIONS**

What is the most appropriate type of data for each of these columns given the values you have observed?

|  |  |
| --- | --- |
| Column Name | Data Type |
| course\_id | text |
| Institution | text |
| Course\_number | text |
| Course\_term | text |
| Course Short Title | text |
| Course Long Title | text |
| userid\_DI | text/int |
| registered | binary |
| viewed | binary |
| explored | binary |
| certified | binary |
| Country | text |
| LoE\_DI | text |
| YoB | datetime |
| Age | int |
| gender | binary |
| grade | double |
| nevents | int |
| ndays\_act | int |
| nplay\_video | int |
| nchapters | int |
| nforum\_posts | int |
| roles | text |
| incomplete\_flag | binary |

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**

**D. Updating the data model**

In this part of the activity, we will build a data model for the EdX data that will be cleaner and easier to use for queries. A recommended set of steps is given, although you are free to approach this as you wish.

First, we need to decide which of our database columns are attributes, which are entities, and which seem to be both. We have included Institution, course\_number, and course\_term, which were the three fields

|  |  |
| --- | --- |
| Column Name | Entity, Attribute, or Both? |
| course\_id | attribute |
| Institution | both |
| Course\_number | attribute |
| Course\_term | attribute |
| Course Short Title | attribute |
| Course Long Title | attribute |
| userid\_DI | attribute |
| registered | attribute |
| viewed | attribute |
| explored | attribute |
| certified | attribute |
| Country | both |
| LoE\_DI | attribute |
| YoB | attribute |
| Age | attribute |
| gender | attribute |
| grade | both |
| nevents | attribute |
| ndays\_act | attribute |
| nplay\_video | attribute |
| nchapters | attribute |
| nforum\_posts | attribute |
| roles | attribute |
| incomplete\_flag | attribute |

What are the major entities that are needed to represent all of the data stored in this dataset? Each row in the dataset represents a student taking a specific course, which is something we should try and represent. We also know that there is information about courses, such as the institution the course it taught at, and information about users, such as their home country and education level. Try to determine which are the major entities to put in your data model that capture these main ideas, and then decide which attributes go into each entity.

User - ~~userId, LoE\_iD, YoB, Age, Gender, Country~~

Course ~~- userID, CourseID, CourseNum, CourseTerm, ShortTitle, LongTitle, institution~~

Grade - userID, courseID, courseTerm, Grade,

Participation - userID, courseID, courseTerm, nevents,ndays\_act, nplay\_video, nchapters, nforum\_posts, roles, incomplete\_flag

Registration - userId, CourseID, CourseTerm registered, viewed, explored, certified

**PAUSE HERE, CHECK ANSWERS WITH CLASS, BRING FORWARD ANY QUESTIONS (Don't turn the page)**

Now that we know the three major entities that we will create in this database project, let’s define them each one by one.

First define the **Courses** entity. This table defines a unique course offered at EdX. What are the attributes that define a course that is available?

|  |
| --- |
| Courses |
| courseID |
| course\_num |
| course\_term |
| short\_title |
| long\_title |
| institution |
|  |
|  |
|  |
|  |
|  |

Now, define the **Users** entity. This defines only data about a specific user on the EdX platform, not which courses that user is in. Decide what attributes should be put into this entity.

|  |
| --- |
| Users |
| user\_id |
| YoB |
| LoE\_iD |
| Age |
| Gender |
| Country |
|  |
|  |
|  |
|  |
|  |

Finally, we need to capture some way to describe which courses each user is in, how much progress he or she has made in exploring the course materials, a grade (if applicable) and how many interactions the user has had on the EdX web platform. Define the **Course\_Users** entity, which will contain this information for each course registration.

|  |
| --- |
| Course\_Users |
| course\_id |
| course\_term |
| user\_id |
| nevents |
| ndays\_act |
| nplay\_video |
| nchapters |
| nforum\_posts |
| roles |
| incomplete\_flag |
| registered |
| viewed |
| explored |
| certified |
|  |

**PAUSE HERE, CHECK ANSWERS WITH CLASS, BRING FORWARD ANY QUESTIONS Keys**

Now we will pick the field or fields that will serve as the primary key for each table. The primary key must uniquely identify the row, so it cannot be duplicated and it cannot be null, or empty. Why can’t the primary key be repeated or empty?

since primary key is used as identification it must be unique and contain a value

What is the best key or keys to use for the **Courses** table? What are the possible choices of key(s)? What would the different choices change in what each row in Courses represents?

course\_id

course\_num + course\_semester

each row would represent a distinct course rather than a distinct user

What is the best key or keys to use for the **Users** table?

user\_id

What is the best key or keys to use for the **User\_Courses** table?

course\_id + user\_id

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**

**Relationships**

Now we will focus on how the tables relate to one another. Think first about the multiplicity of the relationships between each table. What is the relationship between Users and Courses? What is the relationship between Users and Course\_Users? What is the relationship between Courses and Course\_Users?

users and courses <-- not directly related

users and course\_users <-- related by course\_id and user\_id

courses and course-users <-- related by course\_id

Now let’s draw a relationship diagram (also called an entity-relationship or ER diagram.) First draw out all three tables together, and then draw the links between the tables. Place the appropriate connector (one to many, many to many, or one to one) between each table. Draw the lines from the specific linking field in one table to the same linking field in another table.

**E. Loading the data into MySQL**

<https://www.dropbox.com/s/0trkrqktqqv0b88/import_edx.sql?dl=1>

**Download this script – open it with MySQL Workbench – let's walk through it as a class -- F. Split composite fields**

The goal is to split entries such as:

HarvardX/CB22x/2013\_Spring

Into multiple columns:

HarvardX CB22x 2013\_Spring

This is done in two steps:

Create fields to receive segmented entries

Segment text and populate new columns

**Create new fields**

Create new fields (institution, course\_number, course\_term)

ALTER TABLE \_\_\_\_ ADD institution VARCHAR(50);

Now you set up the other two

**Split fields**

The composite course\_id:

HarvardX/CB22x/2013\_Spring

Uses ‘/’ as separator

Split the content into three parts [This statement just lets us EXAMINE what will happen]

SELECT course\_id, SUBSTRING\_INDEX( `course\_id` , '/', 1 ) AS institution

from \_\_\_\_\_;

Can you figure out how to write the next two? It's tricky.

http://www.w3resource.com/mysql/string-functions/mysql-substring\_index-function.php

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**

**Update settings**

Without a primary key in the tables, the settings need to be updated to allow the changes:

SET SQL\_SAFE\_UPDATES = 0;

**UPDATE query**

Proceed with actual query to place fields into the various columns

UPDATE \_\_\_\_\_

SET institution = SUBSTRING\_INDEX( `course\_id` , '/', 1 );

Can you figure out how to write the next two? It's tricky.

http://www.w3resource.com/mysql/string-functions/mysql-substring\_index-

function.php

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**

**Verify results**

Look at the first 100 rows of the new split up data:

SELECT \* FROM \_\_\_\_\_ LIMIT 100;

Or simply look at the unique combinations:

SELECT DISTINCT institution, course\_number, course\_term from \_\_\_\_\_;

**Single table queries**

List represented countries?

List countries represented by students with doctoral degrees?

Hint:

SELECT DISTINCT \_\_\_\_, \_\_\_ FROM \_\_\_\_\_\_\_ WHERE \_\_ = '\_\_\_\_'

'United States' 'China' 'Unknown/Other' 'Other Europe' 'Other Middle East/Central Asia' 'Other Africa' 'Spain' 'India' 'Other East Asia' 'France' 'United Kingdom' 'Brazil'

'Other South Asia' 'Germany' 'Egypt' 'Australia' 'Portugal'

'Other North & Central Amer. and Caribbean' 'Philippines' 'Canada'

'Russian Federation'

How many distinct courses were offered?

16

How many students came from Germany? 789

Hint:

SELECT COUNT(\*) FROM \_\_\_\_\_\_\_ WHERE \_\_ = '\_\_\_\_'

What percentage of students came from the US?

17550/61537

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**

**Create new ‘Course’ table**

List all possible courses:

-- select distinct all columns you plan to include in table

SELECT DISTINCT \_\_\_\_, \_\_\_\_\_, etc.

FROM \_\_\_\_\_\_\_\_;

Is the list as expected?

Create the actual table:

CREATE TABLE \_\_\_\_\_\_

SELECT DISTINCT \_\_\_\_, \_\_\_\_\_, etc.

FROM \_\_\_\_\_\_\_\_;

Verify

SELECT \* FROM \_\_\_\_\_\_\_;

**Create new ‘Users’ table I**

Count all users using:

-- select distinct all columns you plan to include in table

SELECT COUNT(DISTINCT \_\_\_\_, \_\_\_\_\_, etc. )

FROM \_\_\_\_\_\_\_\_;

Check the result’s count and compare with distincts user\_ids

Why are they different? Some people may have the same set of values even if their user ID is not the same

(same country, same birthday, etc)

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**

Using group by constraint TWO APPROACHES:

SELECT userid\_DI, Country, LoE\_DI, YoB, Age, gender FROM

\_\_\_\_\_\_\_\_\_\_ group by userid\_DI

MySQL >= 5.7 (avoid: incompatible with sql\_mode=only\_full\_group\_by)

SELECT userid\_DI, any\_value(Country), any\_value(LoE\_DI), any\_value(YoB), any\_value(Age), any\_value(gender) FROM \_\_\_\_\_\_\_ group by userid\_DI;

Create the new users table using the appropriate select command from above

Verify ‘users’ table:

SELECT \* FROM users;

**Create new ‘Course\_Users’ table**

This is the original table with fewer columns as the data shifted over to the new tables (courses, users)

Create table

Verify that the row is the same as that of the original enrollment\_data\_10pct table

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**

**Set up primary and foreign keys using the following syntax:**

Hints: after TABLE put table name, inside parentheses put regular column name

ALTER TABLE \_\_\_\_\_\_\_\_\_\_\_

ADD PRIMARY KEY (\_\_\_\_\_\_\_\_\_\_\_)

ALTER TABLE \_\_\_\_\_\_\_

ADD FOREIGN KEY (\_\_\_\_\_) REFERENCES \_\_\_\_(\_\_\_\_\_)

**PAUSE HERE, CHECK ANSWERS WITH NEIGHBOR, BRING FORWARD ANY QUESTIONS TO THE CLASS**