Coursera Data Exports Guide

coursera

Published with GitBook

Table of Contents

Introduction

Welcome, Coursera data researchers! This guide will help you get started with your research data exports by providing an introduction to the many tables. The guide will be updated to reflect major changes in the data export.

As a PDF file, this guide will be included in every data export. It will also be accessible online at the link below: https://www.gitbook.com/book/coursera/data-exports/

Please request an account by emailing us at data-support@coursera.org and participate in commenting in this documentation.

Many of the links in this guide refer to related articles in our Coursera Partner Help Center for more details about our platform's features.

We aim to further your researching needs, whether there is a lack of clarity in how to use the data you have or there are other data you wish to see in the export. Please feel free to reach out to us if you have any questions or comments.

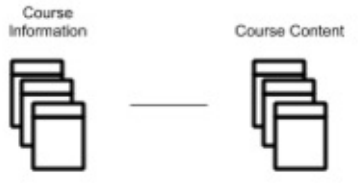
First-Time Readers

If you are a first-time reader, we strongly suggest you cover these sections to help you get started:

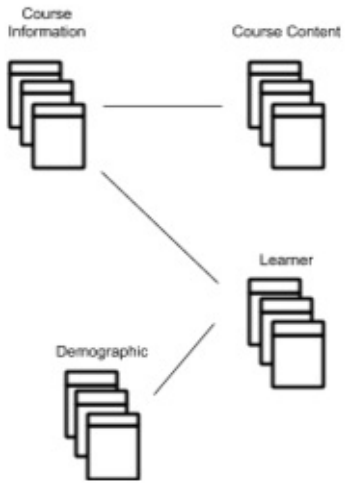
The Data Journey

Below is a journey of how actions from instructors and learners populate the data tables as you see today

- The instructor creates a course.
    - An instructor will create a new "draft" course, which populates Course Information tables, such as the **courses** table.
    - As the instructor creates modules, lessons, quizzes, etc, this populates Course Content tables, such as the **course_modules** table.
    - When ready, an instructor works with Coursera to set course pre-enrollment, launch, and sessions dates, which updates new information in the **courses** table and the **on_demand_sessions** table.



- Learners enroll in the course.
    - When Coursera users preview, pre-enroll, or enroll in a course, this populates the Learner tables, such as the **course_memberships** and **users** tables.
    - Some users voluntarily answer our demographic questions, which populates the Demographic tables.

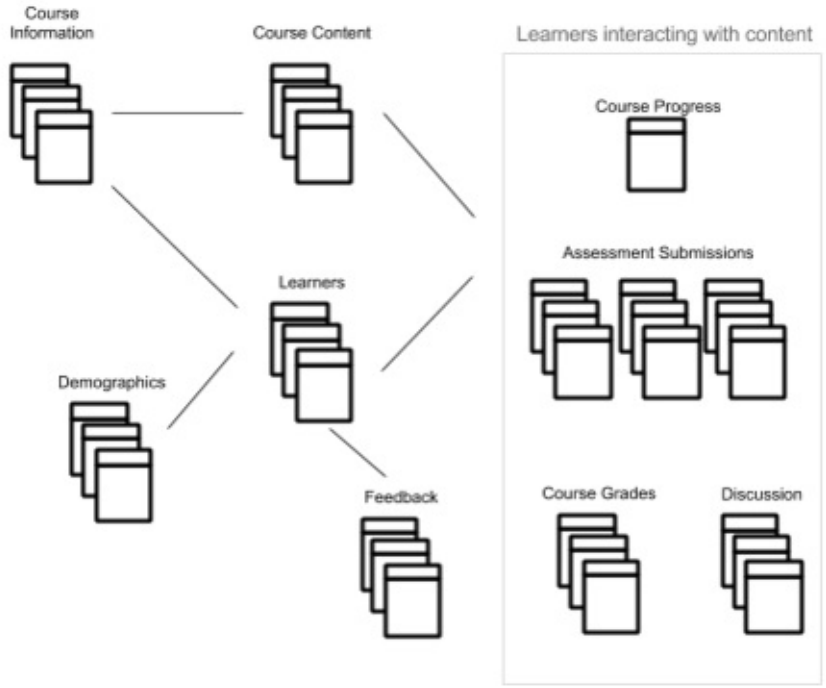


- Class starts and learners interact with course content.
    - As learners start and complete watching lecture videos or taking assessments, as

well as progressing on types of other course content, this populates the **course_progress** table. For detailed actions, like answering incorrectly on a quiz question, this populates Assessments tables, such as the **assessment_responses** table.

- As learners receive grades and complete the course, this populates the Course Grades tables, such as the **course_grades** table.
- As learners post and vote on the course forums, this populates the Discussions tables, such as the **discussion_answers** table.
- As learners provide feedback on course content and the entire course, this populates Feedback tables, such as the **feedback_course_ratings** table.

Below shows the major relationships between tables groups, with minor connections omitted.



Files Included

The data export .zip file includes the following:

- **guide.pdf**: This is the guide for Coursera Data Researchers, which also lives online at this Gitbook link.
- **readme.html:** This file contains documentation for all tables and table columns.
- **CSV** files (e.g. course_grades.csv): Comma-separated value files contain the data header and data values for each table.
- **HTML** files (e.g. course_grades.html): HTML files contain the documentation for each table and a `CREATE TABLE` script for PostgreSQL-compliant databases. In March 2016, we have made our first attempt at adding some `PRIMARY_KEY` and `FOREIGN_KEY` info in the tables.

Loading Data Exports

Coursera collects and stores data for analytics, and we use a data warehouse product, Amazon Redshift, for most of our data processing. The data we provide as research exports are generated by processing and unloading subsets of this data. We want this data to be as useful as possible, and recognize that volume, variety, and formatting may pose challenges.

Parsing the CSV

Each CSV file contains a header with column names followed by zero or more rows of data. If a table contains zero rows, this means that there is no available data. Rows are separated by a newline character (\n). Within each row, columns are enclosed by double-quotes (") and separated by comma characters (,). String column data may contain double-quotes and backslash characters (\). Both of these will be escaped using a backslash character.

As an example with high parsing complexity, here is a very short table with two columns:

```
number,bool,null,string,json
1,t,,"a string","{\"key1\":1,\"key2\":\"\\\"a nested quotation\\\"\"}"
```

The last column includes JSON-formatted data with escaped double-quote and backslash characters. You should take special care to interpret escaped characters correctly when importing data into other programs.

Using Excel

Excel is not an ideal tool for analysis for a few reasons. Firstly, some CSVs may be too large for Excel. Secondly, Excel's automated CSV parsing does not work well with the encoding described above. Finally, our data is highly normalized, so most research would require a significant number of VLOOKUPs.

Nevertheless, you may find success in loading some CSVs into Excel, which can be used to create very basic statistics using, say, the pivot table feature. However, this is likely going to give you similar insights as what you can find in our Course and Specialization Dashboards.

Using a Relational Database

A relational database management system (RDBMS) is ideal for storing and querying the data with SQL. Examples include:

RDBMS	Costs	Environment	Comments
Amazon Redshift	not free	runs in the cloud	what Coursera uses
PostgreSQL	free	runs locally	closest clone to Redshift
MySQL	free	runs locally	another popular RDBMS

If you are unfamiliar with the three examples, we recommend you try out PostgreSQL on your local machine.

PostgreSQL has a community of online resources on how to download, install, and use it. Here are some resources:

- http://postgresguide.com/
- http://www.postgresql.org/docs/9.5/static/tutorial.html
- http://www.postgresqltutorial.com/

The generic efforts of installing any RDBMS and importing data are:

1. Download a RDBMS installation file/package for your OS.
2. Install. The installation software will:
    i. install the database server
    ii. install a SQL client
    iii. prompt you to set up an admin account
3. Run the database server, which will run in background.
4. Run the SQL client program to connect to the database server with your admin account.
5. Write and run the SQL to create a new database and then use that database.
6. From the files in your course data export, import as many data tables as you need. Create the table via the script at the bottom of a HTML file (works with PostgreSQL and not with MySQL), for example:

```
CREATE TABLE courses (
 course_id VARCHAR(50)
 ,course_slug VARCHAR(2000)
 ,course_name VARCHAR(2000)
 ,course_launch_ts TIMESTAMP
 ,course_update_ts TIMESTAMP
 ,course_deleted BOOL
 ,course_graded BOOL
 ,course_desc VARCHAR(10000)
 ,course_restricted BOOL
 ,course_verification_enabled_at_ts TIMESTAMP
 ,primary_translation_equivalent_course_id VARCHAR(50)
 ,course_preenrollment_ts TIMESTAMP
 ,course_workload VARCHAR(100)
 ,course_session_enabled_ts TIMESTAMP
 ,PRIMARY KEY (course_id)
);
```

7. To import into PostgreSQL, use:

```
COPY courses
FROM 'path/to/courses.csv'
CSV
DELIMITER ','
QUOTE '"'
ESCAPE '\'
HEADER;
```

8. Write SQL to analyze data on the loaded table, for example:

```
SELECT *
FROM courses
```

We encourage MOOC data researchers to share tools and techniques with us and to the community. Jasper Ginn, of Leiden University, maintains a thoroughly documented, open-source script to load Coursera export data into a PostgreSQL database.

Python, R, and Other Languages and Tools

Any other language or tool that can handle loading CSVs can be used to import and to "query" the data for research.

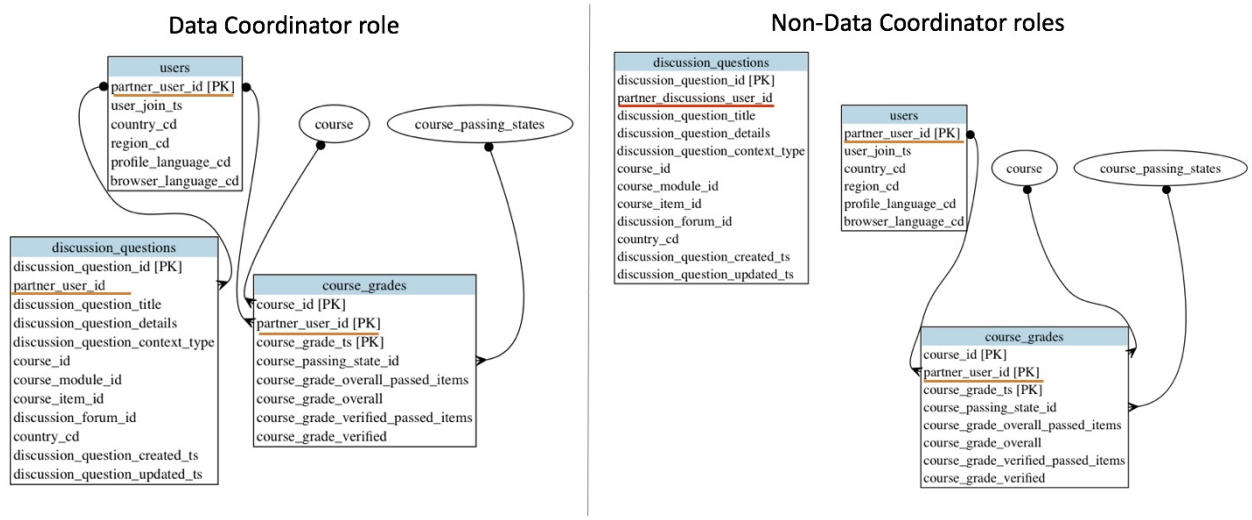
Below is how to configure python to do a successful load of the data export CSV:

```
import csv
csv.register_dialect(
    'coursera-postgres-format',
    delimiter=',',
    doublequote=False,
    escapechar='\\',
    lineterminator='\n',
    quotechar='"')
with open('path/to/courses.csv', 'r') as f:
    reader = csv.reader(f, dialect='coursera-postgres-format')
    header = next(reader)
    rows = list(reader)
```

Coursera welcomes our research partners to share best practices and code examples in these languages and tools outside of RDBMS and SQL.

ID Columns

Per Coursera's Data Sharing Policy and Research Policy, there are two different data export anonymity levels based on the exporter's role. Here is a quick illustration:



Exporters with the Data Coordinator role are allowed to receive data exports where tables with user information will contain the column named *[partner]_user_id*. Therefore, these exporters can identify the same user across the multiple table domains and must follow the PII guidelines in the agreed upon data policy. Data Coordinators can contact us to request this specific type of research data exports.

For exporters without the Data Coordinator role, each table domain uses a separate user ID to distinguish between the different types of learner-generated data. The learner data cannot be joined across domains. This is intended to reduce the scope and impact of accidental PII inclusion and to protect exporters and Coursera when PII data is not required to advance data research.

For example, the **course_grades** table will contain a *[partner]_user_id* column, which will purposely not connectable with the column *[partner]_discussions_user_id* in the **discussion_questions** table. Otherwise, if one provides PII in the discussion forums, a researcher might be able to associate that learner to a particular grade.

If you are not a Data Coordinator, your table domains and user ID columns will be as follows:

ID Column	Domain
[partner]_user_id	in most tables (**users**, **course_progress**, etc.)
[partner]_discussion_user_id	in the **Discussions** tables
[partner]_demographics_user_id	in the **Demographic** tables
[partner]_feedback_user_id	in the **Feedback** tables
[partner]_assessments_user_id	in the **Assessments** tables
[partner]_programming_assignments_id	in the **Assessments** tables
[partner]_peer_assignments_user_id	in the **Assessments** tables

For all exports, ID columns are consistent for a learner across all courses; this allows partners to connect learners' grades and progress across courses.

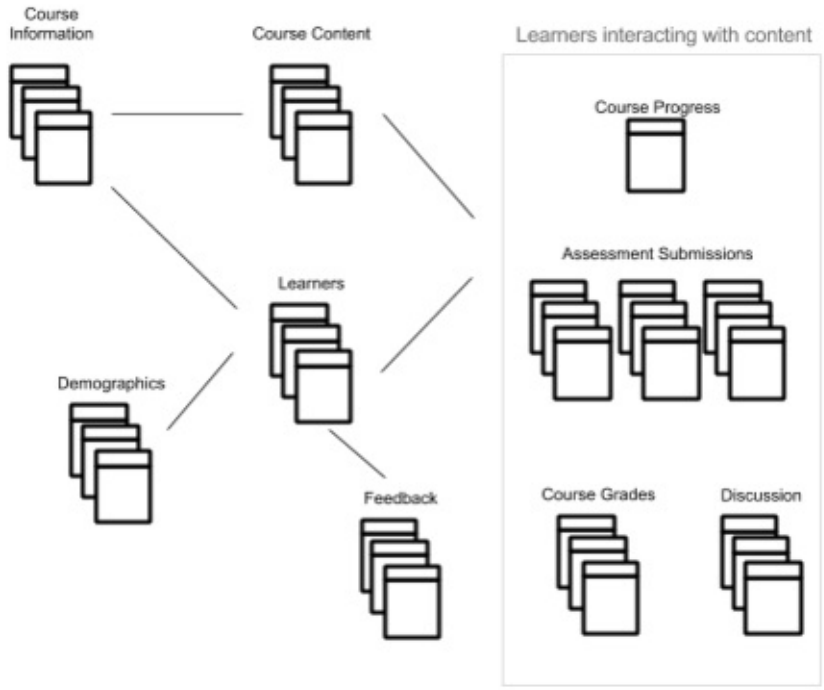
There is also an identifier column in the format of, *[course_slug]_user_id*, that is found in the table **[partner]_course_user_ids**. This table is very useful for instructors to administer surveys to learners, outlined in this Partner Help Center article on External Surveys. Survey responses can be traced to learners' Coursera accounts by including a special tag in the survey URL, such as `http://www.surveyvendor.com/mysurvey?user=%HASHED_USER_ID%`. This `%HASHED_USER_ID%` variable is equal to the *[course_slug]_user_id*.

Data Tables Guide

We will break out the 75+ tables included in your exports into a few key groups.

- **Course Information**: relates to basic information about the course, including a course's name, when its sessions ran, etc.
- **Course Content**: refers to the materials of the course, including modules, lessons, items, etc.
- **Course Progress**: describes learners' interaction with course content.
- **Assessments**: provides in-depth details of interactions with assessments.
- **Course Grades**: details the learners' grades and passing states within a course.
- **Discussions**: contains forums, forum posts, and vote information.
- **Feedback**: contains information regarding user ratings to course content and courses.
- **Learners**: describes learners' info, like when/where the user joined Coursera.
- **Demographics**: contains demographic data based on user surveys.

The major relationships between tables groups, with minor connections omitted, are as



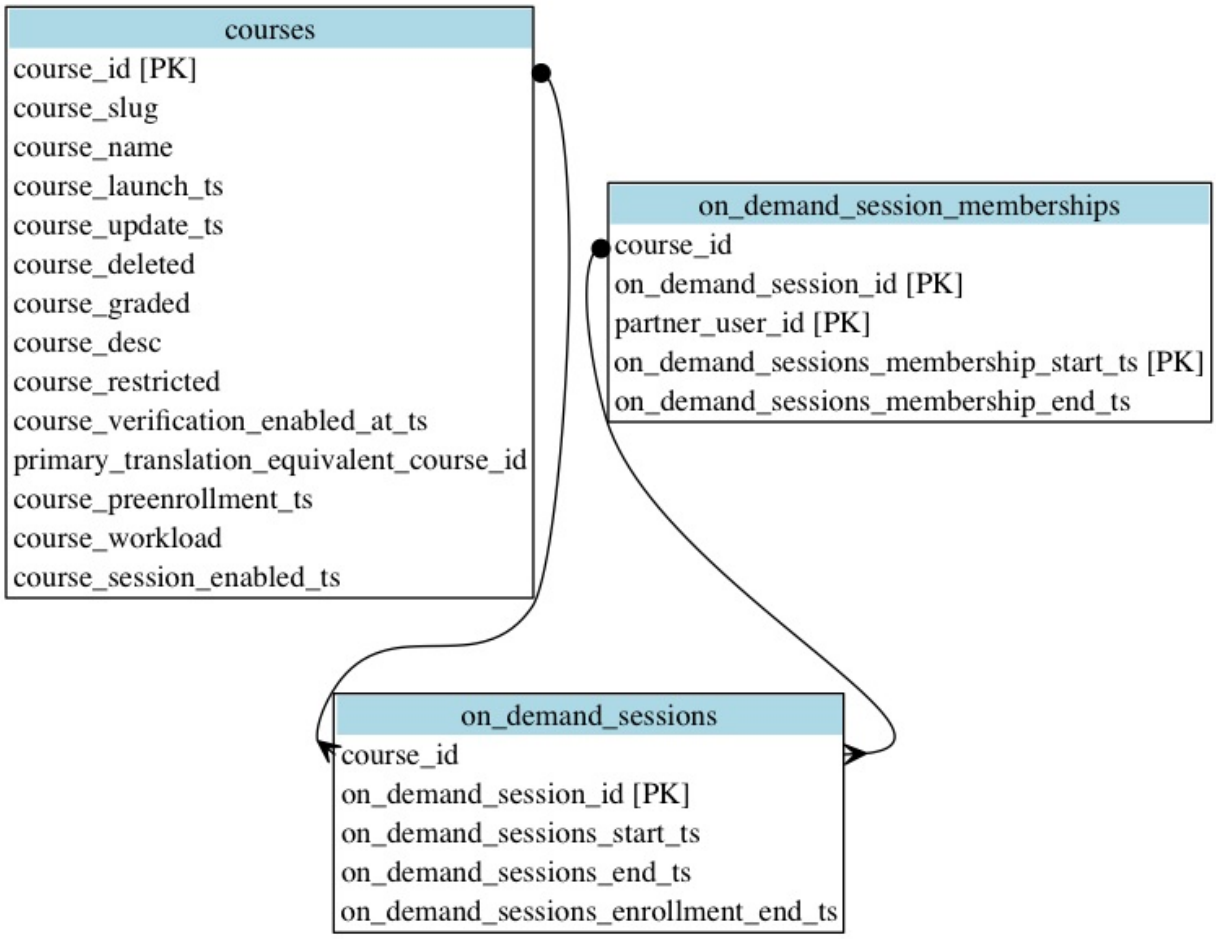
follows:

Course Information Tables

Coursera starts tracking data as soon as an instructor builds a draft course. Course data appears in the **courses** table. When ready, an instructor works with Coursera to set the course pre-enrollment and launch dates. This updates new information in the **courses** table.

Most courses on Coursera are offered in a session (cohort) format. Each session has a start and end date, and learners in the same session work through the course together according to a weekly schedule of suggested deadlines. Sessions are scheduled on a regular cadence (e.g., one start date per month). For more info, please refer to our Partner Help Center article on Course Sessions.

Details about a course's sessions are populated in the **on_demand_sessions** table. When learners enroll in sessions, rows are populated in **on_demand_session_memberships**.



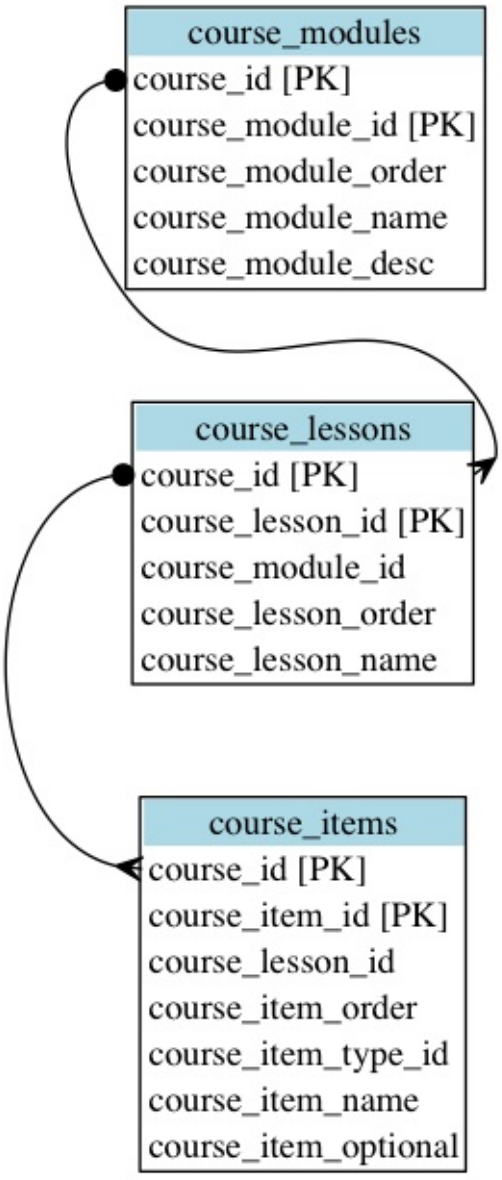
Are there more learners enrolled in my newer (versus older) sessions?

```
SELECT
    on_demand_session_id
    ,on_demand_sessions_start_ts::DATE AS session_start_date
    ,COUNT(DISTINCT institution_user_id) num_learners
FROM on_demand_session_memberships
JOIN on_demand_sessions
    USING (on_demand_session_id)
GROUP BY 1,2
ORDER BY 2;
```

Course Content Tables

Course Content tables are populated when instructors create course content such as modules, lessons, lectures, quizzes, etc.

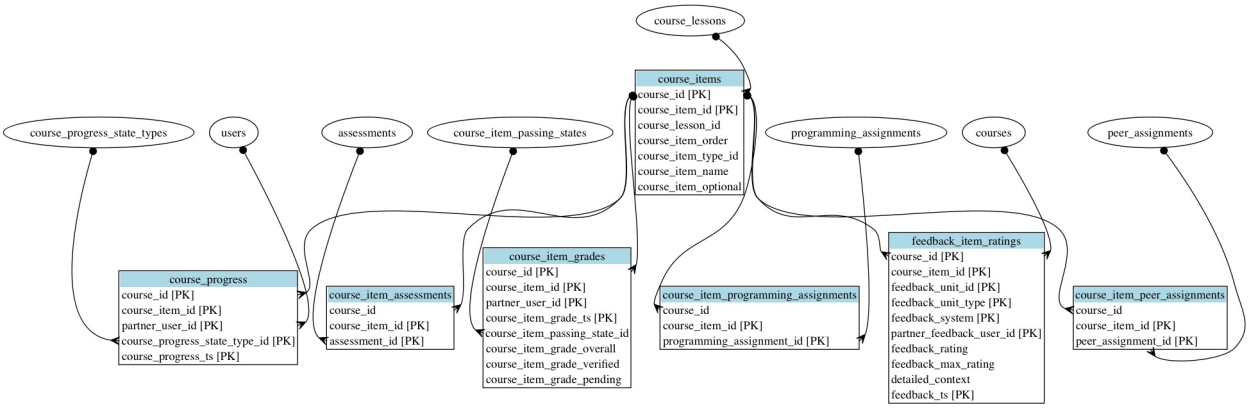
The course is structured as a hierarchy of modules, lessons, and items. The primary three tables are: **course_modules**, **course_lessons**, and **course_items**. Each table contains parent-child foreign keys, names or descriptions, and content order.



Many instructors will tweak, add, or delete content over the lifetime of a course, and these tables will be updated to reflect those changes. Data exports contain the current course offering at the time of export.

Course Versioning is a new Coursera feature launched in 2016 Q1 that allows instructors to create multiple versions, or “branches”, of their course. Please read our Partner Help Center article on Course Versioning. Once this feature becomes widely adopted, our data exports will include that data for research purposes. There will be new tables that list all course content items across all versions.

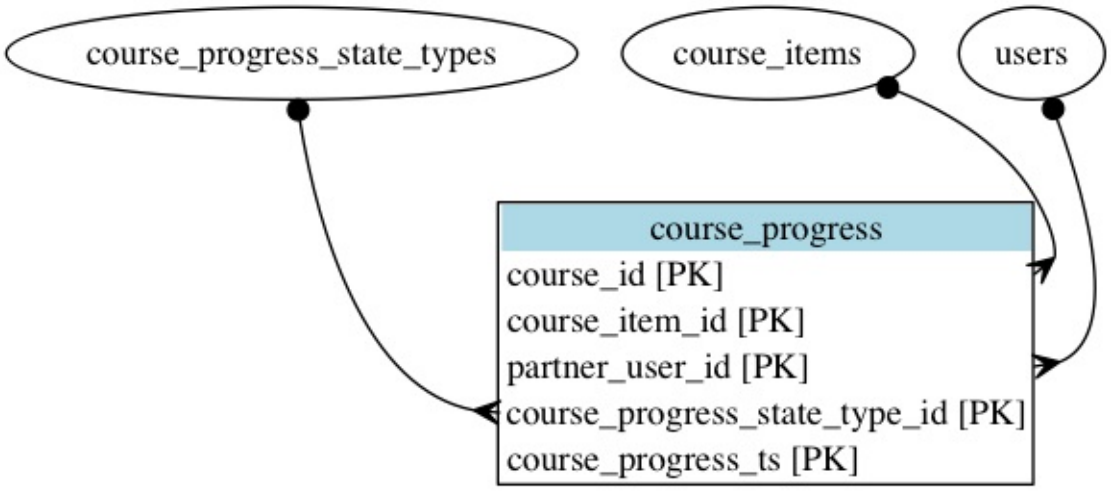
Course items have relationships with many other table groups, such as: Assessments, Course Progress, Course Grades, Feedback, and Discussions.



> click image to enlarge

Course Progress Tables

When a learner interacts with a course item, a row is populated in the **course_progress** table. Each row shows a unique interaction with a user and course item, the time of that interaction, and a progress state of either "started" or "completed".



Each course item type has a different meaning for the progress states.

- For viewing lectures in a browser, playing the video for a few seconds "starts" the lecture, while playing the last few seconds of the video "completes" the lecture.
- For graded quizzes, starting or submitting a non-passing submission "starts" the quiz; submitting a passing submission "completes" the quiz. If the learner passed the completion on the first attempt, there would be no record of a "start", since the learner went straight to a "completed" state.

Related to the topic of instructor's tweaks/additions/deletions of course item contents in the Course Content Table chapter, the **course_progress** table will contain all activity to all *course_item_id*'s that existed in the course.

The **course_progress** table does not include when a user downloads a video lecture. Please see the Frequently Asked Questions chapter for more exceptions.

SQL Example: How many learners started the first graded programming assignment in my course?

```
WITH all_programming_assignments AS (
    SELECT *
        -- window function returns true if it is the first grading programming
        ,1 = ROW_NUMBER() OVER (
            -- sort across all modules and lessons to determine first programming assignm
            ORDER BY course_module_order, course_lesson_order, course_item_order
        ) AS is_first_graded_programming
    FROM course_items
    -- join to allow the search for 'graded programming'
    JOIN course_item_types
        USING (course_item_type_id)
    -- join to get lessons-to-modules
    JOIN course_lessons
        USING (course_lesson_id)
    -- join to get module's course_module_order
    JOIN course_modules
        USING (course_module_id)
    WHERE
        course_item_type_desc = 'graded programming'
)

,first_programming_assignment AS (
    SELECT course_item_id
    FROM all_programming_assignments
    WHERE is_first_graded_programming
)

-- learners that started first programming assignment
SELECT COUNT(DISTINCT [partner]_user_id) AS num_learners
FROM course_progress
-- this inner join only keeps progress on desired course_item_id
JOIN first_programming_assignment
    USING (course_item_id)
JOIN course_progress_state_types
    USING (course_progress_state_type_id)
WHERE course_progress_state_type_desc = 'started';
```

Assessment Tables

For assessment-based course items, a wealth of 30+ tables can provide in-depth details across:

- Quizzes: **assessment_*** tables
- Peer Assignments: **peer_*** tables
- Programming Assignments: **programming_*** tables

The three groups above are similar in structure. They each contain:

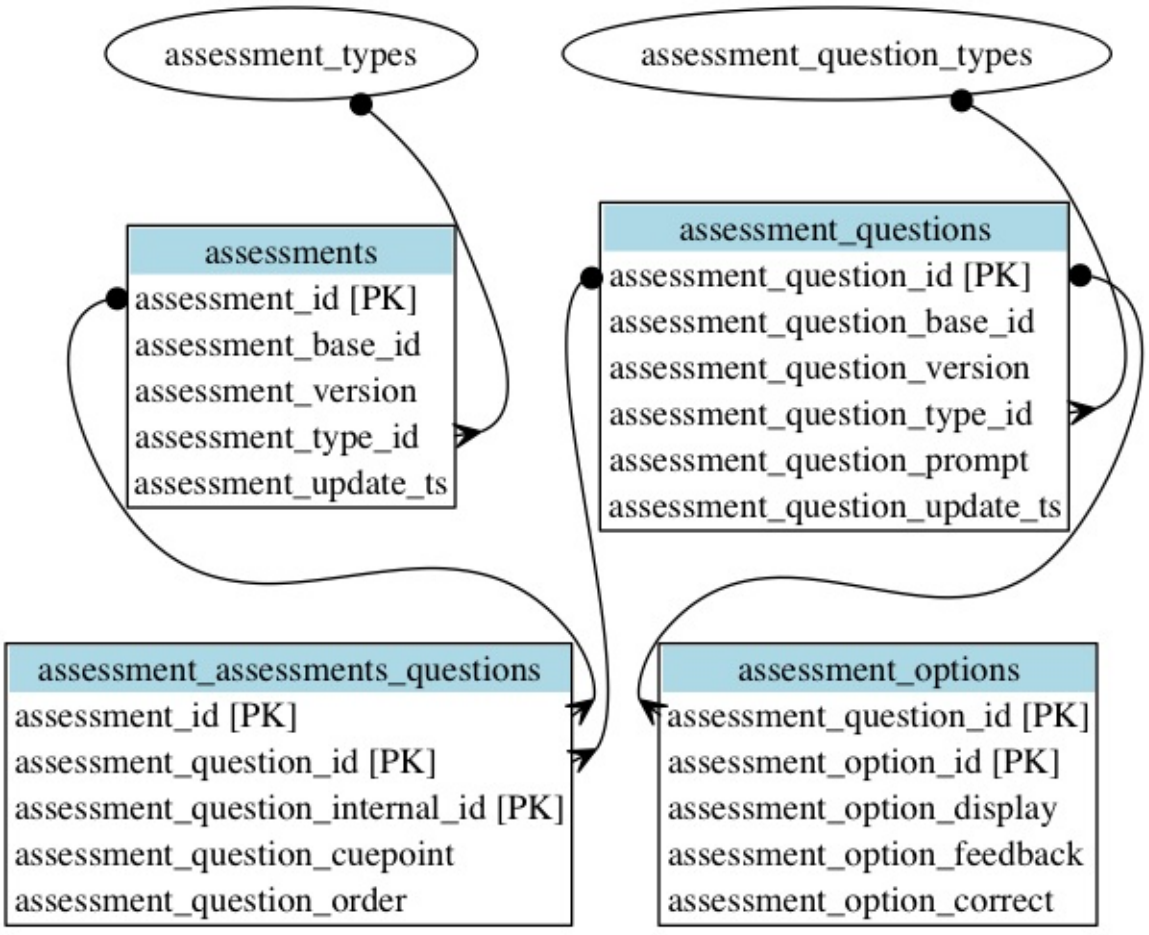
- A set of tables describe how the assessments are configured to test the learners. Each assessment has details such as when it was created, whether it is graded or ungraded, and what is the passing criteria. It also contains the questions and possible responses.
- A set of tables describe the learners' responses. For most assessments, these tables record the answers that learners submitted. Peer Assignments tables also includes the options chosen when learners was reviewing their peer's submissions.
- A combination of specific columns or additional tables has the populated values of assessment score.
- Each group has one table that allows the connection between these assessment tables to other data export tables, e.g. **course_item_assessments**

Quiz Tables

Quizzes are automatically-graded assignments used to test learner knowledge in a course. A quiz, referred to as an "assessment" in the data export tables, could be a multiple-choice exam or an in-video quiz. The **assessment_types** and **assessment_question_types** tables describe the type of quizzes and the type of questions, respectively.

The **assessments** table contains information for each quiz. The *assessment_base_id* field uniquely identifies the quiz within a course, and the *assessment_id* field is a combination of the *assessment_base_id* field and the *assessment_version*.

The **assessment_questions** table contains information for each question. Here, the *assessment_question_base_id* field uniquely identifies a question within a course, and the *assessment_question_id field* is a combination of the *assessment_question_base_id* field and the *assessment_question_version*.

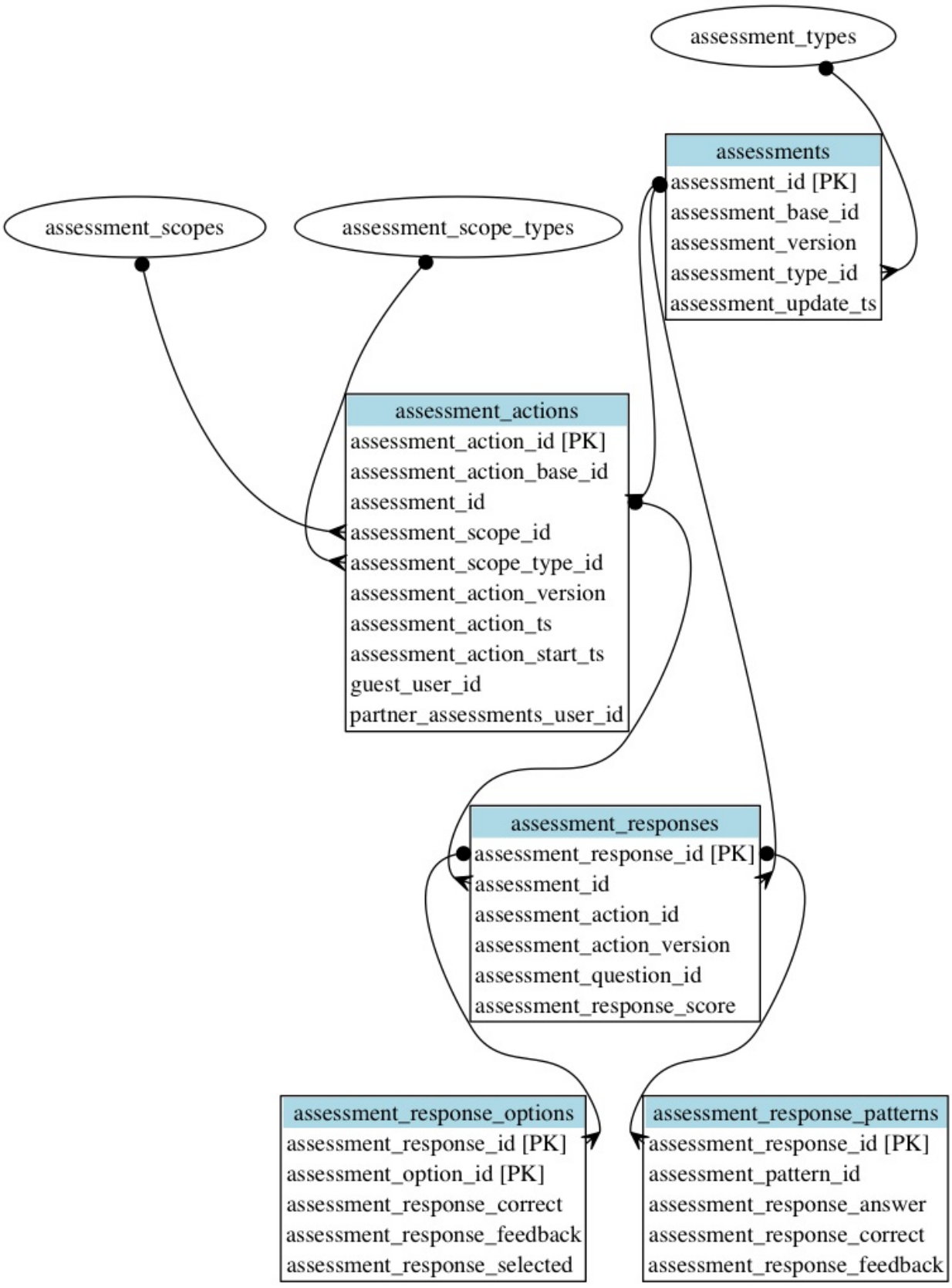


There are seven tables that correspond to different types of quiz questions, as summarized in the following table.

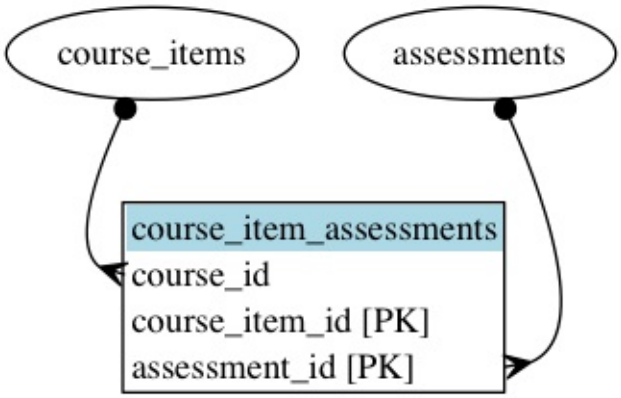
Question Type	Table Name
graded checkbox	**assessment_checkbox_questions**
ungraded checkbox	**assessment_checkbox_reflect_questions**
math expression	**assessment_math_expression_questions**
single numeric entry	**assessment_single_numeric_questions**
graded multiple-choice	**assessment_mcq_questions**
ungraded multiple-choice	**assessment_mcq_reflect_questions**
text matching	**assessment_text_exact_match_questions**

The **assessment_options** table describes the response options of each quiz question, how they are shown to learners, and whether they are correct.

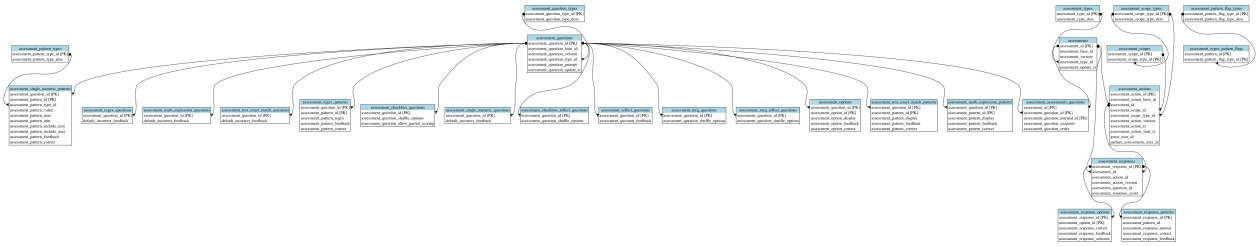
When learners attempt a quiz, their actions will be recorded in the **assessment_actions** table, and after they submit a response to the quiz, their responses will be recorded in the **assessment_responses** table. Learners can attempt a quiz as many times as they like. However, some may require them to wait a certain amount of time before resubmitting, and the highest-scoring attempt will be saved and will count as their final score for the quiz. The **assessment_response_options** table shows the option chosen by the learner, whether it is correct, and the feedback provided to the learner.



The *assessment_id* column can be joined with tables outside of the assessment tables by using the **course_item_assessments** table.



Below is our best attempt to build one ER diagram for all tables with the **assessments_*** prefix.



> click image to enlarge

SQL Example: Which choices did learners select in the in-video quizzes?

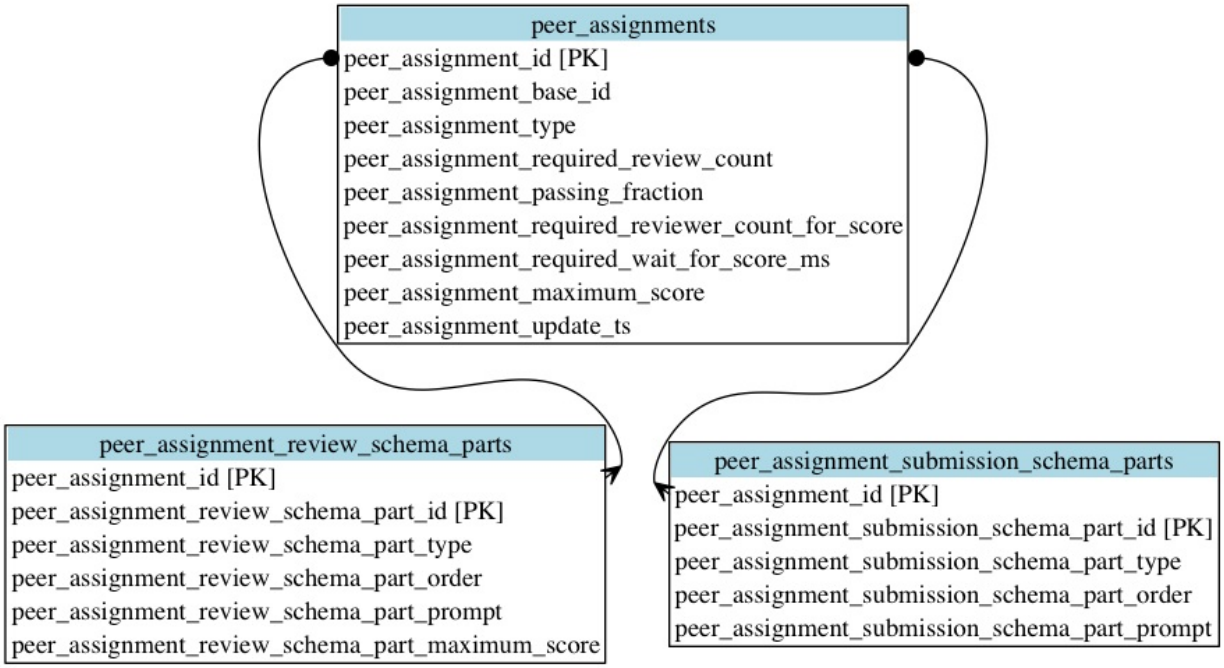
The SQL below illustrates how the multiple assessment tables relate across assessment questions, actions, and responses. It returns the list of each in-video quiz, each quiz question, the lecture it is associated with, what/when the user responded, and the response score.

```
SELECT ci.course_item_name AS lecture_title
   ,aq.assessment_question_prompt
   ,aa.[partner]_assessments_user_id
   ,ao.assessment_option_display
   ,aa.assessment_action_ts
   ,ar.assessment_action_version
   ,ar.assessment_response_score
FROM course_item_assessments cia
-- add assessments table to get the assessment_type_id column
JOIN assessments
    USING(assessment_id)
-- to allow filter for only in-video quizzes
JOIN assessment_types
    USING(assessment_type_id)
-- to get lecture title
JOIN course_items ci
    ON(ci.course_id = cia.course_id
        AND ci.course_item_id = cia.course_item_id)
-- to get assessment_action_id and assessment action timestamp
JOIN assessment_actions aa
    ON(aa.assessment_id = cia.assessment_id)
-- to get assessment_action_version, response_options, and assessment_response_score
JOIN assessment_responses ar
    ON(ar.assessment_id = aa.assessment_id
        AND ar.assessment_action_id = aa.assessment_action_id)
-- next two joins are to get assessment_option_display
JOIN assessment_response_options aro
    ON(aro.assessment_response_id = ar.assessment_response_id)
JOIN assessment_options ao
    ON(ao.assessment_question_id = ar.assessment_question_id
        AND ao.assessment_option_id = aro.assessment_option_id)
-- to get assessment_question_prompt
JOIN assessment_questions aq
    ON(aq.assessment_question_id = ao.assessment_question_id)
WHERE assessment_type_desc = 'in video'
-- only return what the user selected
    AND assessment_response_selected = TRUE;
```

Peer Assignments Tables

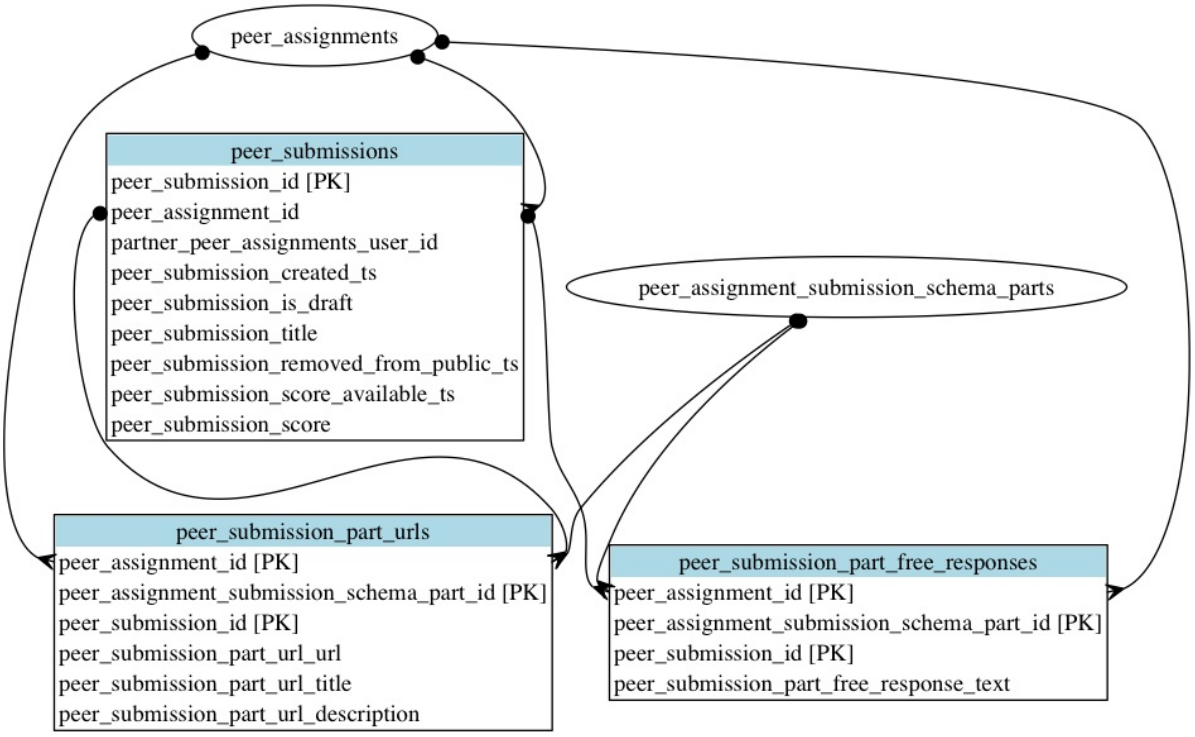
Peer assignments allow learners to grade assignments of other learners. It is composed of instructions, prompts, and rubric parts, each of which is described in Partner Help Center article on Peer Review Assignments.

The **peer_assignments** table contains details on how the instructor has configured the type (e.g., "graded"), the scoring criterion (e.g., a minimum number of reviews), and other information. The instructor also configures the questions (or "parts") and question types in the peer assignment, found in the **peer_assignment_submission_schema_parts** table. Lastly, there is a rubric for reviewers, found in the **peer_assignment_review_schema_parts** table.

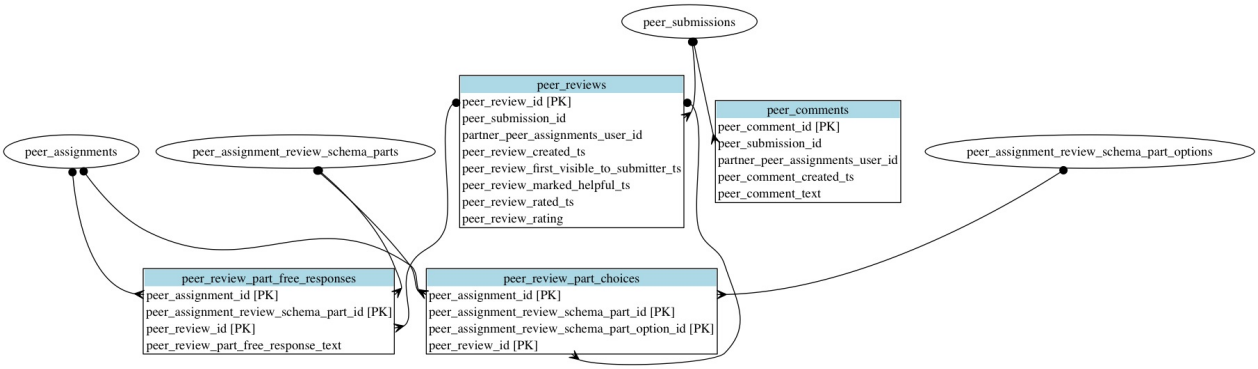


The **peer_submissions** table is populated when learners are taking the peer assignments. They create draft submissions when they click the "save" button. Each draft that a user saves is stored as a separate submission. The answers that learners submit are recorded in the **peer_submission_part_free_responses** and **peer_submission_part_urls** tables.

When a user clicks the "submit" button, this creates a completely new entry in addition to the previous drafts. The *peer_submission_is_draft* field distinguishes saved drafts from the final submission.



The **peer_reviews** table is populated when learners review their classmates' assignments. Unlike submissions, there are no draft reviews, and reviews cannot be deleted or modified. Each reviewer may submit at most one review per submission. The evaluation choices and responses that reviewers submit are recorded in the **peer_review_part_choices** and **peer_review_part_free_responses** tables. A reviewer can also provide comments to submissions, which are stored in the **peer_comments** table.

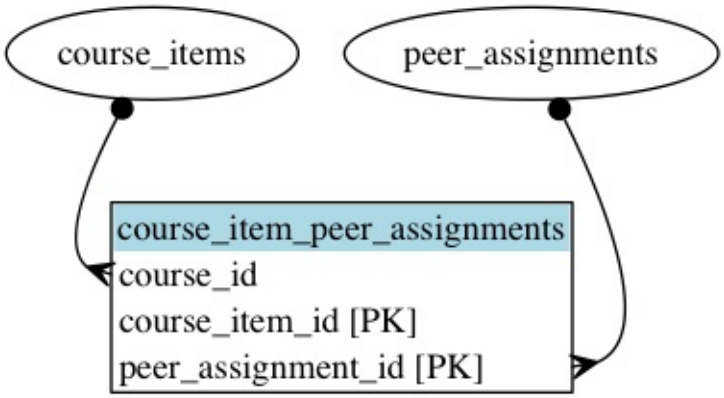


Reviewers may also decline to review a submission by providing a reason or by selecting one of the preset options such as "inappropriateContent", "incompleteSubmission", or "plagiarism". These are recorded in the **peer_skips** table (omitted from the diagram above).

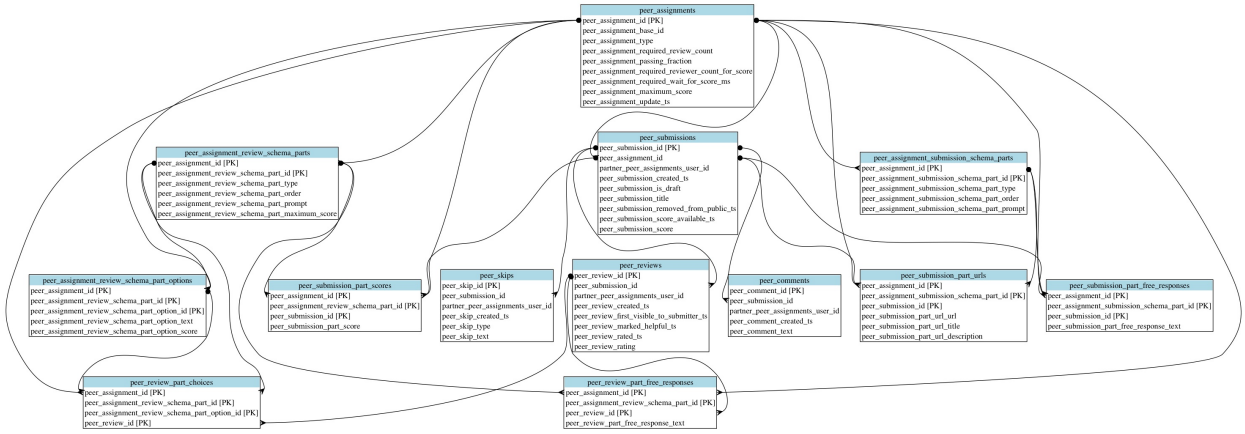
When a peer submission's scoring criteria are met, grades are generated in the **peer_submission_part_scores** table. The part score is the median of the scores that each review gives for that part. A submission's score is the sum of its part scores. To see each

reviewer's score for each part of a submission, use the **peer_assignment_review_schema_part_options** table.

The *peer_assignment_id* column can be joined with tables outside of the peer assignments tables by using the **course_item_peer_assignments** table.



Below is our best attempt to build one ER diagram for all tables with the **peer_*** prefix.



> click image to enlarge

SQL Example: What were the peer review responses in peer submissions?

The SQL below illustrates how the multiple peer assignments tables relate across peer assignment questions, submissions, and received peer reviews. It returns the list of each submission to a peer review question, submission time, submitted content, and each review received for each part of the submission.

```
SELECT ci.course_item_name
```

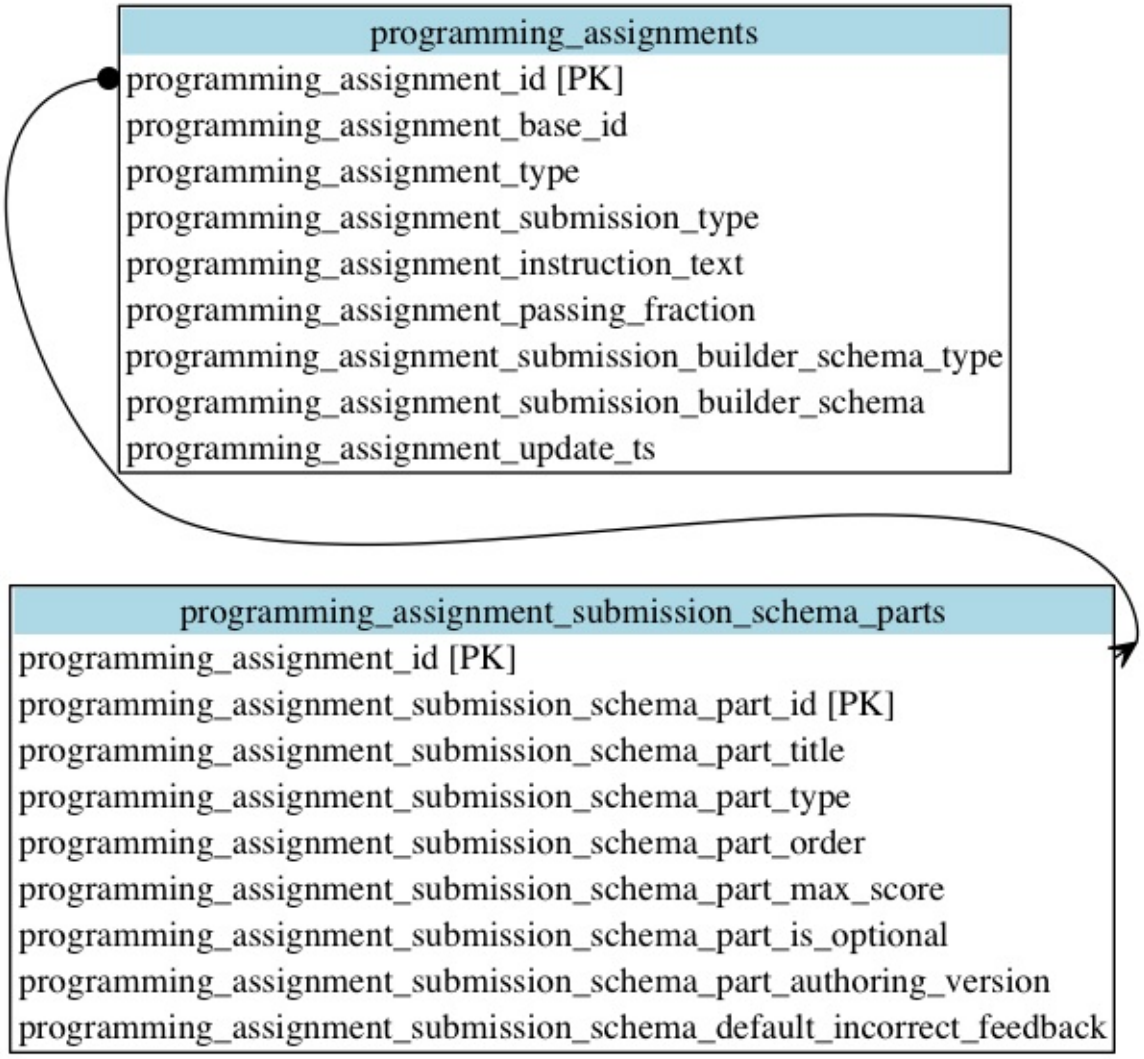
```
    ,ps.[partner]_peer_assignments_user_id
    ,ps.peer_submission_created_ts
    ,ps.peer_submission_title
    ,ps.peer_submission_score
    ,pspr.peer_submission_part_free_response_text
    ,psp.peer_submission_part_url_url
    ,psp.peer_submission_part_url_title
    ,psp.peer_submission_part_url_description
    ,pr.user_id AS reviewer_id
    ,par.peer_assignment_review_schema_part_prompt
    ,parsp.peer_assignment_review_schema_part_option_text
    ,parsp.peer_assignment_review_schema_part_option_score::VARCHAR(50)
    ,prfr.peer_review_part_free_response_text
    ,prpc.peer_assignment_review_schema_part_option_id
FROM course_item_peer_assignments cip
-- to get the item description
JOIN course_items ci
    ON ci.course_item_id = cip.course_item_id
    AND ci.course_id = cip.course_id
-- to get the submitter's user_id, submission timestamp, and score
JOIN course_item_types cit
    ON cit.course_item_type_id = ci.course_item_type_id
JOIN peer_submissions ps
    ON ps.peer_assignment_id = cip.peer_assignment_id
-- to get the free text submission information of the submitted work if applicable
LEFT JOIN peer_submission_part_free_responses pspr
    ON ps.peer_submission_id = pspr.peer_submission_id
    AND ps.peer_assignment_id = pspr.peer_assignment_id
-- to get the url information of the submitted work if applicable
LEFT JOIN peer_submission_part_urls psp
    ON ps.peer_submission_id = psp.peer_submission_id
    AND ps.peer_assignment_id = psp.peer_assignment_id
-- to get information of the received reviews for each submission
LEFT JOIN peer_reviews pr
    ON ps.peer_submission_id = pr.peer_submission_id
-- the following two joins get the reviewer's assessment rubric
JOIN peer_assignment_review_schema_parts par
    ON par.peer_assignment_id = ps.peer_assignment_id
LEFT JOIN peer_review_part_free_responses prfr
    ON prfr.peer_assignment_id = par.peer_assignment_id
    AND prfr.peer_review_id = pr.peer_review_id
    AND par.peer_assignment_review_schema_part_id = prfr.peer_assignment_review_schema_pa
-- the following two joins get the review scores given to each assessment rubric
LEFT JOIN peer_review_part_choices prpc
    ON prpc.peer_assignment_id = par.peer_assignment_id
    AND prpc.peer_review_id = pr.peer_review_id
    AND par.peer_assignment_review_schema_part_id = prpc.peer_assignment_review_schema_pa
LEFT JOIN peer_assignment_review_schema_part_options parsp
    ON parsp.peer_assignment_id = prpc.peer_assignment_id
    AND parsp.peer_assignment_review_schema_part_id = prpc.peer_assignment_review_schema_
    AND parsp.peer_assignment_review_schema_part_option_id = prpc.peer_assignment_review_
WHERE course_item_type_desc ILIKE '%peer%';
```

Programming Assignments

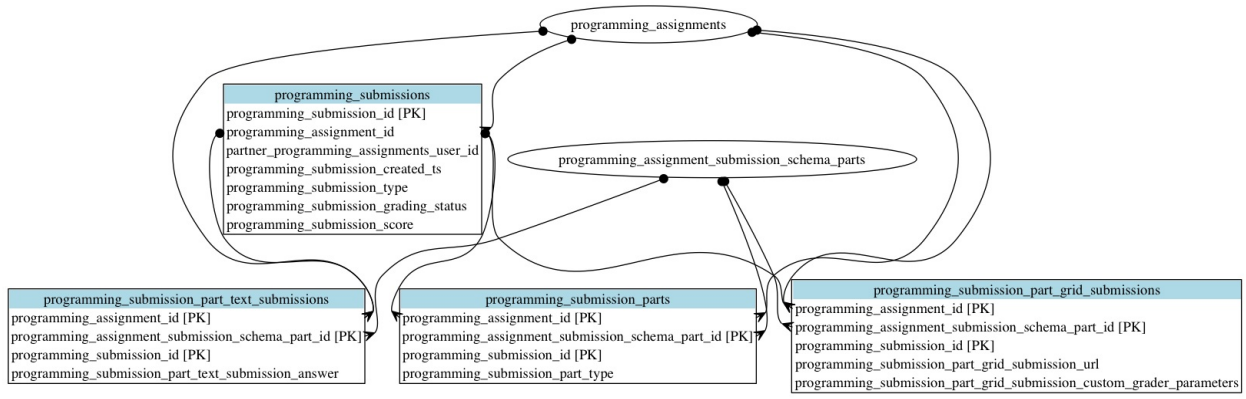
Programming Assignments are assignments that require learners to write and run a computer program. There are two types of assignments based on when they are graded after submission: 1) synchronously graded questions that are described in tables containing **_text_** in their names, and 2) asynchronously graded questions that are described in tables containing **_grid_** in their names.

The **programming_assignments** table gives you information on each programming assignment, assignment type (graded vs. ungraded), instructions, passing criterion, and creation time. The *programming_assignment_base_id* field uniquely identifies a programming assignment within a course, and the *programming_assignment_id* field combines the *programming_assignment_base_id* field and the version of the assignment.

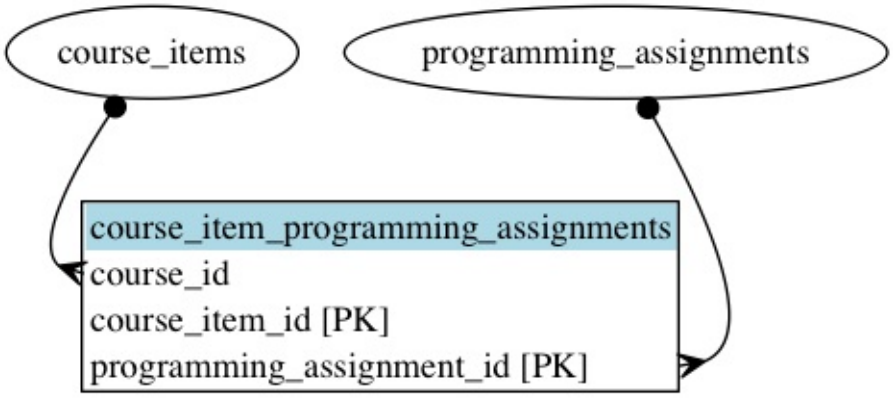
Each part of a programming assignment can be found in the **programming_assignment_submission_schema_parts** table. Possible responses to a part and their orders can be found in the **programming_assignment_submission_schema_part_possible_responses** table.



After learners submit their answers to a part of a programming assignment, their answers and scores are recorded in the **programming_submission_parts** table. If you are interested in learner submissions to asynchronously graded parts, refer to the **programming_submission_part_grid_submissions** table. If you are interested in learner submissions to synchronously graded parts, refer to the **programming_submission_part_text_submissions** table. The **programming_submissions** table tells you the aggregated submission information of the programming assignment by the learner.



The *programming_assignment_id* column can be joined with tables outside of the programming assignments tables by using the **course_item_programming_assignments** table.



SQL Example: What were learners' scores across programming submissions?

The SQL below illustrates how the multiple programming assignment tables relate across assignment questions, submissions, and received scores. It returns the list of each programming question, learner submission, the status of grading the submission, and the received score.

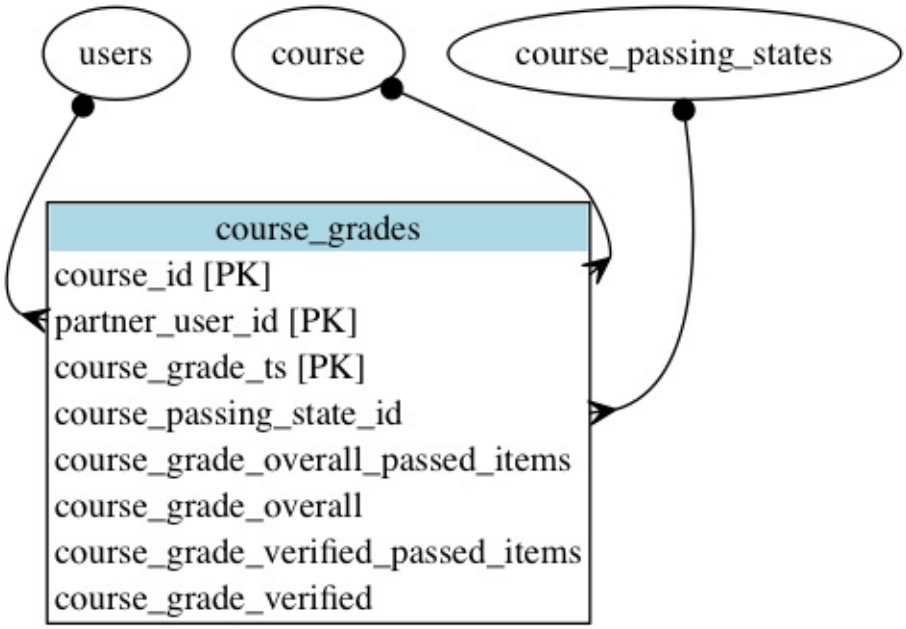
```
SELECT course_item_name
    ,[partner]_programming_assignments_user_id
    ,programming_submission_created_ts
    ,programming_submission_grading_status
    ,programming_submission_score
FROM course_item_programming_assignments cip
JOIN course_items ci
    ON(ci.course_id = cip.course_id AND ci.course_item_id = cip.course_item_id)
JOIN programming_assignments pa
    ON(cip.programming_assignment_id = pa.programming_assignment_id)
JOIN programming_submissions ps
    ON(ps.programming_assignment_id = pa.programming_assignment_id);
```

Course Grades Tables

The three **Course Grades** tables below are populated when learners receive grades or complete the course:

- **course_grades:** tracks how learners are completing course items over time.
- **course_item_grades:** tracks the grade for each course item that are mandatory for course completion.
- **course_formative_quiz_grades:** includes grades for items that not mandatory for course completion; this table is mutually exclusive to the **course_item_grades**.

The **course_grades** table describes whether learners have passed a course. Each row records the grading event when the learner reached their highest grade in the course. It also includes the date and time of that event, how many items were passed at that time, and whether the learner had reached the passing state. The **course_passing_states** tables provides the descriptions of the passing states.



The **course_item_grades** table has very similar logic to the **course_grades** table above; each row contains a learner's highest grade for each course item attempted and whether the learner passed or not. For example, if a learner completed a course with four mandatory items, then this table would contain four rows for this learner, each with a passing state, and the date and time when the item was passed.

The **course_formative_quiz_grades** table has similar logic to the other table above, with the exception that there is no column to denote passing state, since these quizzes are not mandatory for course completion.

Discussion Tables

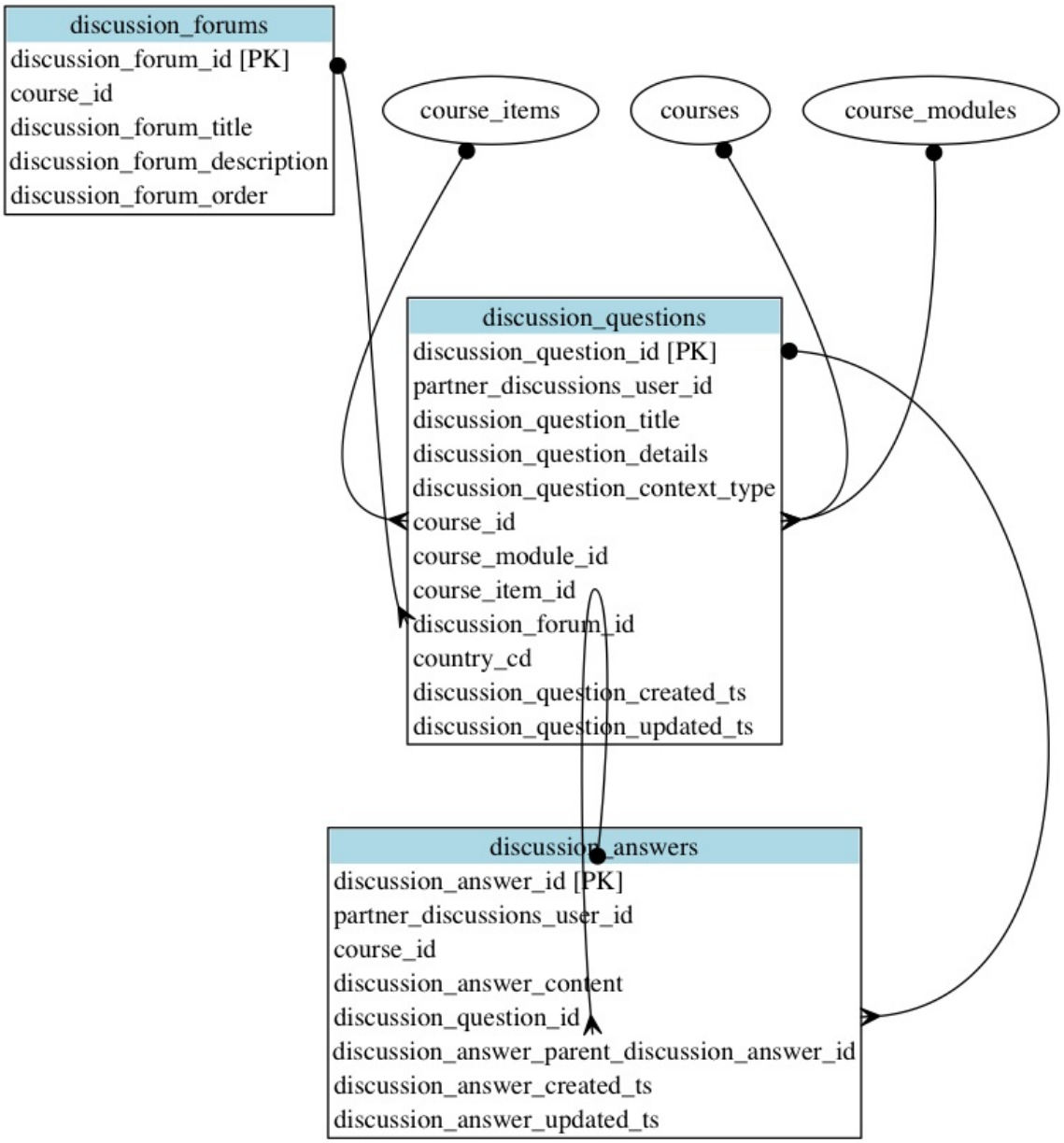
Course discussion forums provide a space for learners to interact, share resources, and help one another with questions about the course materials and assignments. More info can be found in our Partner Help Center article on Discussion Forums.

Across all the tables prefixed with **discussions_***, the main three are:

- **discussion_forums**: lists the forums (e.g. General Discussions, Meet and Greet, etc.)
- **discussion_questions:** includes the original posts in each thread, the author, and the thread's placement within the course
- **discussion_answers**: lists the replies to the threads

All other discussion tables focus on followings, votes, flags (not shown in ER diagram below):

- **discussion_answer_flags**: records when a learner flags an answer for the Coursera community operations team to review, because the posts do not follow community guidelines (offensive or spam)
- **discussion_questions_flags**: records when a learner flags a question for the Coursera community operations team to review, because the posts do not follow community guidelines (offensive or spam)
- **discussion_answer_votes**: records when an answer is upvoted (and when a user "revokes" their upvote)
- **discussion_questions_votes**: records when a discussion question is upvoted
- **discussion_questions_followings**: records when a discussion question is followed or unfollowed


discussion_forums
discussion_forum_id [PK]
course_id
discussion_forum_title
discussion_forum_description
discussion_forum_order
course_items
courses
course_modules
discussion_questions
discussion_question_id [PK]
partner_discussions_user_id
discussion_question_title
discussion_question_details
discussion_question_context_type
course_id
course_module_id
course_item_id
discussion_forum_id
country_cd
discussion_question_created_ts
discussion_question_updated_ts
discussion_answers
discussion_answer_id [PK]
partner_discussions_user_id
course_id
discussion_answer_content
discussion_question_id
discussion_answer_parent_discussion_answer_id
discussion_answer_created_ts
discussion_answer_updated_ts

Feedback Tables

The Feedback tables are populated as learners flag course content, like/dislike course content, or rate courses. The main two tables are **feedback_course_ratings** and **feedback_item_ratings**.

The **feedback_course_ratings** contains two different sets of course ratings, which can be distinguished based on the *feedback_system* column. The value of "STAR" is for the system of 5-star ratings for the entire course.

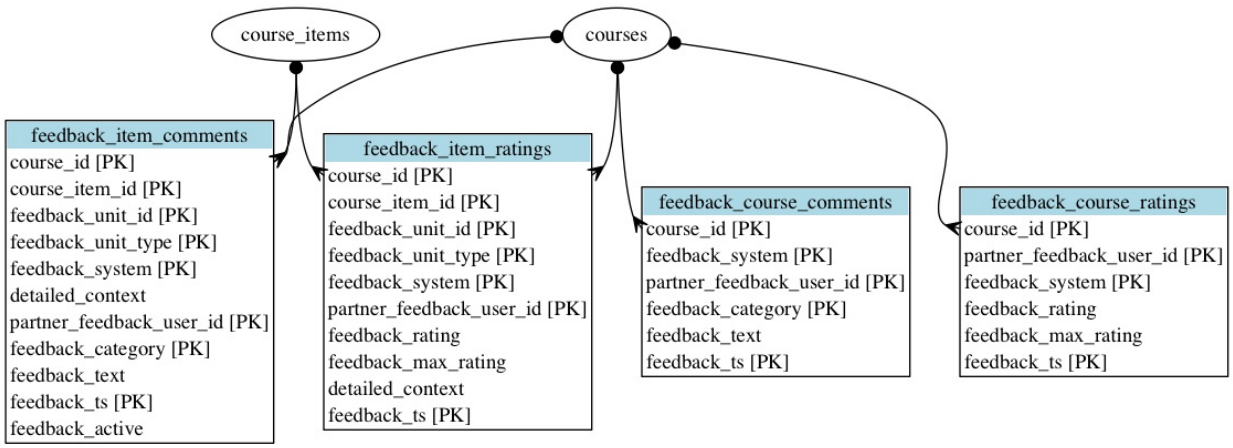
As of the spring of 2016, Coursera is surveying a measurement called Net Promoter Score (NPS) to better understand how satisfied learners are with courses overall. We ask learners, "How likely are you to recommend this course to a friend or colleague?", provide a rating scale between 1 to 10. Net Promoter Score is calculated as the percentage of Promoters (those selecting a rating of 9 or 10) minus the percentage of Detractors (those selecting a rating of 6 or below).

In the *feedback_system* column, the new values for NPS results are "NPS_FIRST_WEEK" and "NPS_END_OF_COURSE". For any data researcher who does not filter on one *feedback_system*, such as "STAR", the results will ***erroneously aggregate scores from multiple systems***.

The **feedback_item_ratings** table contains the "like" or "dislike" ratings for a course item.

Although a learner can rate the same course (item) multiple times, the data provided in exports is filtered to only provide the most recent rating per learner-course (learner-item) pairing.

There are also two feedback tables that log the free-text responses. The **feedback_item_comments** contains comments from learners who flag a course content. The **feedback_course_comments** contains learners' comments about the course in both star and NPS feedback systems. Both of these tables contain the entire history of comments to your course, not just the most recent.



SQL Example: How does my overall course 5-stars rating vary across months?

```
SELECT
    course_id
    ,TO_CHAR(feedback_ts, 'YYYY-MM')
    ,COUNT(*) AS num_feedback
    ,AVG(feedback_rating::FLOAT) AS avg_feedback_rating
FROM feedback_course_ratings
-- Ensure that only one feedback system is used.
WHERE feedback_system = 'STAR'
GROUP BY 1,2
ORDER BY 1,2;
```

Learner Tables

The Learner tables contain information about Coursera's registered accounts.

The **users** table lists all learners that are related to your course and export. There is additional information on learner's browser language preference and the most current country inferred by the user's browser IP via the Maxmind geolocation service. Per Coursera's Data Sharing Policy and Research Policy, this table does not include personally identifiable information (PII).

The **course_membership** table logs when a Coursera user gets assigned to a membership role in the course. The most common values are:

- "LEARNER": users that clicked the Enroll button on an already-launched course
- "PRE_ENROLLED_LEARNER": users that pre-enrolled in a course or session
- "NOT_ENROLLED": users that unenrolled in the course for any reason
- "MENTOR": users that are invited and then accepted to moderate course discussions
- "BROWSER": users that previewed content on the course description page, but has not clicked on the Enroll button

The other possible values are: "UNIVERSITY_ADMIN", "TEACHING_STAFF", and "INSTRUCTOR".

A single learner may have had multiple roles in a course over time; this learner will have multiple records in the table. For example, a learner could have watched the preview content ("BROWSER"), then hit the enroll button to join ("LEARNER"), and then unenrolled a few days later ("NOT_ENROLLED").

This table only contains membership information for users that ever reached the "LEARNER" role. For example, it does not contain any only-"BROWSER" users that previewed course material but never hit the Enroll button.

There is an export table with the name **[partner]_course_user_ids**. It has two id columns:

- *[partner]_user_id*: consistent with other user ID(s) in the data export
- *[course_slug]_user_id*: used in marketing or surveying efforts

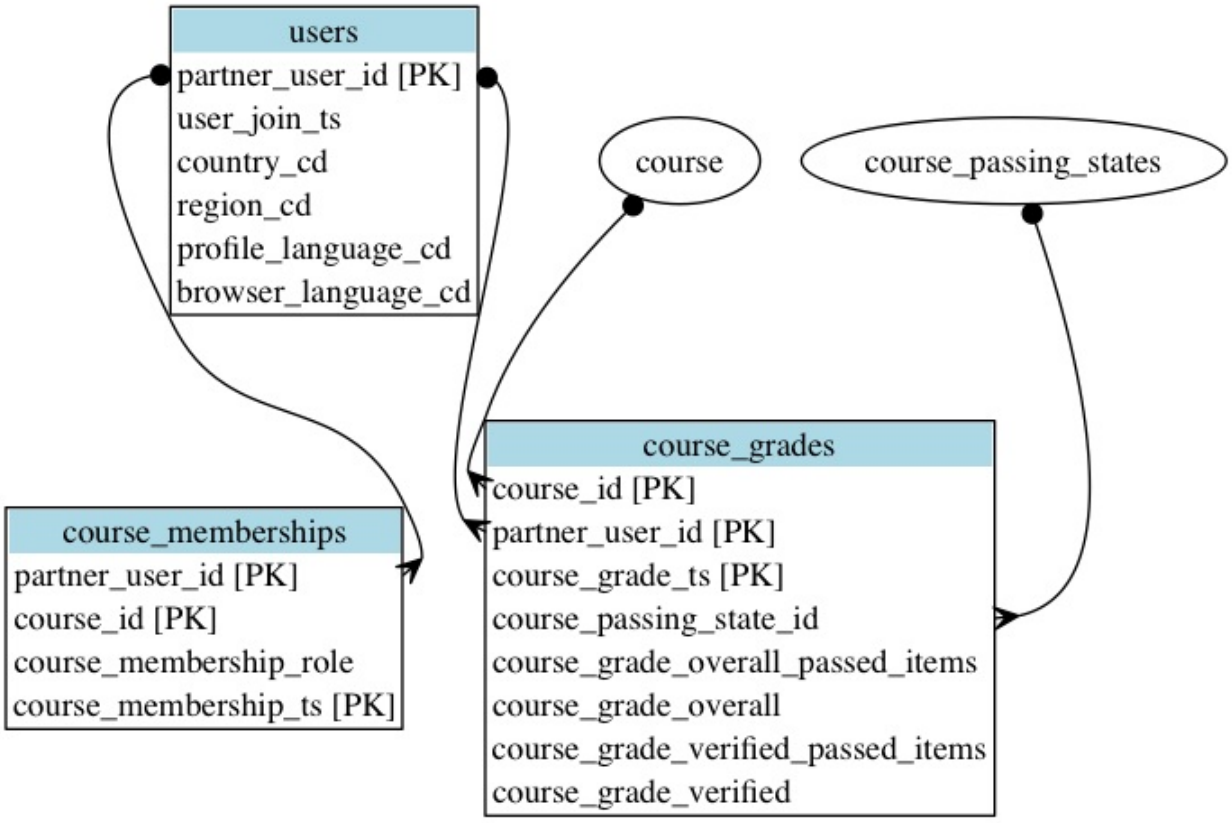
This table is very useful for instructors to administer surveys to learners, outlined in this Partner Help Center article on External Surveys. See the ID Columns chapter for more details.

SQL Example: How many learners, pre-enrollers, and completers do I have from each country?

```
-- learners who pre-enrolled
WITH course_pre_enrollers AS (
    SELECT DISTINCT [partner]_user_id
    FROM course_memberships
    WHERE course_membership_role = 'PRE_ENROLLED_LEARNER'
)

-- learners who passed the course
, course_completers AS (
    SELECT DISTINCT [partner]_user_id
    FROM course_grades
    WHERE
        -- 1 = passed, 2 = verified passed
        course_passing_state_id IN (1,2)
)

-- by country_cd (e.g. 'US'), provide counts of users
SELECT
    country_cd
    ,COUNT(*) AS num_course_learners
    ,COUNT(course_pre_enrollers.[partner]_user_id) AS num_pre_enrollers
    ,COUNT(course_completers.[partner]_user_id) AS num_completers
FROM users
LEFT JOIN course_pre_enrollers
    USING ([partner]_user_id)
LEFT JOIN course_completers
    USING ([partner]_user_id)
GROUP BY 1
ORDER BY 2 DESC;
```


users
partner_user_id [PK]
user_join_ts
country_cd
region_cd
profile_language_cd
browser_language_cd
course
course_passing_states
course_grades
course_id [PK]
partner_user_id [PK]
course_grade_ts [PK]
course_passing_state_id
course_grade_overall_passed_items
course_grade_overall
course_grade_verified_passed_items
course_grade_verified
course_memberships
partner_user_id [PK]
course_id [PK]
course_membership_role
course_membership_ts [PK]

Demographic Tables

At any time, learners can take a voluntary Coursera survey (pdf format) in which they volunteer answers to questions such as:

- In what country do you currently live?
- What is your gender?
- What is the highest level of school you have completed or the highest degree you have received?
- Which of the following best describes your current employment status?
- Besides English, what other languages do you speak?

The complete list of questions is contained in the **demographics_questions** table. The **demographics_answer** table contains the answers from Coursera learners.

This survey contains responses from about 300k Coursera users from 2013 to March 2015. The data export only contain those users who enrolled in your course and volunteered to answer the survey.

Frequently Asked Questions

Can I query for the same learners across courses on the new platform?

Yes, when using data exports across multiple courses, the user ID's across the different CSV files are consistent for one learner.

As an example, if your Specialization consists of four launched courses and an upcoming capstone, you could:

1. generate a data export for each of the courses
2. unzip each export, and view/load each course_grades.csv
3. query for those learners who reached a passing state in each course
4. query to count the number of learners who have passed all four courses, who will be enabled to enroll in the upcoming capstone

What data is absent from data exports on the New Platform?

Here are some data sources which we have not included in the data exports.

- Lecture Downloads: Data exports do not provide information when a learner downloads a lectures video. The **course_progress** table logs learner activity with streamed video, but not downloaded video."
- Video "Heartbeats": Data exports do not provide information of learners' actions while watching a lecture video, such as pauses, rewinds, etc.
- Discussion Views: Data exports do not provide page view information in the discussion forum which could, for example, be useful for identifying the "most viewed discussion question". Potential proxy metrics that are available in the exports include answers, followings, and votes.