Project Demo Protocol (Tentative)—Student Version
Fall 2018

Last Update: November 10, 2018 @ 22:00

1) Each TA books half-hour slots; we'll aim for a demo of 20-25 minutes. The extra 5 minutes can be used as turnaround time for the next group to get settled after the current group clears out.
    - The TAs will create a GoogleDoc to list their available times. The groups should select a time from that list, or seek a mutually agreeable time. It may be difficult to find a perfect time; so, groups will have to be accommodating.

2) The next group should be ready to go before meeting with the TA; otherwise, it cuts into the demo time. In other words, students should *not* use the first part of the demo to boot their laptop and go searching for all the files they plan to use. It will help to have the SQL scripts available to DROP, CREATE, INSERT rows, etc.

3) Right away, the TA asks the group if anything that's in their formal specs (list of deliverables) is NOT working. This will save a bunch of demo time, and it will help the TA with marking the Canvas-submitted deliverables.

4) Before starting on the checklist, the TA will ask students to show that their code has not been changed since submission on Canvas. They should re-compile the code, or prove the timestamps. They should create and populate the tables.

5) The group shows the TA that everything is working (or whatever they've got done). The TA sees that the system is working by viewing the interface, the user's input (or selection from a list of features/queries), and the output that gets nicely displayed. In the case of an INSERT, UPDATE, or DELETE, the TA sees the "before" state of the table (students can use their program, or SQL*Plus or equivalent, to display it) and the "after" state, etc.

    - It may be helpful for the project groups to provide a copy of the schema and sample data in the tables, so that the TA can verify that the queries are producing the right output, and what to expect from any UPDATE or DELETE operation.

    - Tip to Project Groups: Imagine that you are taking a road test for a driver's licence. The TA will ask you to walk through one query at a time, type in query inputs (if appropriate), and check the relation instance to see if the right behaviour is observed. The TA can ask the students to use SQL*Plus (or equivalent) to show the relation instance. Students do not do not need to implement any "print" function in their application.

6) Students should take the lead in the demo by showing the things that are required by our demo checklist (see the big table below). So, it will be a "mixed" type of approach. The group can show all the queries, etc., and the TAs can ask questions and have them modify something, and rerun, etc.

7) Students will be asked an SQL division question, which they should prepare in advance.
    - Students should think of a useful division question for their application, and then demonstrate it to the TA. It can be coded as part of their application code, or it can simply be copied and pasted into SQL*Plus (or equivalent).
    - Then, the TA asks the group to INSERT or DELETE a row, and confirm that the division query no longer produces the same output.

8) The TA can ask the group one or two technical questions. Questions might be: How would you go about changing "this" to "that"? In other words, how would you go through the steps from edit to compile to output ... in order to implement a particular change like a new query, a query with more/fewer fields of output, etc? The group can describe it in words. It may be too time-consuming and pressure-packed to have students do changes on the fly, and it might mess up part of their demo, if it's not asked at the end.

9) The demo will probably be marked out of a number like 12, but we won't give the grade out on the spot. The TAs will need to reflect upon the presentation afterwards after:
    - Checking and reflecting upon their own notes
    - Comparing the differences between the demo and the actual deliverables handed in to Canvas
    - Comparing the differences between the demo and the formal specifications
    - Deciding whether some groups were asked harder/easier questions
    - Consulting the other TAs and instructors when comparing different groups' demos and deliverables, when it comes to bonus marks
    - etc.

10) We're offering a small number of **bonus marks** for the demo part of the project, and the bonuses will be 1 or 2 marks out of something like 12. **Our standards are high for this.** Group won't automatically get the bonus. The maximum bonus is 2 points. Recall that the weighting of the demo is about 31% of the project grade.

    - Having a really fancy GUI is worth 1 point.
    - Implementing and demonstrating useful triggers is worth 1 point (unless the implementation is trivial).
    - Having some "wow" type of feature in their application is worth 1 point.
    - TAs will discuss the bonus points with other TAs and instructors to see whether the bonus should apply to a given situation.

Here is a tentative marking scheme (we may still adjust it, but we'll probably keep it in the 12 mark range), and it gives you an idea of approximately what we're after. The TAs won't tell you ahead of time whether a given feature is "correct" or is worth a mark; so, please don't ask. **The groups can change the order of the items below** because in some cases the SELECT, INSERT, UPDATE, DELETE, aggregation, view, etc. may depend on some other SQL statements.

Component	Details	Number of Points
GUI	• We are accepting a text-based interface, as long as it is clearly laid out and isn't missing any choices. For those who used an actual GUI, it doesn't need to be fancy. Really fancy GUIs can be awarded a bonus point (see below). ○ 1 points = a good GUI or text interface ○ 0.5 points = something is missing ○ 0 points = needs work	1
Same Deliverables as Handed In?	• The group shows the TA that the deliverables haven't changed from what they handed in (e.g., timestamps). ○ Marks will be deducted appropriately in the final deliverables, if there is a difference.	1
Selection Query (Deliverable 8-10)	• The group picks one query from this category. The group should show the *before* and *after* case (either by using SQL*Plus or their application program). ○ 1 = Worked correctly ○ 0 = Incorrect or missing add new equipment: D9	1
INSERT (Deliverable 2)	• Did their *program* (not SQL*Plus or equivalent) perform a successful INSERT statement? The group should show you the *before* and *after* case (either by using SQL*Plus or their application program). ○ 1 = yes ○ 0 = no make a reservation D8	1
DELETE (Deliverable 3)	• Did their *program* (not SQL*Plus or equivalent) perform a successful DELETE statement? The group should show you the *before* and *after* case (either by using SQL*Plus or their application program). ○ 1 = yes ○ 0 = no delete a reservation	1
UPDATE (Deliverable 4)	• Did their *program* (not SQL*Plus or equivalent) perform a successful UPDATE statement? The group should show you the *before* and *after* case (either by using SQL*Plus or their application program). ○ 1 = yes ○ 0 = no udpate equipment	1
Join Query (Deliverable 5 or 6)	• The group picks one query from this category. The group should show you the *before* and *after* case (either by using SQL*Plus or their application program). show customer schedule: join reservation, class, classtype	1

	○ 1 = Worked correctly ○ 0 = Incorrect or missing	
View (Deliverable 11)	• Did they create a VIEW (either by SQL*Plus or equivalent or their program), and then use the *program* to successfully access that view? ○ 1 = Worked correctly ○ 0 = Incorrect or missing	1 create view for class_count_view
Aggregation Query (Deliverable 7)	• The group demonstrates the use of aggregation (e.g., `min`, `max`, `sum`, `average`, or `count`). ○ 1 = Worked correctly ○ 0.5 = It is a GROUP BY, but fairly trivial and not an "interesting" example ○ 0 = Incorrect or missing	1 aggregated count of classes, group by classtype
Division Query (new, but advertised deliverable)	• The group demonstrates the correct use of SQL division, including showing that the result changes after an INSERT or DELETE (which produces a different, but correct outcome). This doesn't necessarily have to be done in the program; it can be done dynamically (e.g., via SQL*Plus or equivalent). ○ 1 = Worked correctly ○ 0 = Incorrect or missing	1
Users' Data	• Was the data in their tables sufficient to demonstrate their project well? ○ 1 = yes ○ 0.5 = mostly, except for a small number of needed rows (that are missing) ○ 0 = no, it needs more than a few more rows of data	1

Other Demo Items and Technical Questions	• This is about the rest of the demo, and how smoothly things went.
	• It could vary between project groups, so there is no specific grading scheme for most of this part.
	• This section includes any technical questions asked to a student or the group, like "how would you go about making this kind of a change"?
	  • If a student didn't answer a question that you think he/she should have answered, then you can deduct from just that student's score (but do this in the section below).
	  • Did anyone else in the group know the answer to a question that the first person should have answered? If *no one* could answer it, then give zero for the whole group for this component.
		1

Group Deductions	• Was the group prepared for their demo, or did they waste time getting started? ○ 0 points = ready to go	– ____

	○ -1 points = spent too much time getting started (we told them to be ready, on the demo instructions)	
Bonus Marks	Create unique features for your application that clearly go over and above what was expected. Some examples include a really fancy GUI, implementing useful trigger(s) and having application logic that deals with it, produces special reports, has a "wow" factor, etc. Keep in mind these bonus points have a HIGH standard to meet. Not all groups will get bonus marks; possibly only a few will get 2 bonus marks.	+ ____
Unofficial GROUP MARK	**OUT OF 12**	**___ / 12**
Individual Deduction	Did the student fail to show up at the demo, or show up late? Make a note of the student's name. • If not there, subtract 3 points. • If more than 15 minutes late, subtract 2 points. • If 5-15 minutes late, subtract 1 point.	− ____
Individual Deduction	• Was an individual student not able to answer a question (as listed above in "Other Demo Items and Technical Questions")?	− ____
Unofficial INDIVIDUAL MARK	**OUT OF 12**	**___ / 12**