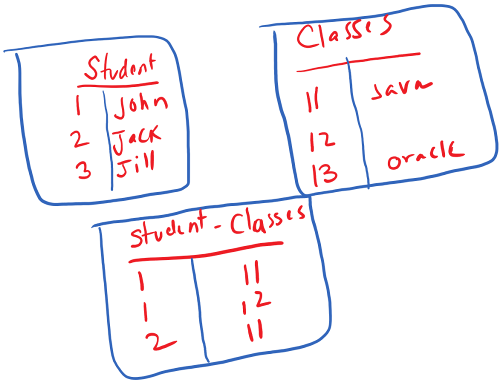
Join Homework

Here are some hints for each of the questions in the join homework

To best approach this homework consider the following dataset. All of my suggestions, below, will be based on this dataset.



Student

| | |
|---|---|
| 1 | John |
| 2 | Jack |
| 3 | Jill |

Classes

| | |
|---|---|
| 11 | Java |
| 12 | |
| 13 | oracle |

Student - Classes

| | |
|---|---|
| 1 | 11 |
| 1 | 12 |
| 2 | 11 |

Notice that Jill is not taking any classes. Oracle is not being taken by any student and classid=2 doesn't have a class description.

1 Give a listing of all the ssns,first names and the class descriptions of all the classes the students are taking. If there are no class _descriptions display 'No description is available yet'. (USE NVL)
A: The information requires that you look at all 3 tables. You are looking for an inner join

2 Give a listing of only the lname and the class_code for students who are taking 'Introduction to C programming'. (Inner join)
A: Even though you are only displaying information that appears in student and student_classes, you have to filter on the class description. This would mean that you need to join all 3 tables.

3 Give a listing of all the class_descriptions and the number of
students enrolled in each class for all students who are older
than the average age where the total number of students for the class
is more than 1 student. Order by the number of students. If there
is no class description replace it with 'Other Classes'
(Note: Take it in steps. First do all those who are older than the average age, then do the group by, then add the having clause and then the order and then combine everything together)

A: Looking at just the 1st part, you want the class_description and the number of students enrolled in each class. This information requires that you use the count function. SInce you want to get a number for each of the classes, you have to do a group by. To do this, you need to use the classes and the student_classes table.

The 2nd part is looking for all who are older than the average age. Since now we are talking about the student table which contains the dob, we have to bring in the student table into the mix as well. To look at those older than the average age, you have to do a subquery

The 3rd part is about the number of students for classes that are more than 1. Since we have to filter on the count function, the having clause would have to be used.

The basic structure of this syntax is
Select … from … where (convert to age) > (subquery) group by .. having … order by

4 Give a listing of all the classes for which no students are
enrolled in (use in or not in clause) (subquery)

A: Give the above dataset, Notice that no one is taking the Oracle class.

5 Give a listing of all the students who are not enrolled in any
classes (Note: Use Exists or not Exists)

A: We are looking for Jill who is not taking any classes

6 create a new table that contains the list of all the students
and class_descriptions. Include In this table the list of all
students who are not enrolled in any classes (display no classes). If there are no class descriptions then display 'no description'
(Use combination of inner join, union and minus)
(Note: minus will deal with the students who are not enrolled in any classes)

A: Notice that we are looking for student names and class descriptions as part of the output. There are multiple parts to this question.

We are looking for students who are taking classes. They may be taking a class for which there is no class description. This would be an inner join just like question 1
Select … from all three tables inner joined

We are also looking to include the students who are not taking any classes such as Jill. The inner join will not pick this up. One way to select Jill is to take all the students and to minus them from all the students who are taking classes

Select … from student
Minus
Select … from inner join between student and student classes

You can then put it all together
Select … from all three tables inner joined (All students taking classes)
union
(Select … from student (students not taking classes)
Minus
Select … from inner join between student and student classes)

Note: Notice the parentheses in order to ensure precedence.

Another part to keep in mind is that the number of columns must be the same for all select statements and they have to match up.

Do each part separately and then start connecting them together.

Note the following as well:
Select name, 'no classes' from student
Will display the name and the words 'no classes' for every row. This hint is what you want to consider for the 2nd part that is being unioned which is to give you a listing of all the students who are not taking classes.

7 repeat question 6 using a combination of inner join, union and not exists
(Note: Not exists will deal with the students who are not enrolled in any classes)

A: Everything is the same as the last question except that instead of using minus, you want to use not exists
Select .. from
Union
Select .. from … not exists (...)

8 create a view. We want to find out which courses are being taken by the different students for all those whose age is greater than the average age. Give a listing of the course descriptions and student names (Inner join)

A: This one is just like the 1st question except that it will be associated with a view
Create view … as select ...

9 We want to find out the courses that each student is not enrolled in.
Give a listing of the course descriptions, and the students (lname) who are not taking that specific course
(Use a cartesian product and a minus)

A: Let's make sure that we know exactly what we want to display. John is not taking Oracle. Jack is not taking class 12 (no description) and Oracle and Jill is not taking any classes.

If we do a cartesian product between student and classes, we get a listing of all the possibilities, which is all the courses that a student can potentially take and then we minus the results from inner join which is a listing of all the classes that the person is taking. This would leave us with all the classes that the person is not taking.
Cartesian product
Minus
Inner join

10) Use the system catalog tables to display the results to find out the following:(Note show me the SQL syntax along with your results) Only a single SQL statement for each question.

A: For all these questions, you want to join the user_constraints and the user_cons_columns tables.
The user_constraints contains a lot of columns including the following which is what we are interested in.
Table_name, contraint_name, r_constraint_name, constraint_type

The user_cons_columns contains
Table_name, constraint_name, column_name

This would mean that if we want to find out what columns make up a constraint, we have to connect both tables. We can connect the table through table_name, constraint_name and r_constraint_name (used for foreign keys)

a) Primary key name and the columns that make up the primary key for student table
A: For primary key, the constraint_type is ‘P’ and table_name is student. You would then need to connect the two tables based on constraint_name

b) Unique key name and the columns that make up the unique key for the student table
A: For unique key, the constraint_type is ‘U’ and table_name is student. You would then need to connect the two tables based on constraint_name

c) Foreign key name, the columns that make up the foreign key and the columns it references in the parent table for student_class table
A: The foreign keys reside in the student_class table and the constraint_type is ‘R’. Matching up the constraint names would give you the column name in the student_class table that makes up the foreign key. Matching up the r_constraint_name against the constraint_name in user_cons_columns would give you the column name in the parent table that the foreign key is connecting to.

d) Name of all the check constraints and their conditions for the student table
A: For this question, you only need to use the user_constraints table. The constraint_type is ‘C’