CSc 460 — Database Design
Spring 2026 (McCann)

http://www2.cs.arizona.edu/classes/cs460/spring26/

Homework #3: SQL Queries

(70 points)

Due Date: March 19th, 2026, at the beginning of class

System Requirement: You MUST use our local Oracle DBMS installation for this assignment.

Overview: This assignment is simply meant to give you the opportunity to get some practice with the formulation of SQL queries. The DBMS we'll be using is the Oracle Database 11g Enterprise Edition, a vintage version of the database system first created in 1978. We won't be using many Oracle–specific features in this assignment; the goal is to give you some practice formulating and testing basic SQL queries.

Software: Oracle 11g runs on a machine in our department named "aloe", but it is accessed from lectura. You each have an Oracle account. The username is your lectura username, and your password is the letter 'a' concatenated to the first 4 digits of your class grade sheet identifier (e.g., `a1234` if your ID is 123456).

To access Oracle's command–line querying program, SQL*Plus, start by SSHing to lectura. Then, run a script named `sqlpl` with a command–line argument of this form:

```
sqlpl username@oracle.aloe
```

where 'username' is your NetID. You'll be prompted for your password, and then you'll get the SQL*Plus prompt (`SQL>`). (You can change your password, with the 'password' command, but there's no compelling reason to, as you won't be storing any sensitive info.)

I've set the tables of the aquarium database (from Homework #2) and the Supplier–Part–Project database to be accessible by you. In addition, you can create your own tables to play with. I strongly suggest that you attempt to access Oracle ASAP to verify that your Oracle access was set up correctly. First, connect to Oracle as shown above. At the `SQL>` prompt, type this query: `select * from mccann.species;` (don't forget the "mccann." and the semicolon!). If you see the content of the species table, all should be well.

Assignment: Basically, the exercise is to redo most of the queries you answered in Homework #2 (with a few substitutions and additions) using SQL. I've created tables that contain the same information as the LEAP aquarium database. Here is the schema again, with slight changes to some of the field names.

```
Species (sno, sname, sfood)
Tank (tno, tname, tcolor, tvolume)
Fish (fno, fname, fcolor, fweight, tno, sno)
Event (eno, fno, edate, enote)
```

Field types are easy to determine from the field content. `edate` is just a 5–character string, not an Oracle `date` type.

Using Oracle and the aquarium database, write SQL queries that answer the following questions. (Remember, there are changes from Homework #2, so don't just redo that list, use the following one instead.) If you find any questions that you can't answer, explain why. (But be aware that we believe all of them can be answered.)

Point distribution: Queries 1 & 3: 2 points each. Q2&4: 3 each. Q5,6,8,12: 5 each. Q7,9,10,11,13: 8 each.

(Continued . . .)

1. What are the names of all of the red fish?
2. What are the colors of all of the tanks named "lagoon"?
3. What is the Cartesian Product of the sname field from Species with the tname field from Tank? List each (sname,tname) pair only once.
4. What are the colors of the sharks (in alphabetical order)?
5. What is the name of the heaviest fish?
6. What are the names of the fish that are sharks and live in cesspools?
7. The database contains names of species, tanks, and fish. Display a result containing all of these names.
8. What are the names of species found in puddles?
9. What are the names of species that are found in the same tank with a shark? List each species name only once in the result.
10. What are the names of the fish that have been born and are swimming?
11. What are the names of the fish that have been born but are NOT swimming?
12. What are the colors of the fish and the average weight of the fish of each color? Include in your result only those colors (with the associated average weights, of course) that have an average group weight under 40, and list the results in descending order by weight.
13. What are the names of the species that eat herring that have a representative in all green tanks?

Note that, unlike in LEAP, you do not need to (and should not!) create any temporary relations to write any of these queries in SQL. See also the Other Requirements and Hints section, below, for additional thoughts.

Hand In: Using `turnin`, submit the following to the `cs460h3` folder:

- Using `script`, create a `typescript` file that includes the query results that Oracle produces when it runs your queries. Produce the answers in the same order as the queries are listed above.
- Your SQL queries, as one combined, or as several separate, `.sql` files.

Want to Learn More About Oracle?

- Oracle 11g documentation (and there's a **lot** of it!) is available on-line:
  `http://docs.oracle.com/cd/E11882_01/`
  For this assignment, the "SQL Language Reference" is likely to be the most useful (it's in the "Supporting Documentation" section). Hopefully, you won't need to reference it at all.
- Oracle offers a free 21c Express Edition. I've never downloaded it, but if you want to play with it, visit:
  `https://www.oracle.com/database/technologies/appdev/xe.html`

Other Requirements and Hints:

- You can easily capture Oracle's output to a file by running `sqlpl` within the `script` command. Another option is to use SQL*Plus's `spool` command.
- For set difference, remember that Oracle uses the MINUS operator instead of EXCEPT.
- In Oracle, executing a file of SQL commands from within SQL*Plus uses the same basic syntax as LEAP: `@ filename`. Example: `@ query01.sql` SQL*Plus looks for `filename` in your current directory.
- Outside of this assignment, if you *really* need to create a temporary table to hold the result of a query, you *can* do that in Oracle ... but you shouldn't, for both philosophical and performance reasons. You could create a table in advance to hold results, and then use the `insert into <relation> <select stmt>;` variation of `insert`. However, in this assignment, doing this is not necessary; you can construct all of these queries without manually creating and populating any additional tables. (If you find yourself wanting to do that, you're probably thinking procedurally, and SQL isn't relational algebra!) Note that other DBMSes support temporary tables.
- Please remember that a correct answer is a query that produces the correct result *in a logically correct way*! Write queries that will work even if the relations' content changes.
- Seems like I'm forgetting something ... Oh, right: Start early!