



PURDUE POLYTECHNIC INSTITUTE
Department of Computer and Information Technology

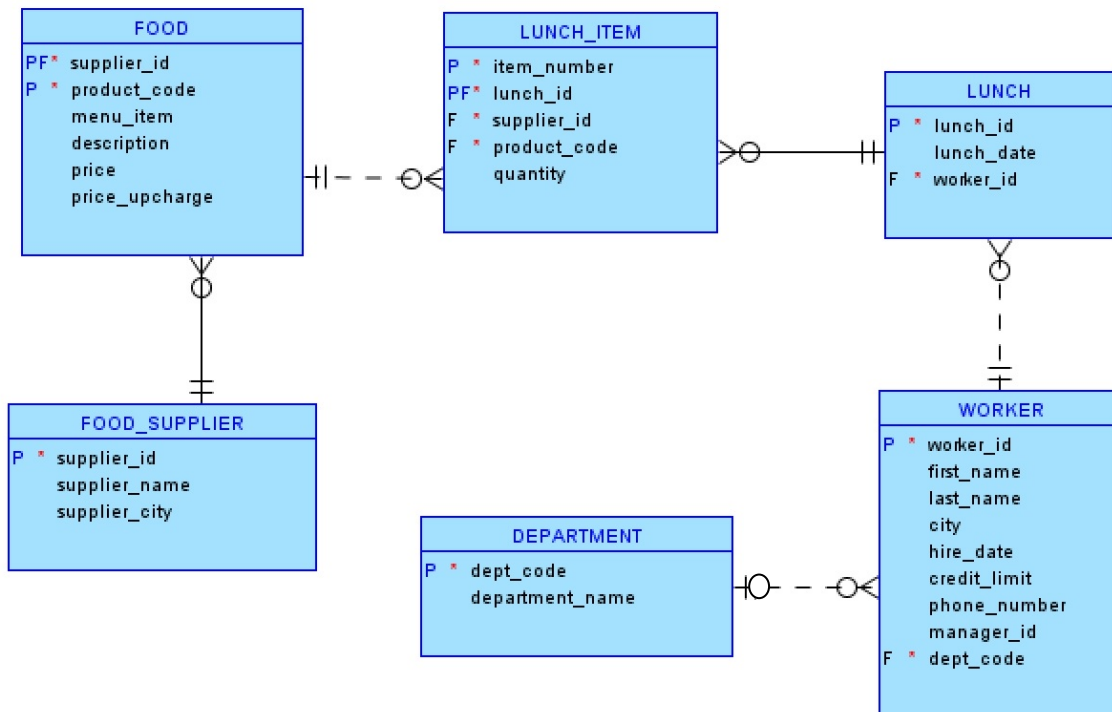
CNIT 27200: Lab #6

25 points

Due Date

- Part A is due via Blackboard on Wednesday, March 25th, by 11:59pm. 10 pts.
- Part B is due via Blackboard on Monday, March 30th, by 11:59 p.m. 15 pts.

Lunches DB ERD (for Part A)



PART A (10 points)

Objectives:

- Be able to perform outer joins, union, minus and intersect

Submission Instructions:

This is an individual assignment, not collaborative or cooperative with other students.

Your PART A answer template should follow a standard format, to be compiled in a .sql file with your name in the file as **studentnameLab6A.sql**. This is for saving your FINAL SQL statements and the output. Fill in the header information and insert the question number to separate the responses – we will be using comments to store all our non-statement content such as question numbers, results, etc.

You can comment in SQL by beginning the line with a --

Alternatively, for multi-line comments, surround the block by /* and */.

The following is a template:

```

/* <Begin Comment>
Your Name
CNIT 27200 Spring 2020
Lab Time:
Duration:
*****

Question 1
<End comment> */
Select * from ...

/* Results:
    <paste table output>

    Explanation: In this query...
*/
/*
*****



Question 2 */

```

etc.

Questions (Use the Lunches DB – See ERD above for help):

1. Use the FOOD_SUPPLIER and FOOD tables from the Lunches DB for the following questions.
 - A) List the supplier ID, supplier name, food description and price for all food supplied.
 - ☐ Use an INNER JOIN
 - ☐ Order by the supplier name
 - ☐ 33 rows selected.
 - B) List the supplier ID, supplier name, food description, and price for all food supplied. Include the suppliers who have yet to supply food to the organization (the food description and price will be null in result set).
 - ☐ Use a LEFT JOIN
 - ☐ 37 rows selected.
 - C) List the supplier ID, supplier name, food description, and price for only suppliers that have NOT supplied food to the organization.
 - ☐ Use a LEFT JOIN and include a WHERE clause to find NULL Supplier IDs from the FOOD table.
 - ☐ 4 rows selected. (These suppliers are the “parent only” records with no “children” in the food table).
2. Use the FOOD and LUNCH_ITEM tables from the Lunches DB for the following questions.
 - A) In the SELECT clause, list the product code, food description, total purchase (Hint: SUM the multiplication of price by quantity), and count per each lunch item.
 - B) If the product code has never been a lunch item, still list the product code and description (Use a LEFT JOIN)
 - C) The relationship includes a composite PK in the ON clause
 - D) Because there are aggregated functions in the SELECT clause, remember that you will need to use a GROUP BY clause
 - E) Look at the result set and you will find 2 products without a subtotaled purchase. These are the product codes that have never been a purchased lunch item.
 - F) 32 rows selected
 - G) Change the LEFT JOIN to an INNER JOIN and rerun it.
 - ☐ Review the difference between an Inner Join and Left Join.
 - ☐ The 2 product codes that have never been purchased are not in the Inner Join result set. For those two items, there are no matching values in the LUNCH_ITEM table.
 - ☐ 30 rows selected

3. Use the DEPARTMENT and WORKER tables from the Lunches DB for the following questions.
- A) List the worker id, city, department code and department name for workers in the database.
- ☐ Set the linesize to 200 for readability.
 - ☐ Use an INNER JOIN
 - ☐ Order by the department code
 - ☐ 25 rows selected.
- B) List the worker id, city, department code and department name for all workers in the database. Include all workers, even if they have not been assigned to a department yet.
- ☐ Use a RIGHT JOIN
 - ☐ Order by the department code
 - ☐ 31 rows selected.
 - ☐ *The database is set up in such a way that the department can be null. This is based on cardinality business rules in the logical entity relationship diagram.* 
- C) List the worker id, city, department code and department name for workers in the database. Include all departments, even if they have no employees.
- ☐ Use a LEFT JOIN
 - ☐ Order by the department code
 - ☐ 29 rows selected
 - ☐ *The database is set up in such a way that departments can be in the database without workers. This is based on cardinality business rules in the logical entity relationship diagram.* 

For Questions 4 through 6 use UNION, INTERSECT or MINUS operations as needed.

4. Use a UNION operation to list the names of the food suppliers and the supplier cities, as well as the concatenated first and last names of the workers and cities into one SQL statement
- ☐ The result set will have two columns (a column for all of the names and a column for all of the cities).
 - ☐ Review concatenation (||) if needed from the SQL Basics slides.
 - ☐ Limit the results to only food suppliers and workers NOT from the cities of **Skokie, Oak Brook, and Chicago**.
 - ☐ To complete this step, you will add a WHERE clause to both of the queries in the UNION.
 - ☐ One will filter the supplier cities, and the other will filter the worker cities.
 - ☐ Sort by the supplier name if you used the food supplier table in the first query, otherwise, sort by the worker names.
 - ☐ 19 rows selected

5. Use the INTERSECT operation to find any workers that are living in the same city as any of the food suppliers. 5 rows selected.
6. Use the MINUS operation to list the cities of the suppliers where no workers live (i.e. city where there are suppliers living, but no workers).
 - A) 2 rows selected.
 - B) Try testing the results.
 - ☐ First run a query of supplier cities
 - ☐ Second run a query of work cities
 - ☐ Compare the two lists.
 - ☐ If you remove the worker cities from the supplier cities, there will be two cities left.

PART B (15 points)

Objectives:

- Understand and know the syntax for outer joins
- Understand and construct queries with UNION and UNION ALL
- Understand and construct queries with INTERSECT
- Understand and construct queries with MINUS

Submitting the Lab File:

Use same answer template from previous labs. Check the formatting of your **studentnameLab6B.sql** file. If the outputs are wrapping around, reduce the margins and use font size when necessary. To line up the output columns, change the font of the whole document to Courier New. Please use a simple text editor instead of MS Word due to formatting. **As a reminder, this is an individual assignment, not collaborative or cooperative with other students.**

Questions are worth a total of 15 pts (all questions are worth 1.5pts each).

Use the Eagle DB for Part B. If you have not already, print out the Eagle ERD for a visual of the entities/attributes/relationships in the database.

Use LEFT or RIGHT outer join operations to answer questions 1 through 3. Read the question, then decide if you need to use a left outer join or a right outer join to complete the task.

1. Using an outer join, list the Supplier ID, contact name, and cell phone of any supplier currently not supplying Eagle with parts (list ONLY these suppliers).
 - a. 5 rows selected
 - b. Explain how to complete this task with a nested query instead of the left outer join.
2. Create a query that will find the total unit cost of supplied parts per supplier. Using an outer join, list the Supplier's Company Name, Contact Name, Phone, and a total unit cost supplied (unitcost).
 - a. Include all suppliers whether or not they supply products
 - b. Label the totaled supplied part unitcost as TOTAL_UC.
 - c. 34 rows selected.
 - d. HINT: Need a GROUP BY in this because of the aggregated function
 - e. Explain what your query is grouping and how you did it.

3. Using an outer join, list the part number, category id, category name, weight, and stock price for all parts with a stock price less than **5** and a weight greater than the **average** weight of all parts in the **CBL** category.
 - a. If a part number does not have a category assigned, still list the part number and weight.
 - b. You will also need to use a nested subquery.
 - c. 9 rows selected
 - d. What is the average weight of all parts in the **CBL** category? – Add the SQL statement and result to your explanation. For your explanation: What is the average weight for all parts in the **CBL** category in the inventory part table? For readability, please round it to 2 decimal places. Add the SQL statement and result to your explanation. Also explain the difference between a left join and a right join.

Use the UNION, INTERSECT, or MINUS operations to answer questions 4 through 10.

4. Use the MINUS operation to show the Customer ID of any customers that currently have no orders. 15 rows selected.
 - a. Explain how the MINUS operation works. Does it matter which query is first? Test it by switching the queries. Explain your findings.
5. Use the INTERSECT operation to find out if any of the Eagle Electronics 's employees are also receiving shipments from the same city.
 - a. Compare the employee first and last name as one column called **MATCHING** to the name on the shipment (use shipname).
 - b. Also compare the employee's city to the shipment city.
 - c. 1 row selected.
 - d. Explain what happens when you add more attributes to both SELECT clauses in an intersect between two queries.
6. Use the INTERSECT operation to show the employee last name, hire date, birth date, and Month of the hire date for any employee hired the same month as his/her birthday month (EX. Hired in January 2012, birthdate in January 1984).
 - a. No employee would be hired in the year of his or her birth, so only use the Month. (Use To_Char)
 - b. Call the Month **HIRE_BIRTHDAY**
 - c. HINT: you are intersecting within the same table
 - d. 7 rows selected.
 - e. Explain how the query is comparing just the month, and how you can intersect in the same table.

7. Use the UNION operation to combine a count of **BRK** part numbers that have been ordered and a count of **BRK** part numbers that are in process routing. List the part number and count in each SELECT clause. Use substr in both WHERE clauses to isolate the part number with **BRK** in the first 3 positions of the field. 21 rows selected.
 - a. Explain how the UNION operation works

8. The SHIPNAME field on the SHIPMENT record is made up of "Firstname Lastname" (note space between). These shipment names should match the first and last names on the customer's record in the CUSTOMER table. Use the MINUS operation to list shipment record names that do not match any names in the CUSTOMER table.
 - a. 54 rows selected.
 - b. How many do match? Use Intersect to find out.
 - c. Explain the difference between an inner join and an outer join. How are they similar?

9. Use the MINUS operation to show the Part Numbers from inventory parts that are not Part Numbers in the CUSTORDERLINE table.
 - a. Explain what this query represents to the user.
 - b. What other way can this be written?
 - c. 17 rows selected.

10. List the supplier id, contact name, unit cost for the supplier id(s) that has the lowest unit cost and supplier id(s) with the highest unit cost.
 - a. Use a subquery to UNION the lowest to the highest unit cost.
 - b. Then use an IN statement to report the selected result information.
 - c. Use an inner join to list the supplier id, supplier contact name, and unit cost.
 - d. 2 rows selected
 - e. Explain when you use **IN** versus **=** for a nested query.