**IVY TECH COMMUNITY COLLEGE**

**DBMS130 – M04 Hands-On Lab Assignment**

**Querying Multiple Tables Using Joins and Using Set Operators**

Use the Small Company ERD to complete the steps listed below.

* Read each of the questions listed in the left column of the table below.
* Write the answer to your question and/or the required SQL statement in the corresponding cell in the right column that answers the question. (Test your statement to confirm the results before submitting.)
* Save this document once you have added your answers and uploaded it as part of your submission.

# **Using Joins (30 points)**

|  |  |
| --- | --- |
| **Joins** | |
| 1. Write a query using a join that displays employees’ IDs, names, and department names. Sort your results by department and employee last name. | SELECT e.employee\_id, e.employee\_name, d.department\_name  FROM employees e, departments d  WHERE e.department\_id = d.department\_id  ORDER BY d.department\_name, e.employee\_name; |
| 1. Write a query using a join that displays employees’ IDs, names, and job names. Sort your results by department and employee last name. | SELECT e.employee\_id, e.employee\_name, j.job\_name  FROM employees e  JOIN jobs j ON e.job\_id = j.job\_id  ORDER BY e.department\_id, e.employee\_name; |
| 1. Write a query using a join that displays employees’ IDs, names, job and department names. Sort your results by department, job, and employee last name. | SELECT e.employee\_id, e.employee\_name, j.job\_name, d.department\_name  FROM employees e  JOIN jobs j ON e.job\_id = j.job\_id  JOIN departments d ON e.department\_id = d.department\_id  ORDER BY d.department\_name, j.job\_name, e.employee\_name; |
| 1. Write a query using a self-join to display employees’ IDs and names along with their manager’s ID and name. Sort your results by the managers’ last names and the employees’ last names. | SELECT employees.employee\_id, employees.employee\_name,  managers.employee\_id AS manager\_id, managers.employee\_name AS manager\_name  FROM employees  LEFT JOIN employees AS managers ON employees.manager\_id = managers.employee\_id  ORDER BY managers.employee\_name, employees.employee\_name; |
| 1. Write a query using a join that displays customers’ IDs, names, order numbers, and order dates. Sort your results by order date (descending) and customers’ names. | SELECT customer\_id, customer\_name, orders.order\_number, orders.order\_date  FROM customers  JOIN orders ON customer\_id = orders.customer\_id  ORDER BY orders.order\_date DESC, customer\_name; |
| 1. Write a query using a join that displays customer customers’ IDs, names, order numbers, order dates, and book titles associated with the order. Sort your results by order date (descending), customers’ last names, and book title. | SELECT customers.customer\_id, customers.customer\_name,  orders.order\_number, orders.order\_date,  books.book\_title  FROM customers  JOIN orders ON customers.customer\_id = orders.customer\_id  JOIN order\_details ON orders.order\_number = order\_details.order\_number  JOIN books ON order\_details.book\_id = books.book\_id  ORDER BY orders.order\_date DESC, customers.customer\_name, books.book\_title; |
| 1. The company would like to send coupons to customers that like to read fiction. Write a query to display a list of customers that have purchased book titles that are categorized as fiction (BOOK\_CATEGORY\_CD = F). Your results should include the customers’ name and email. It should also include the order number, date, and book title for their previous purchase of fiction book. Sort your results by the customers’ last name, order date, and book title. | SELECT customers.customer\_name, customers.email,  orders.order\_number, orders.order\_date,  books.book\_title  FROM customers  JOIN orders ON customers.customer\_id = orders.customer\_id  JOIN order\_details ON orders.order\_number = order\_details.order\_number  JOIN books ON order\_details.book\_id = books.book\_id  WHERE books.book\_category\_cd = 'F'  ORDER BY customers.customer\_name, orders.order\_date DESC, books.book\_title; |
| 1. The company would like to identify all of the customers that have never placed an order. Write a query using a join that provides this information. Your results should include all the customer details and the order number column. Sort your results by the customers’ last name and then first name. | SELECT customer\_id, customer\_name, email, order\_number  FROM customers  LEFT JOIN orders ON customers.customer\_id = orders.customer\_id  WHERE order\_number IS NULL  ORDER BY customer\_name, first\_name; |
| 1. The company is performing an analysis of their inventory. They are considering purging books that are not popular with their customers. To do this they need a list of books that have never been purchased. Write a query using a join that provides this information. Your results should include all the book details and the order number column. Sort your results by the book title. | SELECT book\_id, book\_title, author, genre, publish\_date, order\_number  FROM books  LEFT JOIN order\_details ON books.book\_id = order\_details.book\_id  LEFT JOIN orders ON order\_details.order\_number = orders.order\_number  WHERE order\_number IS NULL  ORDER BY book\_title; |

# Using Set Operators (30 points)

|  |  |
| --- | --- |
| **Set Operators** | |
| 1. The budget at the company is tight. The president is considering making cuts. He would like a report that lists people who meet the following criterion:    * salary > 12500 and manager\_id is not null    * hire\_date > to\_date('1996-01-01','yyyy-mm-dd')   Using the Boolean operators AND and OR produced an inaccurate report. Write a query that uses the set operator INTERSECT to produce an accurate report. | SELECT employee\_id, employee\_name, salary, manager\_id, hire\_date  FROM employees  WHERE salary > 12500  AND manager\_id IS NOT NULL  INTERSECT  SELECT employee\_id, employee\_name, salary, manager\_id, hire\_date  FROM employees  WHERE hire\_date > TO\_DATE('1996-01-01', 'YYYY-MM-DD'); |
| 1. Describe why the using Boolean operators produces different results than the set operator INTERSECT in this scenario. | Using Boolean operators combines conditions within a single query, while the INTERSECT set operator combines the results of separate queries to find the intersection of their result sets. |
| 1. The President of the company is also a customer. Write a SQL statement that adds his customer record to the database. Give him CUSTOMER\_ID 999. His address is 1999 Green Mile Lane in Los Angeles, California 90210. |  |
| 1. The company had a data breach and needs to contact everyone in their database – employees and customers. Write a SQL statement that uses the UNION set operator to create a list of employees and customers. Your results should include the employees’ and customers’ first names, last names, and email addresses and be sorted by last name. |  |
| 1. Write a SQL statement that uses the UNION ALL set operator to create a list of employees and customers. Your results should include the employees’ and customers’ first names, last names, and email addresses and be sorted by last name. |  |
| 1. Write a SQL statement that uses the INTERSECT set operator to create a list of employees and customers. Your results should include the employees’ and customers’ first names, last names, and email addresses and be sorted by last name. |  |
| 1. Write a SQL statement that uses the MINUS set operator to create a list of employees and customers. Your results should include the employees’ and customers’ first names, last names, and email addresses and be sorted by last name. |  |
| 1. Explain the differences in your results to questions #d-g. |  |
| 1. Which SQL statement should you use to create the list needed to contact everyone in the database about the data breach? |  |