# Retrieving data from multiple tables

## Displaying Data from Multiple Tables

UMKC
School of Medicine

# Retrieving data from multiple tables

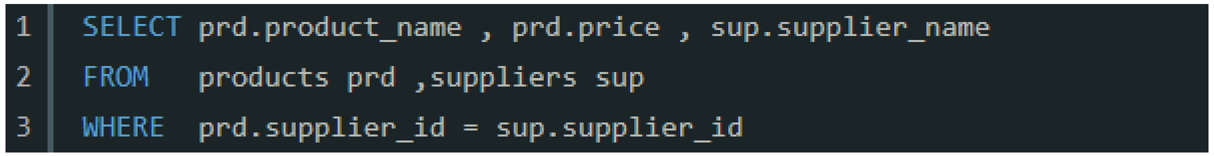
- Objectives
  - The importance of aliases in joining tables
  - Handling unmatched records: inner, left, right, and outer join
  - Join a table to itself by using a self-join

# SQL Aliases

- Aliases are the temporary names given to table or column for the purpose of a particular SQL query. It is used when name of column or table is used other than their original names, but the modified name is only temporary.
    - Aliases are created to make table or column names more readable.
    - The renaming is just a temporary change and table name does not change in the original database.
    - Aliases are useful when table or column names are big or not very readable.
    - These are preferred when there are more than one table involved in a query.

- table alias.

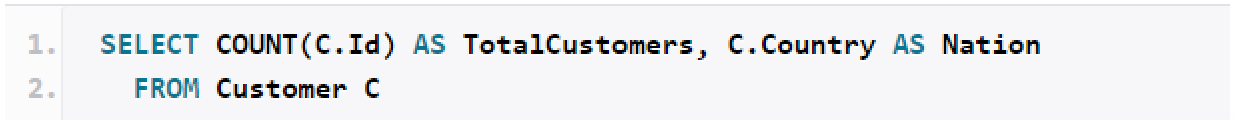
```
SELECT column1, column2....
FROM table_name AS alias_name
WHERE [condition];
```



```
SELECT prd.product_name , prd.price , sup.supplier_name
FROM   products prd ,suppliers sup
WHERE  prd.supplier_id = sup.supplier_id
```

- Column alias.

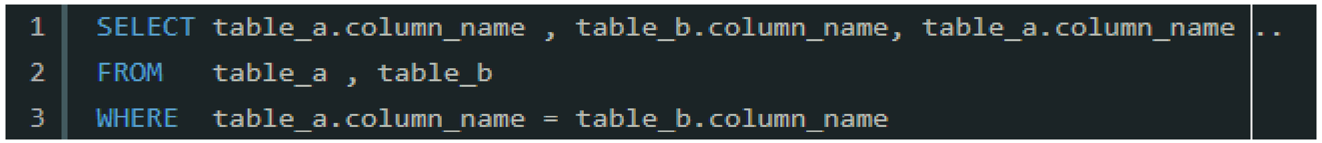
```
SELECT column_name AS alias_name
FROM table_name
WHERE [condition];
```



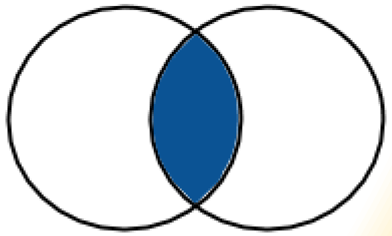
```
SELECT COUNT(C.Id) AS TotalCustomers, C.Country AS Nation
  FROM Customer C
```

# SQL JOINS - Inner JOIN

- Inner JOIN – Joining data items from tables, based on values common to both tables.
  - Multiple tables (two or more tables) can be linked only if they have common values (in this case, supplier number) or a logical connection of some kind.
  - Relating between two tables requires you to determine the join condition. In the example shown above, the join condition was based on the equality operator (=).
  - In the Oracle SELECT clause, precede the column name with the table name for clarity.
  - When a column is common to both tables, it must be prefixed with the table name.
  - In the Oracle FROM clause, you need to specify the tables from which you would like to retrieve the data. These tables are specified with comma (,) between them.
  - After the WHERE keyword, specify the join condition.
  - To determine the relation between table_a and table_b tables – values in the column_name column on both tables must be equal. This type of relation is referred as an Equi Join.
  - Equi joins are also called Simple Joins or Inner Joins.
  - Frequently, this relation involves primary key and foreign key complements.



```sql
SELECT table_a.column_name , table_b.column_name, table_a.column_name ..
FROM   table_a , table_b
WHERE  table_a.column_name = table_b.column_name
```

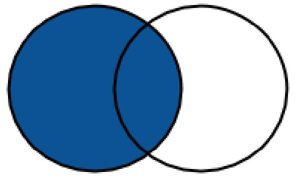


# SQL JOINS - Outer JOIN

- Outer JOIN – Joining data items from tables, based on values common to both tables, while displaying all data from one table regardless of if there is a match on the second table.
- Left [OUTER] Join
  - A LEFT [OUTER] JOIN returns all valid rows from the table on the left side of the JOIN keyword, along with the values from the table on the right side, or NULLs if a matching row doesn't exist.

```
SELECT d.department_name,
       e.employee_name
FROM   departments d
       LEFT OUTER JOIN employees e ON d.department_id = e.department_id
WHERE  d.department_id >= 30
ORDER BY d.department_name, e.employee_name;

DEPARTMENT_NAM EMPLOYEE_N
-------------- ----------
OPERATIONS
SALES          ALLEN
SALES          BLAKE
SALES          JAMES
SALES          MARTIN
SALES          TURNER
SALES          WARD
```

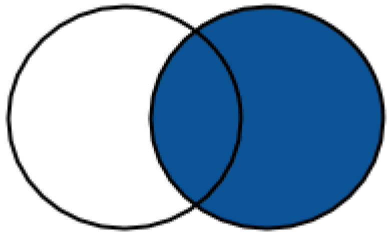


# SQL JOINS - Outer JOIN

- Outer JOIN – Joining data items from tables, based on values common to both tables, while displaying all data from one table regardless of if there is a match on the second table.
- Right [OUTER] Join
  - The RIGHT [OUTER] JOIN is the opposite of the LEFT [OUTER] JOIN. It returns all valid rows from the table on the right side of the JOIN keyword, along with the values from the table on the left side, or NULLs if a matching row doesn't exist. All points raised in the previous section apply here also.

```
SELECT d.department_name,
       e.employee_name
FROM   employees e
       RIGHT OUTER JOIN departments d ON e.department_id = d.department_id
WHERE  d.department_id >= 30
ORDER BY d.department_name, e.employee_name;

DEPARTMENT_NAM EMPLOYEE_N
-------------- ----------
OPERATIONS
SALES          ALLEN
SALES          BLAKE
SALES          JAMES
SALES          MARTIN
SALES          TURNER
SALES          WARD
```



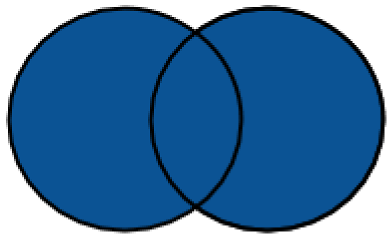
# SQL JOINS - Outer JOIN

- Outer JOIN – Joining data items from tables, based on values common to both tables, while displaying all data from one table regardless of if there is a match on the second table.
- Full [OUTER] Join
  - A FULL [OUTER] JOIN combines all the rows from the tables on the left and right sides of the join. If there is a conventional match it is made. If either side has missing data, it is replaced by NULLs, rather than throwing the row away.
  - Add another employee who is not assigned to a department.

```
INSERT INTO employees VALUES (8888,'JONES','DBA',null,to_date('02-1-1982','dd-mm-yyyy'),1300,NULL,NULL);
```

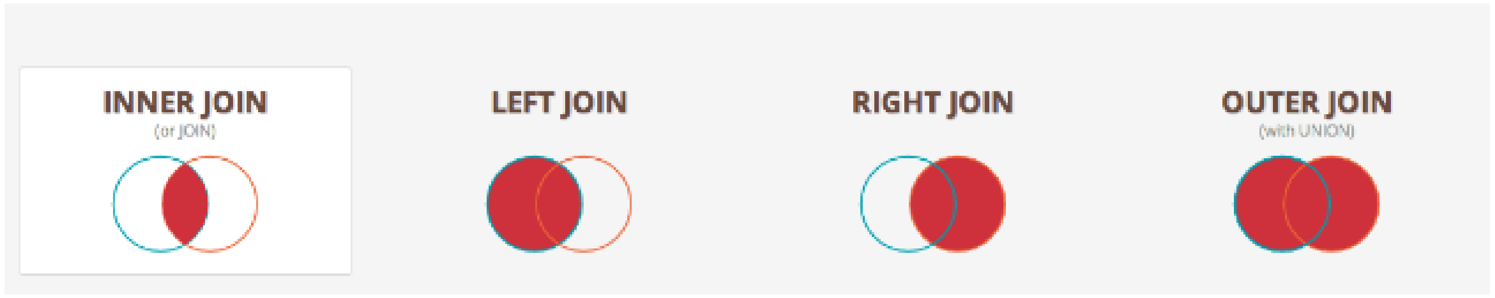
```
SELECT d.department_name,
       e.employee_name
FROM   employees e
       FULL OUTER JOIN departments d ON e.department_id = d.department_id
ORDER BY d.department_name, e.employee_name;

DEPARTMENT_NAM EMPLOYEE_N
-------------- ----------
ACCOUNTING     CLARK
ACCOUNTING     KING
ACCOUNTING     MILLER
OPERATIONS
RESEARCH       ADAMS
RESEARCH       FORD
RESEARCH       JONES
RESEARCH       SCOTT
RESEARCH       SMITH
SALES          ALLEN
SALES          BLAKE
SALES          JAMES
SALES          MARTIN
SALES          TURNER
SALES          WARD
               JONES
```



# SQL JOINS - Outer JOIN

- An outer join extends the result of a simple join. An outer join returns all rows that satisfy the join condition and also returns some or all of those rows from one table for which no rows from the other satisfy the join condition.
  - To write a query that performs an outer join of tables A and B and returns all rows from A (a left outer join), use the LEFT [OUTER] JOIN syntax in the FROM clause, or apply the outer join operator (+) to all columns of B in the join condition in the WHERE clause. For all rows in A that have no matching rows in B, Database returns null for any select list expressions containing columns of B.
  - To write a query that performs an outer join of tables A and B and returns all rows from B (a right outer join), use the RIGHT [OUTER] JOIN syntax in the FROM clause, or apply the outer join operator (+) to all columns of A in the join condition in the WHERE clause. For all rows in B that have no matching rows in A, Database returns null for any select list expressions containing columns of A.
  - To write a query that performs an outer join and returns all rows from A and B, extended with nulls if they do not satisfy the join condition (a full outer join), use the FULL [OUTER] JOIN syntax in the FROM clause.



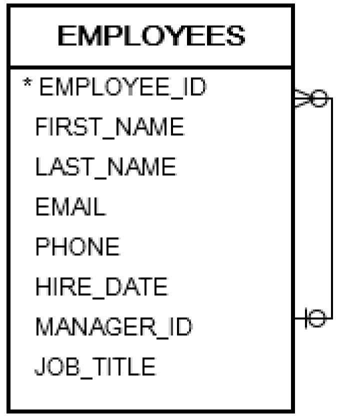
# SQL JOINS - Self JOIN

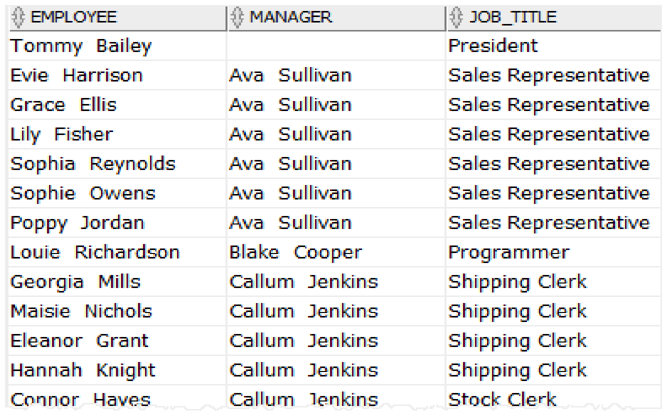
- Self JOIN – A self join is a join of a table to itself. This table appears twice in the FROM clause and is followed by table aliases that qualify column names in the join condition. To perform a self join, Database combines and returns rows of the table that satisfy the join condition.
  - A self join is useful for comparing rows within a table or querying hierarchical data.
  - A self join uses other joins such as inner join and left join. In addition, it uses the table alias to assign the table different names in the same query.

**SELECT column_list FROM P P1 INNER JOIN P P2 ON join_predicate;**

- Self Join example
  - See the following employees table on right.
  - The employees table stores personal information such as id, name, job title. In addition, it has the manager_id column that stores the reporting lines between employees.
  - The President of the company, who does not report to anyone, has a NULL value in the manager_id column. Other employees, who have a manager, have a numeric value in the manager_id column, which indicates the id of the manager.
  - To retrieve the employee and manager data from the employees table, you use a self join as shown in the following statement:

```
SELECT
   (e.first_name || ' ' || e.last_name) employee,
   (m.first_name || ' ' || m.last_name) manager,
   e.job_title
FROM
   employees e
LEFT JOIN employees m ON
   m.employee_id = e.manager_id
ORDER BY
   manager;
```





| EMPLOYEE | MANAGER | JOB_TITLE |
|---|---|---|
| Tommy Bailey | | President |
| Evie Harrison | Ava Sullivan | Sales Representative |
| Grace Ellis | Ava Sullivan | Sales Representative |
| Lily Fisher | Ava Sullivan | Sales Representative |
| Sophia Reynolds | Ava Sullivan | Sales Representative |
| Sophie Owens | Ava Sullivan | Sales Representative |
| Poppy Jordan | Ava Sullivan | Sales Representative |
| Louie Richardson | Blake Cooper | Programmer |
| Georgia Mills | Callum Jenkins | Shipping Clerk |
| Maisie Nichols | Callum Jenkins | Shipping Clerk |
| Eleanor Grant | Callum Jenkins | Shipping Clerk |
| Hannah Knight | Callum Jenkins | Shipping Clerk |
| Connor Hayes | Callum Jenkins | Stock Clerk |

- This query references to the employees table twice: one as e (for employee) and another as m (for manager). The join predicate matches employees and managers using the employee_id and manager_id columns. The picture above picture shows the result.

## Retrieving data from multiple tables

- Questions