



Database Systems

Chapter 4: Structured Query Language (SQL)

session 3

SubQuery and Multiple Table joins



Outline

1

SubQuery

2

Multiple Table joins



SubQuery

- ❑ Subquery is a **query inside another query**
 - A subquery can be nested inside a SELECT, INSERT, UPDATE or DELETE statement or inside another subquery.
 - A subquery is usually added within the WHERE clause of another SQL SELECT statement.
- ❑ In a subquery SELECT statement, can not include:
 - the keyword DISTINCT in the SELECT phrase,
 - an ORDER BY phrase.



Subquery

□ Subqueries can be:

- Noncorrelated subqueries (Self-contained subqueries)
 - have no dependency on outer query
 - execute once for the entire outer query
- Correlated subqueries
 - depend on values from outer query
 - execute once for each row returned by the outer query
 - Can use the EXISTS operator



Subquery

- ❑ A subquery is must be enclosed within **parentheses**
- ❑ Subquery can return:
 - One single value (Scalar Subquery) - One column and one row
 - A list of values (Multi-Valued Subquery) - One column and multiple rows
 - A virtual table - Multicolumn, multirow set of values
 - No value - Output of the outer query might result in an error or a null empty set



Subquery

- ❑ The most common type of subquery uses an inner SELECT subquery on the right side of a WHERE comparison expression (WHERE subqueries)
- ❑ A subquery can be used within a SQL data manipulation language (DML) statement such as INSERT, UPDATE, or DELETE



WHERE Subqueries

❑ Syntax

```
SELECT column_list  
FROM table  
WHERE expression OPERATOR  
                (SELECT column_list  
                 FROM table)
```

- ❑ The subquery (inner query) executes once before the main query.
- ❑ The result of the subquery is used by the main query (outer query).



The contents of Employees table

	employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	manager_id	department_id
1	100	Steven	King	steven.king@sqltutorial.org	515.123.4567	1987-06-17	4	24000	NULL	9
2	101	Neena	Kochhar	neena.kochhar@sqltutorial.org	515.123.4568	1989-09-21	5	17000	100	9
3	102	Lex	De Haan	lex.de haan@sqltutorial.org	515.123.4569	1993-01-13	5	17000	100	9
4	103	Alexander	Hunold	alexander@sqltutorial.org	590.423.4567	1990-01-03	9	9000	102	6
5	104	Bruce	Emst	bruce.emst@sqltutorial.org	590.423.4568	1991-05-21	9	6000	103	6
6	107	Diana	Lorentz	diana.lorentz@sqltutorial.org	590.423.5567	1999-02-07	9	4200	103	6
7	108	Nancy	Greenberg	nancy.greenberg@sqltutorial.org	515.124.4569	1994-08-17	7	12000	101	10
8	109	Daniel	Faviet	daniel.faviet@sqltutorial.org	515.124.4169	1994-08-16	6	9000	108	10
9	113	Luis	Popp	luis.popp@sqltutorial.org	515.124.4567	1999-12-07	6	6900	108	10
10	114	Den	Raphaely	den.raphaely@sqltutorial.org	515.127.4561	1994-12-07	14	11000	100	3
11	120	Matthew	Weiss	matthew.weiss@sqltutorial.org	650.123.1234	1996-07-18	19	8000	100	5
12	123	Shanta	Vollman	shanta.vollman@sqltutorial.org	650.123.4234	1997-10-10	19	6500	100	5
13	176	Jonathon	Taylor	jonathon.taylor@sqltutorial.org	NULL	1998-03-24	16	8600	100	8
14	177	Jack	Livingston	jack.livingston@sqltutorial.org	NULL	1998-04-23	16	8400	100	8
15	192	Sarah	Bell	sarah.bell@sqltutorial.org	650.501.1876	1996-02-04	17	4000	123	5
16	193	Britney	Everett	britney.everett@sqltutorial.org	650.501.2876	1997-03-03	17	3900	123	5
17	200	Jennifer	Whalen	jennifer.whalen@sqltutorial.org	515.123.4444	1987-09-17	3	4400	101	1
18	201	Michael	Hartstein	michael.hartstein@sqltutorial.org	515.123.5555	1996-02-17	10	13000	100	2
19	202	Pat	Fay	pat.fay@sqltutorial.org	603.123.6666	1997-08-17	11	6000	201	2
20	203	Susan	Mavris	susan.mavris@sqltutorial.org	515.123.7777	1994-06-07	8	6500	101	4
21	205	Shelley	Higgins	shelley.higgins@sqltutorial.org	515.123.8080	1994-06-07	2	12000	101	11
22	206	William	Gietz	william.gietz@sqltutorial.org	515.123.8181	1994-06-07	1	8300	205	11



The contents of Departments table

	department_id	department_name	location_id
1	1	Administration	1700
2	2	Marketing	1800
3	3	Purchasing	1700
4	4	Human Resources	2400
5	5	Shipping	1500
6	6	IT	1400
7	7	Public Relations	2700
8	8	Sales	2500
9	9	Executive	1700
10	10	Finance	1700
11	11	Accounting	1700



Scalar Subquery

- ❑ **The Scalar sub query returns one single value to outer query**
 - Uses operators: can be $>$, $<$, $=$, $>=$, or $<=$, $<>$
 - Value generated by the subquery must be of a comparable data type
- ❑ If the query returns multiple values, the DBMS will generate an error.
- ❑ If inner query returns an empty set, result is converted to NULL



Scalar Subquery

□ Syntax

SELECT

FROM

WHERE

θ

(**SELECT**
FROM
WHERE)

θ

can be

=

<>

>

>=

<

<=

Must ensure that the subquery returns a SINGLE VALUE, i.e. a table of one row and one column, or an ERROR results.



Scalar Subquery

Example

```
SELECT employee_id, first_name, last_name, salary
FROM Employees
WHERE salary > (SELECT AVG(salary) FROM employees)
ORDER BY salary
```

Inner query
results



	Salary_Average
1	9350.000000

	employee_id	first_name	last_name	salary
1	114	Den	Raphaely	11000
2	205	Shelley	Higgins	12000
3	108	Nancy	Greenberg	12000
4	201	Michael	Hartstein	13000
5	101	Neena	Kochhar	17000
6	102	Lex	De Haan	17000
7	100	Steven	King	24000



Multi-Valued Subquery

- ❑ Multi-valued subquery returns multiple values to the outer query
- ❑ Query uses the set comparison operators (IN, ALL, ANY).

SYMBOL	MEANING
IN	Equal to any member in a list
ANY	Return rows that match any value on a list
ALL	Return rows that match all values in a list



Multi-Valued Subquery

□ Syntax:

```
SELECT .....  
FROM .....  
WHERE ..... θθ  
              ( SELECT .....  
                FROM .....  
                WHERE ..... )
```

θθ
can be
IN
NOT IN
θ ANY
θ ALL
EXISTS
NOT EXISTS

θ is any of =, <>, >, >=, <, <=

**The subquery can return MULTIPLE VALUES,
i.e. one column of many rows.
However there is no problem if only one row is returned.**


Multi-Valued Subquery with IN

Example

```
SELECT employee_id, first_name, last_name,
       department_id
FROM   Employees
WHERE  department_id IN (SELECT department_id
                        FROM   departments
                        WHERE  location_id = 1700)
```

Can't use '=' comparison.
Subquery can return many
department_id

Inner query
results



	department_id
1	1
2	3
3	9
4	10
5	11

	employee_id	first_name	last_name	department_id
1	100	Steven	King	9
2	101	Neena	Kochhar	9
3	102	Lex	De Haan	9
4	108	Nancy	Greenberg	10
5	109	Daniel	Faviet	10
6	113	Luis	Popp	10
7	114	Den	Raphaely	3
8	200	Jennifer	Whalen	1
9	205	Shelley	Higgins	11
10	206	William	Gietz	11



Multi-Valued Subquery with ALL

❑ Example

```
SELECT employee_id, first_name, last_name, salary, department_id
FROM Employees
WHERE salary > ALL (SELECT salary
                     FROM employees
                     WHERE department_id = 10)
AND department_id <> 10
```

If '**>**' is *true* for every value, ALL returns *true*, else *false*.

	employee_id	first_name	last_name	salary	department_id
1	100	Steven	King	24000	9
2	101	Neena	Kochhar	17000	9
3	102	Lex	De Haan	17000	9
4	201	Michael	Hartstein	13000	2



Multi-Valued Subquery with ANY

❑ Example

```
SELECT employee_id, first_name, last_name, salary, department_id
FROM Employees
WHERE salary > ANY (SELECT salary
                     FROM employees
                     WHERE department_id = 10)
AND department_id <> 10
```

If '**>**' is *true* for at least one value, ANY returns *true*, else *false*.

	employee_id	first_name	last_name	salary	department_id
1	100	Steven	King	24000	9
2	101	Neena	Kochhar	17000	9
3	102	Lex	De Haan	17000	9
4	103	Alexander	Hunold	9000	6
5	114	Den	Raphaely	11000	3
6	120	Matthew	Weiss	8000	5
7	176	Jonathon	Taylor	8600	8
8	177	Jack	Livingston	8400	8
9	201	Michael	Hartstein	13000	2
10	205	Shelley	Higgins	12000	11
11	206	William	Gietz	8300	11



EXISTS/NOT EXISTS

- ❑ When a subquery is used with the keyword EXISTS, it functions as an existence test
 - True or false only – no rows passed back to outer query
- ❑ **EXISTS evaluates to TRUE or FALSE**
 - If any rows are returned by the subquery, EXISTS returns TRUE
 - If no rows are returned, EXISTS returns FALSE
- ❑ Syntax:

```
WHERE [NOT] EXISTS (subquery)
```



EXISTS/NOT EXISTS

- ❑ The keyword EXISTS does not follow a column name or other expression
- ❑ The SELECT list of a subquery introduced by EXISTS typically **only uses an asterisk (*)**



EXISTS/NOT EXISTS

❑ Example: Show all the employees information if there is at least one employee with a salary in excess of 200,000

```
SELECT employee_id, first_name, last_name, salary, department_id
FROM Employees
WHERE EXISTS (SELECT * FROM employees WHERE salary > 200000)
```

The SELECT list typically only uses an asterisk because no data will be returned



Correlated subqueries

❑ Example 1: Finds all employees whose salary is higher than the average salary of the employees in their departments:

```
SELECT employee_id, first_name, last_name, salary, department_id
FROM employees e
WHERE salary > (SELECT AVG(salary) FROM employees
                WHERE department_id = e.department_id)
```

	employee_id	first_name	last_name	salary	department_id
1	201	Michael	Hartstein	13000	2
2	120	Matthew	Weiss	8000	5
3	123	Shanta	Vollman	6500	5
4	103	Alexander	Hunold	9000	6
5	176	Jonathon	Taylor	8600	8
6	100	Steven	King	24000	9
7	108	Nancy	Greenberg	12000	10
8	205	Shelley	Higgins	12000	11



Correlated subqueries

❑ Example 2: returns all departments which have no employees:

```
SELECT
    department_id, department_name
FROM
    departments d
WHERE NOT EXISTS( SELECT * FROM employees e
                  WHERE e.department_id = d.department_id)
```

	department_id	department_name
1	7	Public Relations



Correlated subqueries

❑ Example 3: Which departments have exactly 2 employees working on them ?

```
SELECT * FROM Departments d
WHERE 2 = (SELECT count(*) FROM employees e
           WHERE e.department_id = d.department_id)
```



Subqueries in FROM clause

❑ Example: Get the maximum of the average departmental salaries.

```
SELECT FORMAT(MAX( AvSal ), 'C2') AS MaxAvSal
FROM ( SELECT department_id, ROUND(AVG( Salary),2) AS AvSal
      FROM employees
      GROUP BY department_id ) AS Emp_Averages
```

This is a useful way of applying an aggregate function to the result of an aggregate function.



Subqueries in SELECT clause

❑ Example: Get the additional amount of salary that each employee receives over and above the minimum company salary.

```
SELECT    employee_id, first_name, last_name,  
          ( Salary - ( SELECT    Min(Salary)  
                        FROM employees ) )    AS    Additional  
FROM      employees
```

This is a useful way of doing a scalar calculation in each row that requires the resulting value of an aggregation.



Multiple Table joins



Multiple Tables Joins

- ❑ Ability to combine (join) tables on common attributes is most important distinction between a relational database and other databases
- ❑ **Join clause is used to combine rows from two or more tables, based on a related column between them.**
- ❑ **Join is generally composed of an equality comparison between the foreign key and the primary key of related tables**



Multiple Tables Joins

- Type of joins
 - Equijoin or Inner Join
 - Self join
 - Outer join
 - Cross Join



The contents of Product table

	P_Code	P_Descript	P_InDate	P_QOH	P_Min	P_Price	P_Discount	V_Code
1	11QER/31	Power painter, 15 psi., 3-nozzle	2017-11-03	8	5	109.99	0.00	25595
2	13-Q2/P2	7.25-in. pwr. saw blade	2017-01-13	32	15	14.99	0.05	21344
3	14-Q1/L3	9.00-in. pwr. saw blade	2017-11-13	18	12	17.49	0.00	21344
4	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	2018-01-15	15	8	39.95	0.00	23119
5	1558-QW1	Hrd. cloth, 1/2-in., 3x50	2018-01-15	23	5	43.99	0.00	23119
6	2232/QTY	B&D jigsaw, 12-in. blade	2017-12-30	8	5	109.92	0.05	24288
7	2232/QWE	B&D jigsaw, 8-in. blade	2017-12-24	6	5	99.87	0.05	24288
8	2238/QPD	B&D cordless drill, 1/2-in.	2018-01-20	12	5	38.95	0.05	25595
9	23109-HB	Claw hammer	2018-01-20	23	10	9.95	0.10	21225
10	23114-AA	Sledge hammer, 12 lb.	2018-01-02	8	5	14.40	0.05	NULL
11	54778-2T	Rat-tail file, 1/8-in. fine	2017-12-15	43	20	4.99	0.00	21344
12	89-WRE-Q	Hicut chain saw, 16 in.	2018-02-07	11	5	256.99	0.05	24288
13	PVC23DRT	PVC pipe, 3.5-in., 8ft	2018-02-20	188	75	5.87	0.00	NULL
14	SM-18277	1.25-in. metal screw, 25	2018-03-01	172	75	6.99	0.00	21225
15	SW-23116	2.5-in. wd. screw, 50	2018-02-24	237	100	8.45	0.00	21231
16	WR3/TT3	Steel matting, 4'x8'x1/6", .5" ...	2018-01-17	18	5	119.95	0.10	25595



The contents of Vendor table

	V_Code	V_Name	V_Contact	V_AreaCode	V_Phone	V_State	V_Order
1	21225	Bryson, Inc.	Smithson	615	223-3234	TN	Y
2	21226	SuperLoo, Inc.	Flushing	904	215-8995	FL	N
3	21231	D&E Supply	Singh	615	228-3245	TN	Y
4	21344	Gomez Bros.	Ortega	615	889-2546	KY	N
5	22567	Dome Supply	Smith	901	678-1419	GA	N
6	23119	Randsets Ltd.	Anderson	901	678-3998	GA	Y
7	24004	Brackman Bros.	Browning	615	228-1410	TN	N
8	24288	ORDVA, Inc.	Hakford	615	898-1234	TN	Y
9	25443	B&K, Inc.	Smith	904	227-0093	FL	N
10	25501	Damal Supplies	Smythe	615	890-3529	TN	N
11	25595	Rubicon Systems	Orton	904	456-0092	FL	Y

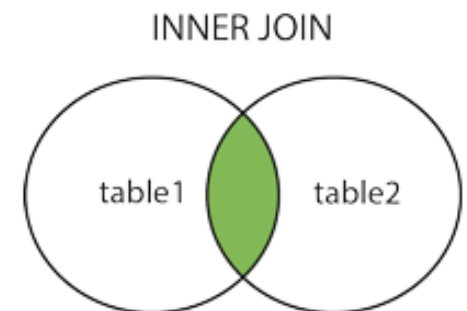
VKU Inner Join

❑ Also called as Equijoin

- This join combines the rows retrieved from two or more tables, based on a common field between them.
- This join creates a new result table by combining columns values of the tables based upon the join condition.
- To join n tables together, you need a minimum of $n-1$ join conditions

❑ Uses an equality operator (=) to join tables.

❑ INNER JOIN or simply JOIN is used in the query





Inner join

❑ The INNER JOIN keyword selects all rows from tables as long as there is a match between the columns

❑ ANSI syntax

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name
```

❑ You can join more than two tables based upon the join conditions.



Inner join

☐ Example

```
SELECT Product.P_Descript, Product.P_Price,  
Vendor.V_Name, Vendor.V_Contact, Vendor.V_AreaCode,  
Vendor.V_Phone  
FROM Product INNER JOIN Vendor  
ON Product.V_Code = Vendor.V_Code
```

☐ Or use EquiJoin based on *equality* V_Code (old-style join)

```
SELECT Product.P_Descript, Product.P_Price, Vendor.V_Name,  
Vendor.V_Contact, Vendor.V_AreaCode, Vendor.V_Phone  
FROM Product, Vendor  
WHERE Product.V_Code = Vendor.V_Code
```



Inner join

□ Result

	P_Descript	P_Price	V_Name	V_Contact	V_AreaCode	V_Phone
1	Power painter, 15 psi., 3-nozzle	109.99	Rubicon Systems	Orton	904	456-0092
2	7.25-in. pwr. saw blade	14.99	Gomez Bros.	Ortega	615	889-2546
3	9.00-in. pwr. saw blade	17.49	Gomez Bros.	Ortega	615	889-2546
4	Hrd. cloth, 1/4-in., 2x50	39.95	Randsets Ltd.	Anderson	901	678-3998
5	Hrd. cloth, 1/2-in., 3x50	43.99	Randsets Ltd.	Anderson	901	678-3998
6	B&D jigsaw, 12-in. blade	109.92	ORDVA, Inc.	Hakford	615	898-1234
7	B&D jigsaw, 8-in. blade	99.87	ORDVA, Inc.	Hakford	615	898-1234
8	B&D cordless drill, 1/2-in.	38.95	Rubicon Systems	Orton	904	456-0092
9	Claw hammer	9.95	Bryson, Inc.	Smithson	615	223-3234
10	Rat-tail file, 1/8-in. fine	4.99	Gomez Bros.	Ortega	615	889-2546
11	Hicut chain saw, 16 in.	256.99	ORDVA, Inc.	Hakford	615	898-1234
12	1.25-in. metal screw, 25	6.99	Bryson, Inc.	Smithson	615	223-3234
13	2.5-in. wd. screw, 50	8.45	D&E Supply	Singh	615	228-3245
14	Steel matting, 4'x8'x1/6", .5" mesh	119.95	Rubicon Systems	Orton	904	456-0092



Inner join

❑ Joining Tables With a Alias

- An alias may be used to identify the source table from which the data is taken

```
SELECT P.P_Descript,P.P_Price,V.V_Name,  
V.V_Contact,V.V_AreaCode, V.V_Phone  
FROM Product P INNER JOIN Vendor V  
ON P.V_Code = V.V_Code
```



Self Join

- ❑ A Self join or Recursive Join is a type of SQL join which is used to join a table to itself.
- ❑ The table has a Foreign key that references its own Primary key.
- ❑ An alias is especially useful when a table must be joined to itself in a recursive query to avoid column ambiguity.



Self Join

❑ Example: list of all employees with their managers names by joining the EMP table to itself.

```
SELECT E.EMP_Num, E.EMP_LName, E.EMP_MGR, M.EMP_LName
FROM EMP E
INNER JOIN EMP M
ON E.EMP_MGR = M.EMP_Num
ORDER BY E.EMP_MGR
```



The content of Emp table

	EMP_Num	EMP_Title	EMP_LName	EMP_FName	EMP_Initial	EMP_DOB	EMP_Hire_Date	EMP_AreaCode	EMP_Phone	EMP_MGR
1	100	Mr.	Kolmycz	George	D	1942-06-15 00:00:00.000	1985-03-15 00:00:00.000	615	324-5456	NULL
2	101	Ms.	Lewis	Rhonda	G	1965-03-19 00:00:00.000	1986-04-25 00:00:00.000	615	324-4472	100
3	102	Mr.	Vandam	Rhett	NULL	1958-11-14 00:00:00.000	1990-12-20 00:00:00.000	901	675-8993	100
4	103	Ms.	Jones	Anne	M	1974-10-16 00:00:00.000	1994-08-28 00:00:00.000	615	898-3456	100
5	104	Mr.	Lange	John	P	1971-11-08 00:00:00.000	1994-10-20 00:00:00.000	901	504-4430	105
6	105	Mr.	Williams	Robert	D	1975-03-14 00:00:00.000	1998-11-08 00:00:00.000	615	890-3220	NULL
7	106	Mrs.	Smith	Jeanine	K	1968-02-12 00:00:00.000	1989-01-05 00:00:00.000	615	324-7883	105
8	107	Mr.	Diante	Jorge	D	1974-08-21 00:00:00.000	1994-07-02 00:00:00.000	615	890-4567	105
9	108	Mr.	Wiesenbach	Paul	R	1966-02-14 00:00:00.000	1992-11-18 00:00:00.000	615	897-4358	NULL
10	109	Mr.	Smith	George	K	1961-06-18 00:00:00.000	1989-04-14 00:00:00.000	901	504-3339	108
11	110	Mrs.	Genkazi	Leighla	W	1970-05-19 00:00:00.000	1990-12-01 00:00:00.000	901	569-0093	108
12	111	Mr.	Washington	Rupert	E	1966-01-03 00:00:00.000	1993-06-21 00:00:00.000	615	890-4925	105
13	112	Mr.	Johnson	Edward	E	1961-05-14 00:00:00.000	1983-12-01 00:00:00.000	615	898-4387	100
14	113	Ms.	Smythe	Melanie	P	1970-09-15 00:00:00.000	1999-05-11 00:00:00.000	615	324-9006	105
15	114	Ms.	Brandon	Marie	G	1956-11-02 00:00:00.000	1979-11-15 00:00:00.000	901	882-0845	108
16	115	Mrs.	Saranda	Hemine	R	1972-07-25 00:00:00.000	1993-04-23 00:00:00.000	615	324-5505	105
17	116	Mr.	Smith	George	A	1965-11-08 00:00:00.000	1988-12-10 00:00:00.000	615	890-2984	108



Self Join

□ Result

	EMP_Num	EMP_LName	EMP_MGR	EMP_LName
1	101	Lewis	100	Kolmycz
2	102	Vandam	100	Kolmycz
3	103	Jones	100	Kolmycz
4	112	Johnson	100	Kolmycz
5	113	Smythe	105	Williams
6	115	Saranda	105	Williams
7	111	Washington	105	Williams
8	104	Lange	105	Williams
9	106	Smith	105	Williams
10	107	Diante	105	Williams
11	109	Smith	108	Wiesenbach
12	110	Genkazi	108	Wiesenbach
13	116	Smith	108	Wiesenbach
14	114	Brandon	108	Wiesenbach



The content of Product table

	P_Code	P_Descript	P_InDate	P_QOH	P_Min	P_Price	P_Discount	V_Code
1	11QER/31	Power painter, 15 psi., 3-nozzle	2017-11-03	8	5	109.99	0.00	25595
2	13-Q2/P2	7.25-in. pwr. saw blade	2017-01-13	32	15	14.99	0.05	21344
3	14-Q1/L3	9.00-in. pwr. saw blade	2017-11-13	18	12	17.49	0.00	21344
4	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	2018-01-15	15	8	39.95	0.00	23119
5	1558-QW1	Hrd. cloth, 1/2-in., 3x50	2018-01-15	23	5	43.99	0.00	23119
6	2232/QTY	B&D jigsaw, 12-in. blade	2017-12-30	8	5	109.92	0.05	24288
7	2232/QWE	B&D jigsaw, 8-in. blade	2017-12-24	6	5	99.87	0.05	24288
8	2238/QPD	B&D cordless drill, 1/2-in.	2018-01-20	12	5	38.95	0.05	25595
9	23109-HB	Claw hammer	2018-01-20	23	10	9.95	0.10	21225
10	23114-AA	Sledge hammer, 12 lb.	2018-01-02	8	5	14.40	0.05	NULL
11	54778-2T	Rat-tail file, 1/8-in. fine	2017-12-15	43	20	4.99	0.00	21344
12	89-WRE-Q	Hicut chain saw, 16 in.	2018-02-07	11	5	256.99	0.05	24288
13	PVC23DRT	PVC pipe, 3.5-in., 8-ft	2018-02-20	188	75	5.87	0.00	NULL
14	SM-18277	1.25-in. metal screw, 25	2018-03-01	172	75	6.99	0.00	21225
15	SW-23116	2.5-in. wd. screw, 50	2018-02-24	237	100	8.45	0.00	21231
16	WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	2018-01-17	18	5	119.95	0.10	25595



The content of Vendor table

	V_Code	V_Name	V_Contact	V_AreaCode	V_Phone	V_State	V_Order
1	21225	Bryson, Inc.	Smithson	615	223-3234	TN	Y
2	21226	SuperLoo, Inc.	Flushing	904	215-8995	FL	N
3	21231	D&E Supply	Singh	615	228-3245	TN	Y
4	21344	Gomez Bros.	Ortega	615	889-2546	KY	N
5	22567	Dome Supply	Smith	901	678-1419	GA	N
6	23119	Randsets Ltd.	Anderson	901	678-3998	GA	Y
7	24004	Brackman Bros.	Browning	615	228-1410	TN	N
8	24288	ORDVA, Inc.	Hakford	615	898-1234	TN	Y
9	25443	B&K, Inc.	Smith	904	227-0093	FL	N
10	25501	Damal Supplies	Smythe	615	890-3529	TN	N
11	25595	Rubicon Systems	Orton	904	456-0092	FL	Y



Outer join

- ❑ Outer join is a join in which rows that do not have matching values in common columns are also included in the result table .
- ❑ Null values appear in columns where there is not a match between tables
- ❑ Outer join
 - Left outer join
 - Right outer join
 - Full outer join



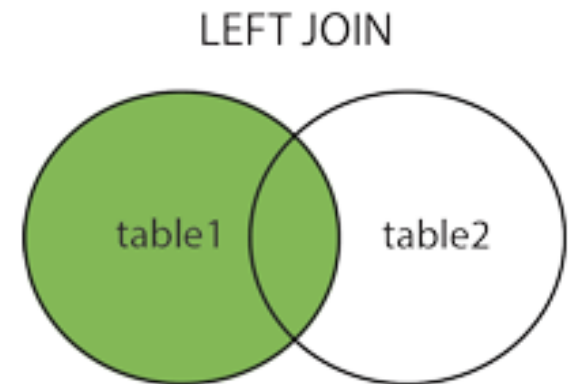
Left Outer Join

❑ In Left outer join (or Left Join)

- Returns all rows from the left table (table1) and the matching rows from the right table (table2)
- If no records match from the left table, it shows those records with NULL values

❑ Syntax:

```
SELECT column_name(s)  
FROM table1  
LEFT JOIN table2  
ON table1.column_name = table2.column_name
```





Left Outer Join

- ❑ Example: Show V_Code, V_Name for all vendors listed in the Vendor table. Include P_Code, P_Descript even if there is no product for that vendor.

```
SELECT  V.V_Code, V_Name, P.P_Code, P.P_Descript  
FROM    VENDOR V LEFT JOIN PRODUCT P  
ON      V.V_Code = P.V_Code
```



Left Outer Join

□ Result

	V_Code	V_Name	P_Code	P_Descript
1	21225	Bryson, Inc.	23109-HB	Claw hammer
2	21225	Bryson, Inc.	SM-18277	1.25-in. metal screw, 25
3	21226	SuperLoo, Inc.	NULL	NULL
4	21231	D&E Supply	SW-23116	2.5-in. wd. screw, 50
5	21344	Gomez Bros.	13-Q2/P2	7.25-in. pwr. saw blade
6	21344	Gomez Bros.	14-Q1/L3	9.00-in. pwr. saw blade
7	21344	Gomez Bros.	54778-2T	Rat-tail file, 1/8-in. fine
8	22567	Dome Supply	NULL	NULL
9	23119	Randsets Ltd.	1546-QQ2	Hrd. cloth, 1/4-in., 2x50
10	23119	Randsets Ltd.	1558-QW1	Hrd. cloth, 1/2-in., 3x50
11	24004	Brackman Bros.	NULL	NULL
12	24288	ORDVA, Inc.	2232/QTY	B&D jigsaw, 12-in. blade
13	24288	ORDVA, Inc.	2232/QWE	B&D jigsaw, 8-in. blade
14	24288	ORDVA, Inc.	89-WRE-Q	Hicut chain saw, 16 in.
15	25443	B&K, Inc.	NULL	NULL
16	25501	Damal Supplies	NULL	NULL
17	25595	Rubicon Systems	11QER/31	Power painter, 15 psi., 3-nozzle
18	25595	Rubicon Systems	2238/QPD	B&D cordless drill, 1/2-in.
19	25595	Rubicon Systems	WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh



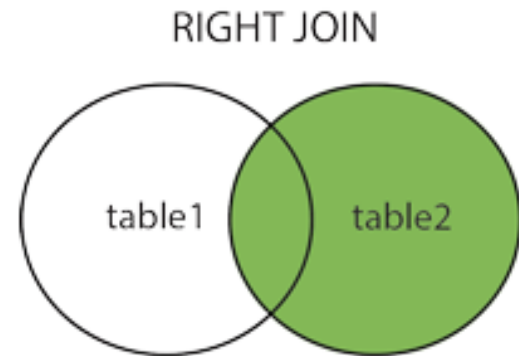
Right Outer Join

❑ In Right outer join (or Right Join)

- Returns all rows from the right table (table2) and the matching rows from the left table (table1)
- If no records match from the right table, it shows those records with NULL values

❑ Syntax

```
SELECT column_name(s)  
FROM table1  
RIGHT JOIN table2  
ON table1.column_name = table2.column_name
```





Right Outer Join

❑ Example: Show P_Code, P_Descript for all products listed in the Product table. Include V_Code, V_Name even if there is no Vendor for that product.

```
SELECT  P.P_Code, P.P_Descript, P.V_Code, V_Name
FROM    VENDOR V RIGHTJOIN PRODUCT P
ON      V.V_Code = P.V_Code
```



Right Outer Join

□ Result

	P_Code	P_Descript	V_Code	V_Name
3	14-Q1/L3	9.00-in. pwr. saw blade	21344	Gomez Bros.
4	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	23119	Randsets Ltd.
5	1558-QW1	Hrd. cloth, 1/2-in., 3x50	23119	Randsets Ltd.
6	2232/QTY	B&D jigsaw, 12-in. blade	24288	ORDVA, Inc.
7	2232/QWE	B&D jigsaw, 8-in. blade	24288	ORDVA, Inc.
8	2238/QPD	B&D cordless drill, 1/2-in.	25595	Rubicon Systems
9	23109-HB	Claw hammer	21225	Bryson, Inc.
10	23114-AA	Sledge hammer, 12 lb.	NULL	NULL
11	54778-2T	Rat-tail file, 1/8-in. fine	21344	Gomez Bros.
12	89-WRE-Q	Hicut chain saw, 16 in.	24288	ORDVA, Inc.
13	PVC23DRT	PVC pipe, 3.5-in., 8-ft	NULL	NULL
14	SM-18277	1.25-in. metal screw, 25	21225	Bryson, Inc.
15	SW-23116	2.5-in. wd. screw, 50	21231	D&E Supply
16	WR3/TT3	Steel matting, 4'x8'x1/6", .5" ...	25595	Rubicon Systems



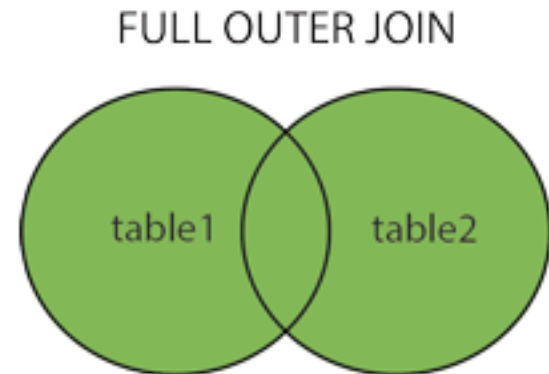
Full Outer Join

❑ Full Outer Join or Full Join

- returns all records when there is a match in left (table1) or right (table2) table records.

❑ Syntax

```
SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name = table2.column_name
WHERE condition
```





Full Outer Join

❑ Example: List P_Code, P_Descript, V_Code and V_Name of all products and vendors, regardless of whether or not the products have vendor or the vendors supply the products.

```
SELECT  P.P_Code, P.P_Descript, V.V_Code, V_Name
FROM    VENDOR V FULL JOIN PRODUCT P
ON      V.V_Code = P.V_Code
```



Full Outer Join

□ Result

	P_Code	P_Descript	V_Code	V_Name
1	23109-HB	Claw hammer	21225	Bryson, Inc.
2	SM-18277	1.25-in. metal screw, 25	21225	Bryson, Inc.
3	NULL	NULL	21226	SuperLoo, Inc.
4	SW-23116	2.5-in. wd. screw, 50	21231	D&E Supply
5	13-Q2/P2	7.25-in. pwr. saw blade	21344	Gomez Bros.
6	14-Q1/L3	9.00-in. pwr. saw blade	21344	Gomez Bros.
7	54778-2T	Rat-tail file, 1/8-in. fine	21344	Gomez Bros.
8	NULL	NULL	22567	Dome Supply
9	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	23119	Randsets Ltd.
10	1558-QW1	Hrd. cloth, 1/2-in., 3x50	23119	Randsets Ltd.
11	NULL	NULL	24004	Brackman Bros.
12	2232/QTY	B&D jigsaw, 12-in. blade	24288	ORDVA, Inc.
13	2232/QWE	B&D jigsaw, 8-in. blade	24288	ORDVA, Inc.
14	89-WRE-Q	Hicut chain saw, 16 in.	24288	ORDVA, Inc.
15	NULL	NULL	25443	B&K, Inc.
16	NULL	NULL	25501	Damal Supplies
17	11QER/31	Power painter, 15 psi., 3-nozzle	25595	Rubicon Systems
18	2238/QPD	B&D cordless drill, 1/2-in.	25595	Rubicon Systems
19	WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	25595	Rubicon Systems
20	23114-AA	Sledge hammer, 12 lb.	NULL	NULL
21	PVC23DRT	PVC pipe, 3.5-in., 8-ft	NULL	NULL



Cross Join

- ❑ A cross join performs a relational product (also known as the *Cartesian product*) of two tables

```
SELECT column_name(s)
FROM table1
CROSS JOIN table2
```



Cross Join

❑ Example

```
SELECT  P.P_Code, P.P_Descript, V.V_Code, V_Name  
FROM    VENDOR V CROSSJOIN PRODUCT P
```

❑ Or using other syntax (all DBMSs use this style)

```
SELECT  P.P_Code, P.P_Descript, V.V_Code, V_Name  
FROM    VENDOR V, PRODUCT P
```



Cross Join

❑ The command performs a cross join of the **VENDOR** and **PRODUCT** tables that generates 176 rows. (There are 11 vendor rows and 16 product rows, yielding $11 \times 16 = 176$ rows.)

❑ Some rows of the result

	P_Code	P_Descript	V_Code	V_Name
1	11QER/31	Power painter, 15 psi., 3-nozzle	21225	Bryson, Inc.
2	13-Q2/P2	7.25-in. pwr. saw blade	21225	Bryson, Inc.
3	14-Q1/L3	9.00-in. pwr. saw blade	21225	Bryson, Inc.
4	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	21225	Bryson, Inc.
5	1558-QW1	Hrd. cloth, 1/2-in., 3x50	21225	Bryson, Inc.
6	2232/QTY	B&D jigsaw, 12-in. blade	21225	Bryson, Inc.
7	2232/QWE	B&D jigsaw, 8-in. blade	21225	Bryson, Inc.
8	2238/QPD	B&D cordless drill, 1/2-in.	21225	Bryson, Inc.
9	23109-HB	Claw hammer	21225	Bryson, Inc.
10	23114-AA	Sledge hammer, 12 lb.	21225	Bryson, Inc.
11	54778-2T	Rat-tail file, 1/8-in. fine	21225	Bryson, Inc.
12	89-WRE-Q	Hicut chain saw, 16 in.	21225	Bryson, Inc.
13	PVC23DRT	PVC pipe, 3.5-in., 8ft	21225	Bryson, Inc.
14	SM-18277	1.25-in. metal screw, 25	21225	Bryson, Inc.
15	SW-23116	2.5-in. wd. screw, 50	21225	Bryson, Inc.
16	WR3/TT3	Steel matting, 4x8x1/6", .5" ...	21225	Bryson, Inc.
17	11QER/31	Power painter, 15 psi., 3-nozzle	21226	SuperLoo, Inc.
18	13-Q2/P2	7.25-in. pwr. saw blade	21226	SuperLoo, Inc.
19	14-Q1/L3	9.00-in. pwr. saw blade	21226	SuperLoo, Inc.
20	1546-QQ2	Hrd. cloth, 1/4-in., 2x50	21226	SuperLoo, Inc.



ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

Thank You !