JOHNS HOPKINS
CAREY BUSINESS SCHOOL

Database Management

BU.330.770

Session 5 (Part II)

Instructor: Changmi Jung, Ph.D.

Joining Data from Multiple Tables

Session Objectives (1/2)

- Identify a Cartesian join
- Create an equality join using the WHERE clause
- Create an equality join using the JOIN keyword
- Create a non-equality join using the WHERE clause
- Create a non-equality join using the JOIN…ON approach

Session Objectives (2/2)

- Create a self-join using the WHERE clause
- Create a self-join using the JOIN keyword
- Distinguish an inner join from an outer join
- Create an outer join using the WHERE clause
- Create an outer join using the OUTER keyword
- Use set operators (UNION, INTERSECT, MINUS) to combine the results of multiple queries

Purpose of Joins

- Joins are used to link tables and reconstruct data in a relational database

  (i) What was the sales volume for books in the family category last month?
  (ii) Customers in which state purchase books priced higher than $50?

- Joins can be created through:
  - Conditions in a **WHERE** clause
  - Use of **JOIN** keywords in FROM clause

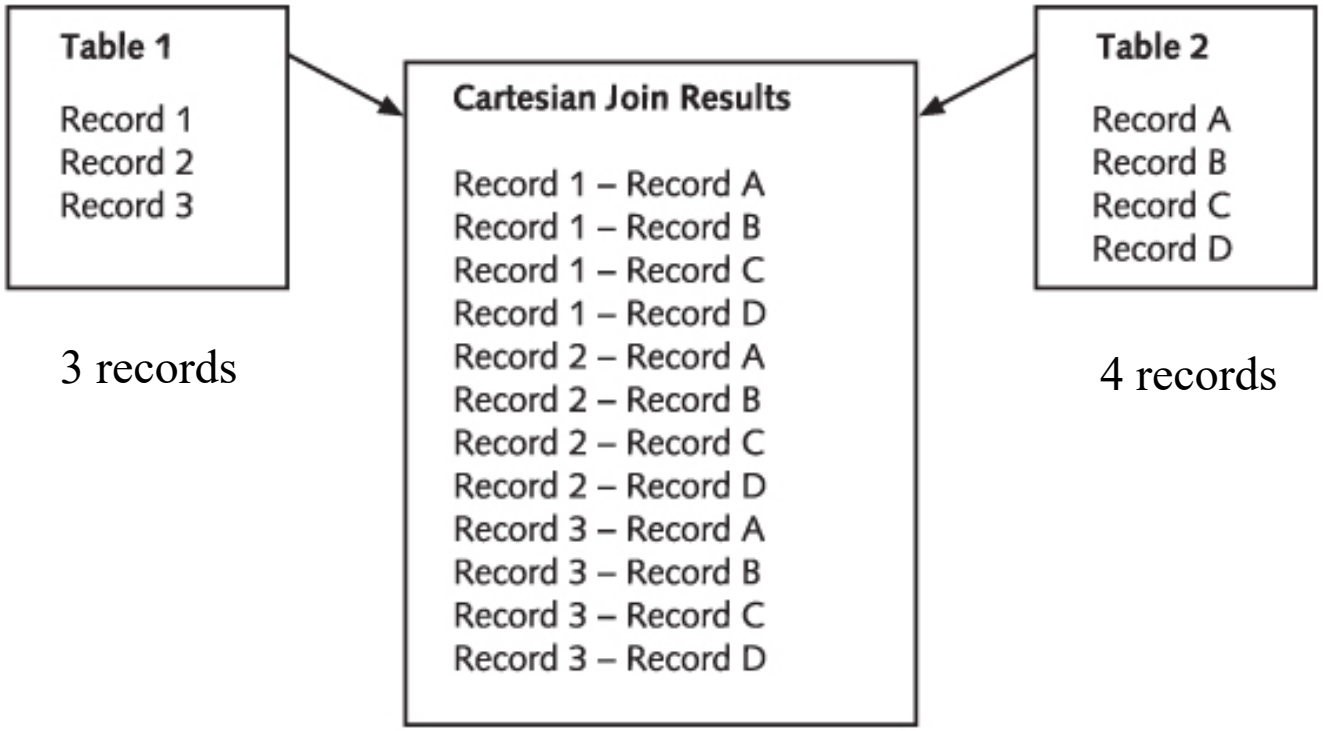
Using the JOIN keyword is preferable

Cartesian Joins

- Results in every possible row combination of records from two tables (generates *m * n records*)
- Rarely used
- Created when
  - Omitting joining condition in the WHERE clause
  - CROSS JOIN keywords in the FROM clause

Cartesian join is also called a Cartesian product or Cross Join.

Cartesian Join Results



3 records

4 records

3 x 4 = 12 records

Cartesian Join Example: Omitted Join Condition

Worksheet | Query Builder

```
SELECT title, name
FROM books, publisher;
```

Script Output | Query Result

SQL | All Rows Fetched: 70 in 0.031 seconds

	TITLE	NAME
1	BODYBUILD IN 10 MINUTES A DAY	PRINTING IS US
2	REVENGE OF MICKEY	PRINTING IS US
3	BUILDING A CAR WITH TOOTHPICKS	PRINTING IS US
4	DATABASE IMPLEMENTATION	PRINTING IS US
5	COOKING WITH MUSHROOMS	PRINTING IS US
6	HOLY GRAIL OF ORACLE	PRINTING IS US
7	HANDCRANKED COMPUTERS	PRINTING IS US
8	E-BUSINESS THE EASY WAY	PRINTING IS US
9	PAINLESS CHILD-REARING	PRINTING IS US
10	THE WOK WAY TO COOK	PRINTING IS US
11	BIG BEAR AND LITTLE DOVE	PRINTING IS US
12	HOW TO GET FASTER PIZZA	PRINTING IS US
13	HOW TO MANAGE THE MANAGER	PRINTING IS US
14	SHORTEST POEMS	PRINTING IS US
15	BODYBUILD IN 10 MINUTES A DAY	PUBLISH OUR WAY
16	REVENGE OF MICKEY	PUBLISH OUR WAY
17	BUILDING A CAR WITH TOOTHPICKS	PUBLISH OUR WAY
18	DATABASE IMPLEMENTATION	PUBLISH OUR WAY

Without the join condition, SQL automatically produces a Cartesian join.

70 records

Results in 70 rows =
14 rows from BOOKS
x 5 rows from PUBLISHER

Cartesian Join Example: CROSS JOIN Keywords

Worksheet | Query Builder

```
SELECT isbn, title, location, '          ' count
FROM books CROSS JOIN warehouses
ORDER BY location, title;
```

Script Output | Query Result

SQL | All Rows Fetched: 42 in 0.021 seconds

	ISBN	TITLE	LOCATION	COUNT
1	8117949391	BIG BEAR AND LITTLE DOVE	Boston	
2	1059831198	BODYBUILD IN 10 MINUTES A DAY	Boston	
3	4981341710	BUILDING A CAR WITH TOOTHPICKS	Boston	
4	3437212490	COOKING WITH MUSHROOMS	Boston	
5	8843172113	DATABASE IMPLEMENTATION	Boston	
6	9959789321	E-BUSINESS THE EASY WAY	Boston	
7	1915762492	HANDCRANKED COMPUTERS	Boston	
8	3957136468	HOLY GRAIL OF ORACLE	Boston	
9	0132149871	HOW TO GET FASTER PIZZA	Boston	
10	9247381001	HOW TO MANAGE THE MANAGER	Boston	
11	2491748320	PAINLESS CHILD-REARING	Boston	
12	0401140733	REVENGE OF MICKEY	Boston	
13	2147428890	SHORTEST POEMS	Boston	
14	0299282519	THE WOK WAY TO COOK	Boston	
15	8117949391	BIG BEAR AND LITTLE DOVE	Norfolk	
16	1059831198	BODYBUILD IN 10 MINUTES A DAY	Norfolk	
17	4981341710	BUILDING A CAR WITH TOOTHPICKS	Norfolk	
18	3437212490	COOKING WITH MUSHROOMS	Norfolk	
19	8843172113	DATABASE IMPLEMENTATION	Norfolk	

NO Commas before or after CROSS JOIN

42 records

Equality Joins

- Link rows through equivalent data that exists in both tables

  equivalent data? Both Books and Publisher tables have a common field PUBID – match the two using equality condition

- Created by:
  - Creating equivalency condition in the WHERE clause
  - Using NATURAL JOIN, JOIN…USING, or JOIN…ON keywords in the FROM clause

- Also called equijoin, inner join, or simple join

Equality Joins: WHERE Clause Example

Worksheet | Query Builder

```
SELECT title, name
FROM books, publisher
WHERE books.pubid = publisher.pubid;
```

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.025 seconds

	TITLE	NAME
1	REVENGE OF MICKEY	PRINTING IS US
2	HOW TO MANAGE THE MANAGER	PRINTING IS US
3	E-BUSINESS THE EASY WAY	PUBLISH OUR WAY
4	BUILDING A CAR WITH TOOTHPICKS	PUBLISH OUR WAY
5	HOLY GRAIL OF ORACLE	AMERICAN PUBLISHING
6	DATABASE IMPLEMENTATION	AMERICAN PUBLISHING
7	HANDCRANKED COMPUTERS	AMERICAN PUBLISHING
8	COOKING WITH MUSHROOMS	READING MATERIALS INC.
9	THE WOK WAY TO COOK	READING MATERIALS INC.
10	HOW TO GET FASTER PIZZA	READING MATERIALS INC.
11	BODYBUILD IN 10 MINUTES A DAY	READING MATERIALS INC.
12	PAINLESS CHILD-REARING	REED-N-RITE
13	BIG BEAR AND LITTLE DOVE	REED-N-RITE
14	SHORTEST POEMS	REED-N-RITE

Column qualifier: tablename where the column *pubid* belongs to

Qualifying Column Names

- Columns existing in both tables must be qualified

Worksheet | Query Builder

```
SELECT title, pubid, name
FROM books, publisher
WHERE books.pubid = publisher.pubid;
```

Script Output | Query Result

SQL | Executing:SELECT title, pubid, nameFROM books, p

```
ORA-00918: column ambiguously defined
00918. 00000 - "column ambiguously defined"
*Cause:
*Action:
Error at Line: 123 Column: 15
```

WHERE Clause Supports Join and Other Search Conditions

- Search conditions can be added to the WHERE clause with join condition

Worksheet | Query Builder

```
SELECT title, books.pubid, name
FROM books, publisher
WHERE books.pubid = publisher.pubid
AND books.retail > 25;
```

Join condition

Search condition

Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.015 seconds

	TITLE	PUBID	NAME
1	HOW TO MANAGE THE MANAGER	1	PRINTING IS US
2	BUILDING A CAR WITH TOOTHPICKS	2	PUBLISH OUR WAY
3	E-BUSINESS THE EASY WAY	2	PUBLISH OUR WAY
4	HOLY GRAIL OF ORACLE	3	AMERICAN PUBLISHING
5	DATABASE IMPLEMENTATION	3	AMERICAN PUBLISHING
6	THE WOK WAY TO COOK	4	READING MATERIALS INC.
7	HOW TO GET FASTER PIZZA	4	READING MATERIALS INC.
8	BODYBUILD IN 10 MINUTES A DAY	4	READING MATERIALS INC.
9	SHORTEST POEMS	5	REED-N-RITE
10	PAINLESS CHILD-REARING	5	REED-N-RITE

Using Table Aliases

- Use table aliases to simplify the task of qualifying columns

Worksheet | Query Builder

```
SELECT b.title, b.pubid, p.name, b.cost
FROM books b, publisher p
WHERE b.pubid = p.pubid
AND (b.cost < 15 OR p.pubid = 1)
ORDER BY title;
```

Once table alias is given in FROM clause, we must use the alias in the SQL statement whenever the table is referenced.

Script Output | Query Result

SQL | All Rows Fetched: 4 in 0.027 seconds

	TITLE	PUBID	NAME	COST
1	BIG BEAR AND LITTLE DOVE	5	REED-N-RITE	5.32
2	COOKING WITH MUSHROOMS	4	READING MATERIALS INC.	12.5
3	HOW TO MANAGE THE MANAGER	1	PRINTING IS US	15.4
4	REVENGE OF MICKEY	1	PRINTING IS US	14.2

Joining More Than Two Tables

- Joining four tables requires three JOIN conditions

Worksheet | Query Builder

```
SELECT c.lastname, c.firstname, b.title
FROM customers c, orders o, orderitems oi, books b
WHERE c.customer# = o.customer#
AND o.order# = oi.order#
AND oi.isbn = b.isbn
ORDER BY c.lastname, c.firstname;
```

Three JOIN conditions

Script Output | Query Result

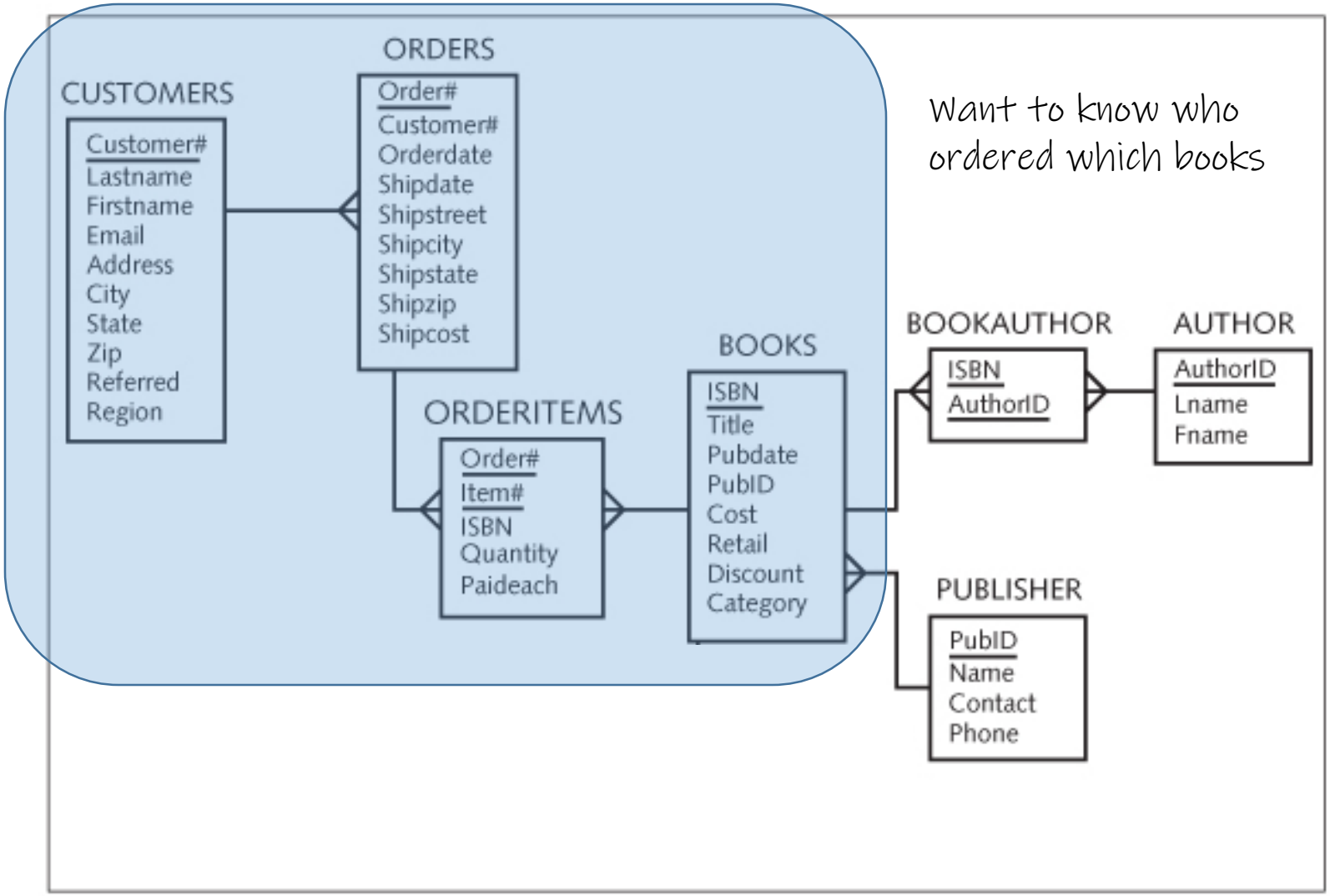
SQL | All Rows Fetched: 32 in 0.19 seconds

	LASTNAME	FIRSTNAME	TITLE
1	FALAH	KENNETH	PAINLESS CHILD-REARING
2	FALAH	KENNETH	COOKING WITH MUSHROOMS
3	GIANA	TAMMY	BIG BEAR AND LITTLE DOVE
4	GIANA	TAMMY	E-BUSINESS THE EASY WAY
5	GIANA	TAMMY	REVENGE OF MICKEY
6	GIANA	TAMMY	HOLY GRAIL OF ORACLE
7	GIANA	TAMMY	DATABASE IMPLEMENTATION
8	GIRARD	CINDY	REVENGE OF MICKEY
9	GIRARD	CINDY	COOKING WITH MUSHROOMS
10	GIRARD	CINDY	COOKING WITH MUSHROOMS
11	JONES	KENNETH	COOKING WITH MUSHROOMS

32 records

JustLee needs a list of customer names with all the books they have purchased!

Review the E-R Model



Equality Joins: NATURAL JOIN

- Creates a join automatically between two tables based on columns with matching names and types

  Joins on all common field between tables.

- Can't use a column qualifier for the common field

- Carefully use – two tables may have columns with the same name that are not related to each other

No column qualifier!

Worksheet | Query Builder

```
SELECT b.title, pubid, p.name
FROM books b NATURAL JOIN publisher p;
```

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.02 seconds

	TITLE	PUBID	NAME
1	REVENGE OF MICKEY	1	PRINTING IS US
2	HOW TO MANAGE THE MANAGER	1	PRINTING IS US
3	E-BUSINESS THE EASY WAY	2	PUBLISH OUR WAY
4	BUILDING A CAR WITH TOOTHPICKS	2	PUBLISH OUR WAY
5	HOLY GRAIL OF ORACLE	3	AMERICAN PUBLISHING
6	DATABASE IMPLEMENTATION	3	AMERICAN PUBLISHING
7	HANDCRANKED COMPUTERS	3	AMERICAN PUBLISHING
8	COOKING WITH MUSHROOMS	4	READING MATERIALS INC.
9	THE WOK WAY TO COOK	4	READING MATERIALS INC.
10	HOW TO GET FASTER PIZZA	4	READING MATERIALS INC.
11	BODYBUILD IN 10 MINUTES A DAY	4	READING MATERIALS INC.
12	PAINLESS CHILD-REARING	5	REED-N-RITE
13	BIG BEAR AND LITTLE DOVE	5	REED-N-RITE
14	SHORTEST POEMS	5	REED-N-RITE

No Qualifiers with a NATURAL JOIN

- When using NATURAL JOIN, do not use a column qualifier for the column used to join the tables.

We can't use a column qualifier for this column.

Worksheet | Query Builder

```
SELECT b.title, b.pubid, p.name
FROM books b NATURAL JOIN publisher p;
```

Script Output | Query Result

SQL | Executing:SELECT b.title, b.pubid, p.nameFROM bo

```
ORA-25155: column used in NATURAL join cannot have qualifier
25155. 00000 - "column used in NATURAL join cannot have qualifier"
*Cause:    Columns that are used for a named-join (either a NATURAL join
           or a join with a USING clause) cannot have an explicit qualifier.
*Action:   Remove the qualifier.
Error at Line: 171 Column: 17
```

Equality Joins: JOIN…USING

- Specify a column that will be used to join the tables
- Can't use a column qualifier for the column specified in USING clause

Worksheet | Query Builder

```
SELECT b.title, pubid, p.name
FROM books b JOIN publisher p
USING (pubid);
```

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.022 seconds

	TITLE	PUBID	NAME
1	REVENGE OF MICKEY	1	PRINTING IS US
2	HOW TO MANAGE THE MANAGER	1	PRINTING IS US
3	E-BUSINESS THE EASY WAY	2	PUBLISH OUR WAY
4	BUILDING A CAR WITH TOOTHPICKS	2	PUBLISH OUR WAY
5	HOLY GRAIL OF ORACLE	3	AMERICAN PUBLISHING
6	DATABASE IMPLEMENTATION	3	AMERICAN PUBLISHING
7	HANDCRANKED COMPUTERS	3	AMERICAN PUBLISHING
8	COOKING WITH MUSHROOMS	4	READING MATERIALS INC.
9	THE WOK WAY TO COOK	4	READING MATERIALS INC.
10	HOW TO GET FASTER PIZZA	4	READING MATERIALS INC.
11	BODYBUILD IN 10 MINUTES A DAY	4	READING MATERIALS INC.
12	PAINLESS CHILD-REARING	5	REED-N-RITE
13	BIG BEAR AND LITTLE DOVE	5	REED-N-RITE
14	SHORTEST POEMS	5	REED-N-RITE

Equality Joins: JOIN…ON

- Required if column names in the tables are different
- Most preferred way of joining tables

Worksheet | Query Builder

```
SELECT b.title, b.pubid, p.name
FROM publisher2 p JOIN books b
  ON p.id = b.pubid;
```

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.016 seconds

	TITLE	PUBID	NAME
1	BODYBUILD IN 10 MINUTES A DAY	4	READING MATERIALS INC.
2	REVENGE OF MICKEY	1	PRINTING IS US
3	BUILDING A CAR WITH TOOTHPICKS	2	PUBLISH OUR WAY
4	DATABASE IMPLEMENTATION	3	AMERICAN PUBLISHING
5	COOKING WITH MUSHROOMS	4	READING MATERIALS INC.
6	HOLY GRAIL OF ORACLE	3	AMERICAN PUBLISHING
7	HANDCRANKED COMPUTERS	3	AMERICAN PUBLISHING
8	E-BUSINESS THE EASY WAY	2	PUBLISH OUR WAY
9	PAINLESS CHILD-REARING	5	REED-N-RITE
10	THE WOK WAY TO COOK	4	READING MATERIALS INC.
11	BIG BEAR AND LITTLE DOVE	5	REED-N-RITE
12	HOW TO GET FASTER PIZZA	4	READING MATERIALS INC.
13	HOW TO MANAGE THE MANAGER	1	PRINTING IS US
14	SHORTEST POEMS	5	REED-N-RITE

Equality Joins: JOIN…ON (Multiple Joins)

Worksheet | Query Builder

```
SELECT c.lastname, c.firstname, b.title
FROM customers c JOIN orders O ON c.customer# = o.customer#
     JOIN orderitems oi ON o.order# = oi.order#
     JOIN books b ON oi.isbn = b.isbn
WHERE b.category = 'COMPUTER'
ORDER BY c.lastname, c.firstname;
```

Three JOIN conditions are used to join four tables

Script Output | Query Result

SQL | All Rows Fetched: 10 in 0.06 seconds

	LASTNAME	FIRSTNAME	TITLE
1	GIANA	TAMMY	DATABASE IMPLEMENTATION
2	GIANA	TAMMY	E-BUSINESS THE EASY WAY
3	GIANA	TAMMY	HOLY GRAIL OF ORACLE
4	LEE	JASMINE	DATABASE IMPLEMENTATION
5	MCGOVERN	REESE	DATABASE IMPLEMENTATION
6	MORALES	BONITA	DATABASE IMPLEMENTATION
7	MORALES	BONITA	DATABASE IMPLEMENTATION
8	NELSON	BECCA	HANDCRANKED COMPUTERS
9	SMITH	JENNIFER	DATABASE IMPLEMENTATION
10	SMITH	LEILA	E-BUSINESS THE EASY WAY

JustLee needs a list of customer names with all books in computer category each customer has purchased!

JOIN Keyword Overview

- Use JOIN…USING when tables have one or more columns in common
- Use JOIN…ON when the same named columns are not involved, or a condition is needed to specify a relationship other than equivalency (next section)
- Using the JOIN keyword frees the WHERE clause for exclusive use in restricting rows – WHERE is used only for search conditions, which makes the code much more organized and readable

Non-Equality Joins

- Sometimes, the relationships between columns from two tables are not based on equality conditions.
- In the WHERE clause, use any comparison operator other than the equal sign
- In the FROM clause, use JOIN…ON keywords with a non-equivalent condition

Non-Equality Joins

Suppose you need to send a package to your family for Christmas. The shipping fee charged by freight companies is based on the item's weight, and these shipping fees are mostly based on **a range of weights** rather than having a fee table for every single possible weight.

Shipping Fee Table:

Weight Range (kg)	Shipping Fee ($)
0 - 5	10
5.01 - 10	15
10.01 - 20	25
20.01 - 50	50
Above 50	100

```
MinWeight | MaxWeight | Fee
----------|-----------|-----
0         | 5         | 10
5.01      | 10        | 15
10.01     | 20        | 25
20.01     | 50        | 50
50.01     | NULL      | 100
```

Non-Equality Joins: WHERE, JOIN ON

Worksheet | Query Builder

```
SELECT b.title, p.gift
FROM books b, promotion p
WHERE b.retail BETWEEN p.minretail AND p.maxretail;

SELECT b.title, p.gift
FROM books b JOIN promotion p
  ON b.retail BETWEEN p.minretail AND p.maxretail;
```

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.046 seconds

	TITLE	GIFT
1	BIG BEAR AND LITTLE DOVE	BOOKMARKER
2	COOKING WITH MUSHROOMS	BOOK LABELS
3	REVENGE OF MICKEY	BOOK LABELS
4	HANDCRANKED COMPUTERS	BOOK LABELS
5	THE WOK WAY TO COOK	BOOK COVER
6	HOW TO GET FASTER PIZZA	BOOK COVER
7	BODYBUILD IN 10 MINUTES A DAY	BOOK COVER
8	HOW TO MANAGE THE MANAGER	BOOK COVER
9	SHORTEST POEMS	BOOK COVER
10	E-BUSINESS THE EASY WAY	BOOK COVER
11	DATABASE IMPLEMENTATION	BOOK COVER
12	BUILDING A CAR WITH TOOTHPICKS	FREE SHIPPING
13	HOLY GRAIL OF ORACLE	FREE SHIPPING
14	PAINLESS CHILD-REARING	FREE SHIPPING

- Note: make sure no range values overlap

Suppose JustLee Books offers a promotion based on the price of the books.

The Promotion table has several min and max retail price ranges for gifts.

Self-Joins

- Used to link a table to itself
  - Sometimes, data in one column of a table has a relationship with another column in the same table.

Refer your friend; you will get a $50 game credit when your friend signs up!

FORTNITE REFER A FRIEND 3.0: PLAY TOGETHER FOR IN-GAME REWARDS!

- Requires the use of table aliases (two same tables)

- Requires the use of a column qualifier: all columns are ambiguous

Image source: http://forbesdentalcare.com/new-patient-referral-program/
https://www.fortnite.com/news/fortnite-refer-a-friend-3-0-play-together-and-earn-rewards?lang=en-US

Customer Table Example

Customer 1003 (Leila Smith) has referred two customers (Tammy Giana and Jorge Perez)

CUSTOMER#	LASTNAME	FIRSTNAME	ADDRESS	CITY	STATE	ZIP	REFERRED
1001	MORALES	BONITA	P.O. BOX 651	EASTPOINT	FL	32328	
1002	THOMPSON	RYAN	P.O. BOX 9835	SANTA MONICA	CA	90404	
1003	SMITH	LEILA	P.O. BOX 66	TALLAHASSEE	FL	32306	
1004	PIERSON	THOMAS	69821 SOUTH AVENUE	BOISE	ID	83707	
1005	GIRARD	CINDY	P.O. BOX 851	SEATTLE	WA	98115	
1006	CRUZ	MESHIA	82 DIRT ROAD	ALBANY	NY	12211	
1007	GIANA	TAMMY	9153 MAIN STREET	AUSTIN	TX	78710	1003
1008	JONES	KENNETH	P.O. BOX 137	CHEYENNE	WY	82003	
1009	PEREZ	JORGE	P.O. BOX 8564	BURBANK	CA	91510	1003
1010	LUCAS	JAKE	114 EAST SAVANNAH	ATLANTA	GA	30314	
1011	MCGOVERN	REESE	P.O. BOX 18	CHICAGO	IL	60606	
1012	MCKENZIE	WILLIAM	P.O. BOX 971	BOSTON	MA	02110	
1013	NGUYEN	NICHOLAS	357 WHITE EAGLE AVE.	CLERMONT	FL	34711	1006

Customer 1006 (Meshia Cruz) has referred one customer (Nicholas Nguyen)

customers who refer a new customer to JustLee Books receive a discount coupon for a future purchase.

Now, suppose we need to display the name of the customer who referred another customer.

Example with CUSTOMERS

Leila Smith referred Tammy Giana and earned a referral bonus.

Customers		Referring Customers	
Tammy	Giana	Leila	Smith

Customers' Names in CUSTOMERS

These referring customers' names are also in CUSTOMERS

The 'Referred' column contains referring customers' ID

Let's join Customers (C) and Customers (M) using:
C.Referred = M.customer#

Self-Joins: WHERE Clause Example

Worksheet | Query Builder

```
SELECT c.firstname, c.lastname, m.firstname "Referred by"
FROM customers c, customers m
WHERE c.referred = m.customer#;
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.026 seconds

	FIRSTNAME	LASTNAME	Referred by
1	TAMMY	GIANA	LEILA
2	JORGE	PEREZ	LEILA
3	JENNIFER	SMITH	LEILA
4	NICHOLAS	NGUYEN	MESHIA
5	MICHELL	DAUM	JAKE

Joining the CUSTOMERS table with itself by linking Referred with Customer#

Using the same table as a master table (lookup table)

- List the same table twice in the FROM clause: need to make them look like two different tables by **assigning table aliases**

Self-Joins: JOIN…ON Example

Worksheet | Query Builder

```
SELECT c.firstname, c.lastname, m.firstname "Referred by"
FROM customers c, customers m
WHERE c.referred = m.customer#;
```

Query Result

SQL | All Rows Fetched: 5 in 0.021 seconds

	FIRSTNAME	LASTNAME	Referred by
1	TAMMY	GIANA	LEILA
2	JORGE	PEREZ	LEILA
3	JENNIFER	SMITH	LEILA
4	NICHOLAS	NGUYEN	MESHIA
5	MICHELL	DAUM	JAKE

So far, equality join, non-equality join, and self-joins will return a row only if a corresponding record in each table is queried. –

Outer Joins

- Use outer joins to include rows that do not have a match in the other table

  You may want all customers and their orders displayed regardless of customers having any orders.

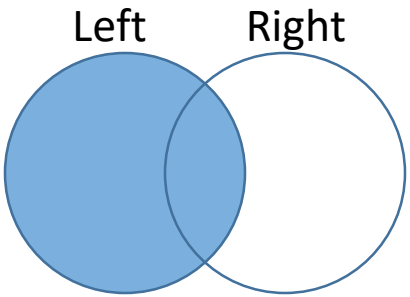
  A Principal wants the list of all students with their primary sports activity (some students may not participate in any sports teams).

- In FROM clause, use FULL, LEFT, or RIGHT with OUTER JOIN keywords

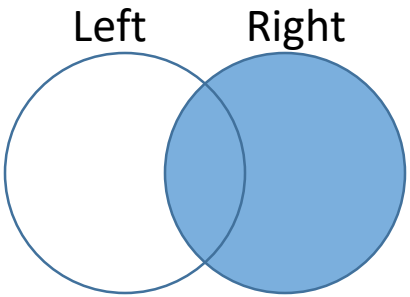
  Optional

Outer Joins: OUTER JOIN Keyword Example

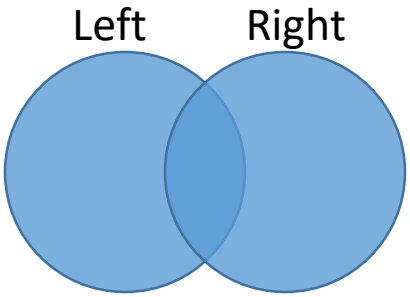
- LEFT (OUTER) JOIN: all records from the left side table will be included in the result regardless of the match.



- RIGHT (OUTER) JOIN: all records from the right side table will be included in the result regardless of the match



- FULL (OUTER) JOIN: all records from both tables will be included regardless of match.



How Does It Work?

Customer#	Lastname	Firstname	# orders placed
1001	MORALES	BONITA	2
1002	THOMPSON	RYAN	
1003	SMITH	LEILA	2
1004	PIERSON	THOMAS	1
1005	GIRARD	CINDY	2
1006	CRUZ	MESHIA	
1007	GIANA	TAMMY	2
1008	JONES	KENNETH	1
1009	PEREZ	JORGE	
1010	LUCAS	JAKE	2
1011	MCGOVERN	REESE	1
1012	MCKENZIE	WILLIAM	
1013	NGUYEN	NICHOLAS	
1014	LEE	JASMINE	1
1015	SCHELL	STEVE	1
1016	DAUM	MICHELL	
1017	NELSON	BECCA	1
1018	MONTIASA	GREG	2
1019	SMITH	JENNIFER	1
1020	FALAH	KENNETH	2
		Total # orders	21

Customers **LEFT JOIN** Orders

: 27 records = 21 matching records + 6 customers without orders will be matched with null orders

Customers **RIGHT JOIN** Orders

: 21 records = because all orders have exactly one corresponding customer

Customers **FULL JOIN** Orders

: 27 records = 21 matching records + 6 customers without orders

OUTER JOIN Examples

Worksheet | Query Builder

```
SELECT c.lastname, c.firstname, o.order#
FROM customers c LEFT OUTER JOIN orders o
    ON c.customer# = o.customer#
ORDER BY c.lastname, c.firstname;
```

Script Output | Query Result

SQL | All Rows Fetched: 27 in 0.019 seconds

	LASTNAME	FIRSTNAME	ORDER#
1	CRUZ	MESHIA	(null)
2	DAUM	MICHELL	(null)
3	FALAH	KENNETH	1004
4	FALAH	KENNETH	1015
5	GIANA	TAMMY	1007
6	GIANA	TAMMY	1014
7	GIRARD	CINDY	1009
8	GIRARD	CINDY	1000
9	JONES	KENNETH	1020
10	LEE	JASMINE	1013
11	LUCAS	JAKE	1001
12	LUCAS	JAKE	1011
13	MCGOVERN	REESE	1002
14	MCKENZIE	WILLIAM	(null)
15	MONTIASA	GREG	1005
16	MONTIASA	GREG	1019
17	MORALES	BONITA	1018

Worksheet | Query Builder

```
SELECT oi.order#, oi.item#, b.title
FROM orderitems oi RIGHT OUTER JOIN books b
    ON oi.isbn = b.isbn
ORDER BY b.title;
```

Script Output | Query Result

SQL | All Rows Fetched: 35 in 0.014 seconds

	ORDER#	ITEM#	TITLE
1	1017	1	BIG BEAR AND LITTLE DOVE
2	1007	3	BIG BEAR AND LITTLE DOVE
3	1012	1	BIG BEAR AND LITTLE DOVE
4	1003	2	BODYBUILD IN 10 MINUTES A DAY
5	(null)	(null)	BUILDING A CAR WITH TOOTHPICKS
6	1003	3	COOKING WITH MUSHROOMS
7	1000	1	COOKING WITH MUSHROOMS
8	1008	1	COOKING WITH MUSHROOMS
9	1020	1	COOKING WITH MUSHROOMS
10	1009	1	COOKING WITH MUSHROOMS
11	1018	1	COOKING WITH MUSHROOMS
12	1015	1	COOKING WITH MUSHROOMS
13	1010	1	DATABASE IMPLEMENTATION
14	1018	2	DATABASE IMPLEMENTATION
15	1013	1	DATABASE IMPLEMENTATION
16	1003	1	DATABASE IMPLEMENTATION
17	1007	4	DATABASE IMPLEMENTATION
18	1002	1	DATABASE IMPLEMENTATION

Set Operators

- Used to combine the results of two or more SELECT statements

Set Operator	Description
UNION	Returns the combined results of two queries and **suppresses duplicate rows**
UNION ALL	Returns the combined results of two queries but does not suppress duplicates
INTERSECT	Returns only those rows included in both queries and suppresses duplicate rows
MINUS	Returns rows selected by the first query that are not included in the second query

Set Operators: UNION Example

Worksheet | Query Builder

```
SELECT ba.authorid
FROM books b JOIN bookauthor ba
    USING (isbn)
WHERE b.category = 'FAMILY LIFE'
UNION
SELECT ba.authorid
FROM books b JOIN bookauthor ba
    USING (isbn)
WHERE b.category = 'CHILDREN';
```

A

J100
B100
F100
R100

B

K100
R100

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.063 seconds

	AUTHORID
1	B100
2	F100
3	J100
4	K100
5	R100

Union works like:
A U B

Where A is the first select query result and B is the second query result

List all author IDs of those who wrote books in the Children or Family Life categories.

Set Operators: Multiple Column Example

- All columns are included to perform the set comparison
- Each query must contain the <u>same number of columns in the matching order</u>
- Column names can be different in the queries
- Do not use a column qualifier in the ORDER BY clause

Worksheet | Query Builder

```
SELECT pubid, name
FROM publisher
UNION
SELECT id, name
FROM publisher3
ORDER BY pubid;
```

References the column names from the first query only

Script Output | Query Result

SQL | All Rows Fetched: 8 in 0.016 seconds

	PUBID	NAME
1	1	PRINTING IS US
2	2	PUBLISH OUR WAY
3	3	AMERICAN PUB
4	3	AMERICAN PUBLISHING
5	4	READING MATERIALS INC.
6	5	REED-N-RITE
7	6	PRINTING HERE
8	7	PRINT THERE

Set Operators: INTERSECT Example

Worksheet | Query Builder

```
SELECT customer#
FROM customers
INTERSECT
SELECT customer#
FROM orders;
```

A

B

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.021 se

	CUSTOMER#
1	1005
2	1010
3	1011
4	1001
5	1020
6	1018
7	1003
8	1007
9	1004
10	1019
11	1017
12	1014
13	1015
14	1008

INTERSECT works like:
A ∩ B

Returns customer numbers of customers who have placed any orders recently.

Set Operators: MINUS Example

Worksheet | Query Builder

```
SELECT customer#
FROM customers
MINUS
SELECT customer#
FROM orders;
```

A

B

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.01

	CUSTOMER#
1	1006
2	1002
3	1009
4	1013
5	1012
6	1016

MINUS works like:
A – (A ∩ B)

Returns customer numbers of the customers who have not placed any orders recently.

Summary (1/2)

- Data stored in multiple tables regarding a single entity can be linked together through the use of joins
- A Cartesian join between two tables returns every possible combination of rows from the tables; the resulting number of rows is always $m * n$
- An equality join is created when the data joining the records from two different tables are an exact match
- A non-equality join establishes a relationship based upon anything other than an equal condition
- Self-joins are used when a table must be joined to itself to retrieve needed data

Summary (2/2)

- Inner joins are categorized as being equality, non-equality, or self-joins
- An outer join is created when records need to be included in the results without having corresponding records in the join tables
  - The record is matched with a NULL record so it will be included in the output
- Set operators such as UNION, UNION ALL, INTERSECT, and MINUS can be used to combine the results of multiple queries

Let’s Check Our Learning!