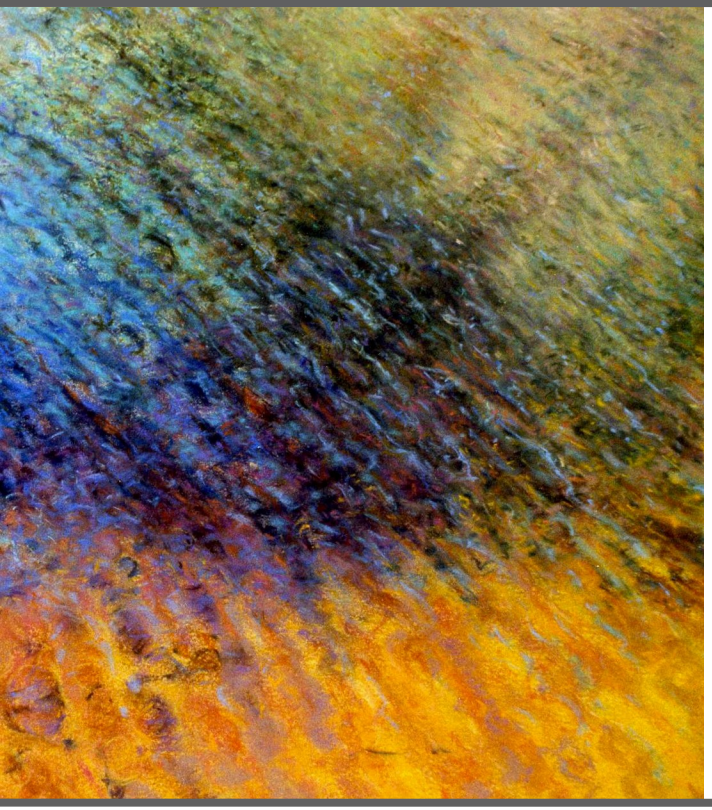
David M. Kroenke and David J. Auer

Database Processing:

Fundamentals, Design, and Implementation



Chapter Two:

Introduction to Structured Query Language

Part 2: Multiple Table Queries

Chapter Objectives

- Create SQL queries that retrieve data from a single table but restrict the data based upon data in another table (subquery)
- Create SQL queries that retrieve data from multiple tables using the SQL join and JOIN ON operations

Extracted Data Format

Table	Column	Data Type
RETAIL_ORDER	OrderNumber	Integer
	StoreNumber	Integer
	StoreZIP	Character (9)
	OrderMonth	Character (12)
	OrderYear	Integer
	OrderTotal	Currency
ORDER_ITEM	OrderNumber	Integer
	SKU	Integer
	Quantity	Integer
	Price	Currency
	ExtendedPrice	Currency
SKU_DATA	SKU	Integer
	SKU_Description	Character (35)
	Department	Character (30)
	Buyer	Character (30)
CATALOG_SKU_20##	CatalogID	Integer
	SKU	Integer
	SKU_Description	Character (35)
	Department	Character (30)
	CatalogPage	Integer
	DateOnWebSite	Date

Querying Two or More Tables with SQL

SQL provides two different techniques for querying data from multiple tables:

- The **SQL subquery**
- The **SQL join**

Querying Multiple Tables with SQL Subqueries
The Logic of Subqueries I

We want to know the revenue for Water Sports items, which have SKU values of 100100, 100200, 101100, and 101200.

Given that we know the SKU values, we can use this query:

```
/* *** SQL-Query-CH02-53 *** */
SELECT     SUM(ExtendedPrice) AS WaterSportsRevenue
FROM       ORDER_ITEM
WHERE      SKU IN (100100, 100200, 101100, 101200);
```

	WaterSportsRevenue
1	750.00

Querying Multiple Tables with SQL Subqueries
The Logic of Subqueries II

What if we don't know the SKU values? We can determine them with this query:

```
/* *** SQL-Query-CH02-54 *** */
SELECT      SKU
FROM        SKU_DATA
WHERE       Department = 'Water Sports'
```

	SKU
1	100100
2	100200
3	101100
4	101200

Querying Multiple Tables with SQL Subqueries
The Logic of Subqueries III

We can combine these two as shown in SQL-Query-02-55. The second query, which is enclosed in parentheses, is call an **SQL subquery**. Note how the subquery returns a set of values for use by the **top level query**, and note the use of the **SQL IN keyword**.

```
/* *** SQL-Query-CH02-55 *** */
SELECT     SUM(ExtendedPrice) AS WaterSportsRevenue
FROM       ORDER_ITEM
WHERE      SKU IN
           (SELECT    SKU
            FROM      SKU_DATA
            WHERE     Department = 'Water Sports');
```

	WaterSportsRevenue
1	750.00

Querying Multiple Tables with SQL Subqueries

```
/* *** SQL-Query-CH02-57 *** */
SELECT     Buyer, Department, COUNT(SKU) AS Number_Of_SKU_Sold
FROM       SKU_DATA
WHERE      SKU IN
           (SELECT    SKU
            FROM      ORDER_ITEM
            WHERE     OrderNumber  IN
                      (SELECT      OrderNumber
                       FROM        RETAIL_ORDER
                       WHERE       OrderMonth='January'
                         AND       OrderYear=2015))
GROUP BY   Buyer, Department
ORDER BY   Number_Of_SKU_Sold;
```

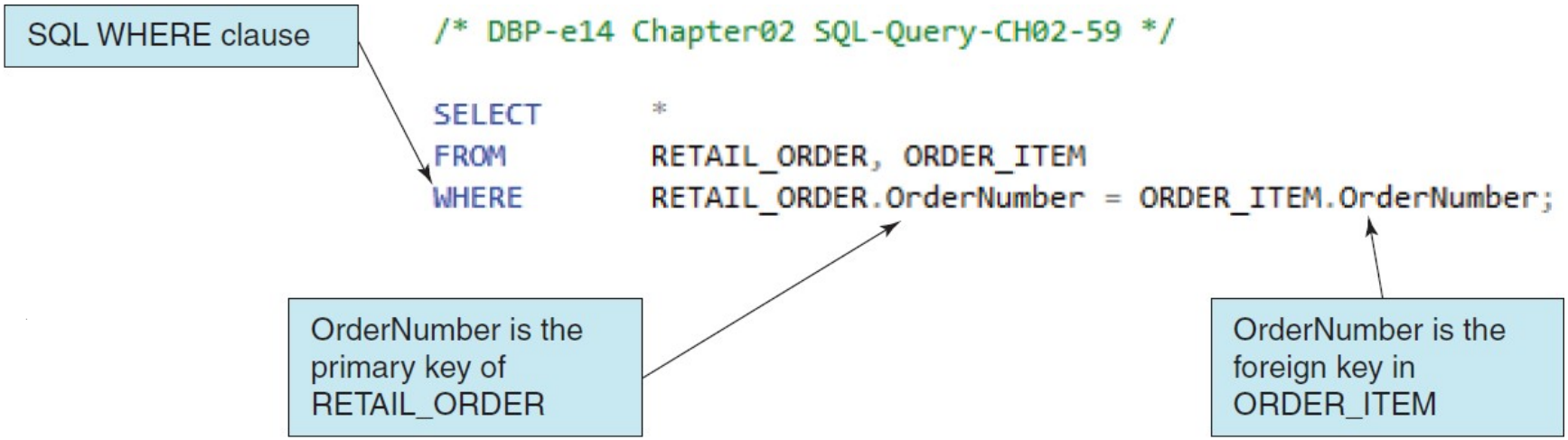
	Buyer	Department	Number_Of_SKU_Sold
1	Pete Hansen	Water Sports	1
2	Nancy Meyers	Water Sports	2

Querying Multiple Tables with SQL Joins
The Logic of Joins I

- In an **SQL join operation**, the **SQL JOIN operator** is used to combine parts or all of two or more tables.
  - **Explicit join** – the SQL JOIN operator is used as part of the SQL statement.
  - **Implicit join** – the SQL JOIN operator is ***not*** used as part of the SQL statement.

Querying Multiple Tables with SQL Joins
The Logic of Joins III – Implicit SQL INNER JOIN

- By selecting rows by matching by the primary key values of one table with the foreign key values of a second table, we produce an **SQL INNER JOIN**.



- Because the **SQL JOIN keyword** does ***not*** appear in the SQL statement, this is an **implicit join**.

Querying Multiple Tables with SQL Joins
The Logic of Joins IV – Implicit SQL INNER JOIN

```
/* *** SQL-Query-CH02-59 *** */

SELECT      *

FROM        RETAIL_ORDER, ORDER_ITEM

WHERE       RETAIL_ORDER.OrderNumber = ORDER_ITEM.OrderNumber;
```

	OrderNumber	StoreNumber	StoreZip	OrderMonth	OrderYear	OrderTotal	OrderNumber	SKU	Quantity	Price	ExtendedPrice
1	3000	10	98110	January	2015	480.00	3000	100200	1	300.00	300.00
2	2000	20	02335	December	2014	310.00	2000	101100	4	50.00	200.00
3	3000	10	98110	January	2015	480.00	3000	101100	2	50.00	100.00
4	2000	20	02335	December	2014	310.00	2000	101200	2	50.00	100.00
5	3000	10	98110	January	2015	480.00	3000	101200	1	50.00	50.00
6	1000	10	98110	December	2014	445.00	1000	201000	1	300.00	300.00
7	1000	10	98110	December	2014	445.00	1000	202000	1	130.00	130.00

Querying Multiple Tables with SQL Joins
The Logic of Joins V – Implicit SQL INNER JOIN

With an **SQL ORDER BY clause** for easier reading by OrderNumber

```
/* *** SQL-Query-CH02-60 *** */

SELECT      *

FROM        RETAIL_ORDER, ORDER_ITEM

WHERE       RETAIL_ORDER.OrderNumber=ORDER_ITEM.OrderNumber

ORDER BY    RETAIL_ORDER.OrderNumber, ORDER_ITEM.SKU;
```

	OrderNumber	StoreNumber	StoreZip	OrderMonth	OrderYear	OrderTotal	OrderNumber	SKU	Quantity	Price	ExtendedPrice
1	1000	10	98110	December	2014	445.00	1000	201000	1	300.00	300.00
2	1000	10	98110	December	2014	445.00	1000	202000	1	130.00	130.00
3	2000	20	02335	December	2014	310.00	2000	101100	4	50.00	200.00
4	2000	20	02335	December	2014	310.00	2000	101200	2	50.00	100.00
5	3000	10	98110	January	2015	480.00	3000	100200	1	300.00	300.00
6	3000	10	98110	January	2015	480.00	3000	101100	2	50.00	100.00
7	3000	10	98110	January	2015	480.00	3000	101200	1	50.00	50.00

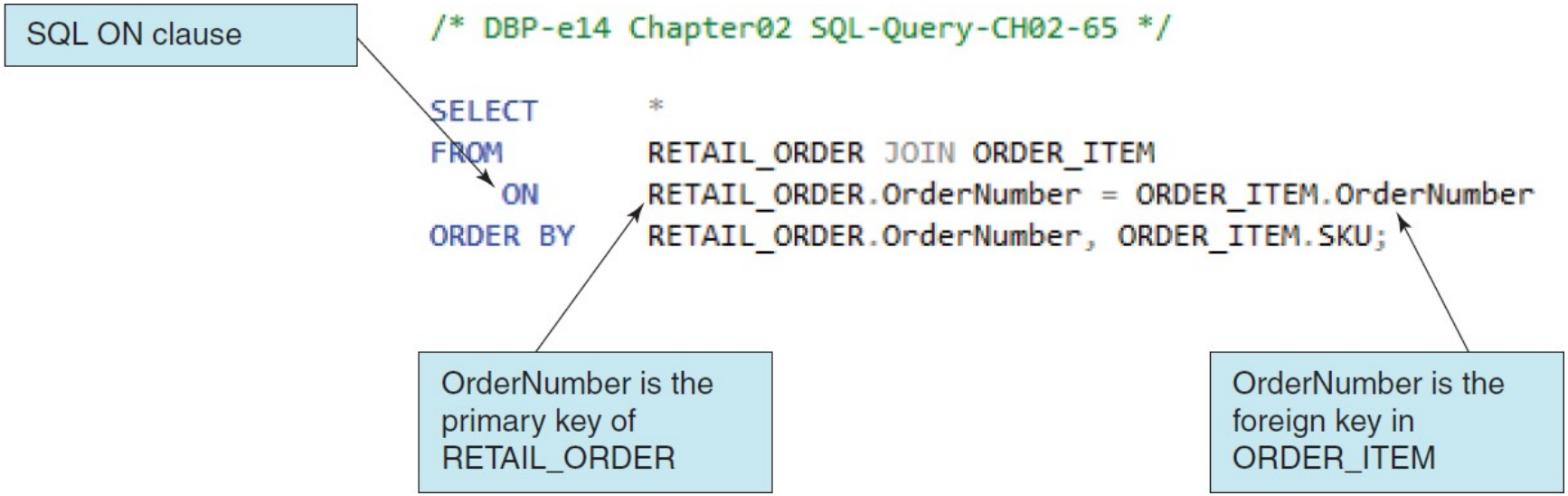
Querying Multiple Tables with SQL Joins
The Logic of Joins VII

```
/* *** SQL-Query-CH02-62 *** */
SELECT      Buyer, SKU_DATA.SKU, SKU_Description,
            SUM(ExtendedPrice) AS BuyerSKURevenue
FROM        SKU_DATA, ORDER_ITEM
WHERE       SKU_DATA.SKU=ORDER_ITEM.SKU
GROUP BY    Buyer, SKU_DATA.SKU, SKU_Description
ORDER BY    BuyerSKURevenue DESC;
```

	Buyer	SKU	SKU_Description	BuyerSKURevenue
1	Pete Hansen	100200	Std. Scuba Tank, Magenta	300.00
2	Nancy Meyers	101100	Dive Mask, Small Clear	300.00
3	Cindy Lo	201000	Half-dome Tent	300.00
4	Nancy Meyers	101200	Dive Mask, Med Clear	150.00
5	Cindy Lo	202000	Half-dome Tent Vestibule	130.00

Querying Multiple Tables with SQL Joins
The Logic of Joins VIII – Explicit SQL INNER JOIN

- By selecting rows by matching by the primary key values of one table with the foreign key values of a second table, we produce an **SQL INNER JOIN**.



- Because the **SQL JOIN keyword *does*** appear in the SQL statement, this is an **explicit join**.

Querying Multiple Tables with SQL Joins
The Logic of Joins X – SQL JOIN ON Syntax

```
/* *** SQL-Query-CH02-66 *** */
SELECT      *
FROM        RETAIL_ORDER JOIN ORDER_ITEM
      ON    RETAIL_ORDER.OrderNumber = ORDER_ITEM.OrderNumber
WHERE       OrderYear = '2014'
ORDER BY    RETAIL_ORDER.OrderNumber, ORDER_ITEM.SKU;
```

	OrderNumber	StoreNumber	StoreZip	OrderMonth	OrderYear	OrderTotal	OrderNumber	SKU	Quantity	Price	ExtendedPrice
1	1000	10	98110	December	2014	445.00	1000	201000	1	300.00	300.00
2	1000	10	98110	December	2014	445.00	1000	202000	1	130.00	130.00
3	2000	20	02335	December	2014	310.00	2000	101100	4	50.00	200.00
4	2000	20	02335	December	2014	310.00	2000	101200	2	50.00	100.00

Querying Multiple Tables with SQL Set Operators

The Logic of Set Operators VII – SQL Set Operators

Note that in order to use **SQL set operators**, the table columns involved in the operations ***must*** be the same number in each SELECT component, and corresponding columns ***must*** have the same or compatible (e.g., CHAR and VARCHAR) data types!

SQL Set Operators	
Operator	**Meaning**
UNION	The result is all the row values in one or both tables
INTERSECT	The result is all the row values common to both tables
EXCEPT	The result is all the row values in the first table but not the second

Querying Multiple Tables with SQL Set Operators

The Logic of Set Operators VIII – SQL UNION Operator

"What products were available for sale (by either catalog or Web site) in 2014 and 2015?"

```
/* *** SQL-Query-CH02-76 *** */
SELECT     SKU, SKU_Description, Department
FROM       CATALOG_SKU_2014
UNION
SELECT     SKU, SKU_Description, Department
FROM       CATALOG_SKU_2015;
```

	SKU	SKU_Description	Department
1	100100	Std. Scuba Tank, Yellow	Water Sports
2	100200	Std. Scuba Tank, Magenta	Water Sports
3	100300	Std. Scuba Tank, Light Blue	Water Sports
4	100400	Std. Scuba Tank, Dark Blue	Water Sports
5	101100	Dive Mask, Small Clear	Water Sports
6	101200	Dive Mask, Med Clear	Water Sports
7	201000	Half-dome Tent	Camping
8	202000	Half-dome Tent Vestibule	Camping
9	203000	Half-dome Tent Vestibule - Wide	Camping
10	301000	Light Fly Climbing Harness	Climbing
11	302000	Locking Carabiner, Oval	Climbing

Querying Multiple Tables with SQL Set Operators

The Logic of Set Operators X – SQL INTERSECT Operator

"What products were available for sale (by either catalog or Web site) in *both* 2014 and 2015?" [MySQL 5.6 does not support the INTERSECT Operator]

```
/* *** SQL-Query-CH02-77 *** */
SELECT      SKU, SKU_Description, Department
FROM        CATALOG_SKU_2014
INTERSECT
SELECT      SKU, SKU_Description, Department
FROM        CATALOG_SKU_2015;
```

	SKU	SKU_Description	Department
1	100100	Std. Scuba Tank, Yellow	Water Sports
2	101100	Dive Mask, Small Clear	Water Sports
3	101200	Dive Mask, Med Clear	Water Sports
4	201000	Half-dome Tent	Camping
5	202000	Half-dome Tent Vestibule	Camping
6	301000	Light Fly Climbing Harness	Climbing
7	302000	Locking Carabiner, Oval	Climbing

Querying Multiple Tables with SQL Set Operators

The Logic of Set Operators XI – SQL EXCEPT Operator

"What products were available for sale (by either catalog or Web site) in 2014 but *not* in 2015?" [Oracle Database calls this the **SQL MINUS operator**, and MySQL 5.6 does not support this operation]

```
/* *** SQL-Query-CH02-78 *** */
SELECT      SKU, SKU_Description, Department
FROM        CATALOG_SKU_2014
EXCEPT
SELECT      SKU, SKU_Description, Department
FROM        CATALOG_SKU_2015;
```

	SKU	SKU_Description	Department
1	100300	Std. Scuba Tank, Light Blue	Water Sports
2	100400	Std. Scuba Tank, Dark Blue	Water Sports

End of Presentation:
Chapter Two Part Two