Chaper 2 Introduction to Structured Query Language

1.Cape Codd Outdoor Sports

The source of data.

Table	Column	DataType
RRTAIL_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)

2.SQL Background

1. Create SQL was developed by the IBM Corporation in the late 1970s. It was endorsed as a national standard by the American National Standards Institute (ANSI) in 1986.
2. Three categories
    - Data definition language (DDL) statements, which are used for creating tables, relationships, and other structures
    - Data manipulation language (DML) statements, which are used for querying, inserting, modifying, and deleting data

3.The SQL SELECT/FROM/WHERE Framework

1. Framework
    - The SQL SELECT clause specifies which columns are to be listed in the query results
    - The SQL FROM clause specifies which tables are to be used in the query
    - The SQL WHERE clause specifies which rows are to be listed in the query results
2. Reading Specified Columns from a Single Table
    1. Sample examples

```
SELECT Department,Buyer FROM SKU_DATA;
```

> Notice that SQL statements terminate with a semicolon (;) character. Allow you to omit the semicolon

    2. Using DISTINCT Eliminate duplicates

```
SELECT DISTINCT Department,Buyer FROM SKU_DATA;
```

> The reason that SQL does not automatically eliminate duplicate rows is that it can be very time consuming to do so.

    3. View all of the columns

```
SELECT * FROM SKU_DATA;
```

3. Reading Specified Rows from a Single Table WHERE

```
SELECT * FROM SKU_DATA WHERE Department='Water Sports';
```

> If the column contains text or date data, the comparison values must be enclosed in single quotation marks ('{text or date data}'). If the column contains numeric data, however, the comparison values need not be in quotes.

4. Reading Specified Columns and Rows from a Single Table

```
SELECT SKU_Description,Department FROM SKU_DATA WHERE
Department='Climbing';
```

4.Submitting SQL Statements to the DBMS

1. Using SQL in Microsoft Access 2007

- “Does Not Work with Microsoft Access ANSI-89 SQL”
- Access Options
- Object Designers
- SQL Server Compatible Syntax

2. Using SQL in Microsoft SQL Server 2008
3. Using SQL in Oracle Database 11g
4. Using SQL in Oracle MySQL 5.5

5.SQL Enhancements for Querying a Single Table

1. Sorting the SQL Query Results

```
SELECT * FROM ORDER_ITEM ORDER BY OrderNumber;
```

Two columns

```
SELECT * FROM ORDER_ITEM ORDER BY OrderNumber,Price;
```

> By default, rows are sorted in ascending order. Descending order

```
SELECT * FROM ORDER_ITEM ORDER BY Price DESC, OrderNumber ASC;
```

2. SQL WHERE Clause Options
    1. Compound WHERE Clauses SQL AND, OR, IN, NOT IN

```
SELECT * FROM SKU_DATA WHERE Department='Water Sports' AND Buyer='Nancy Meyers';
SELECT * FROM SKU_DATA WHERE Department='Camping' OR Department='Climbing';
SELECT * FROM SKU_DATA WHERE Buyer IN ('Nancy Meyers','Cindy Lo','Jerry Martin');
SELECT * FROM SKU_DATA WHERE Buyer NOT IN ('Nancy Meyers','Cindy Lo', 'Jerry Martin');
```

    2. Ranges in SQL WHERE Clauses SQL BETWEEN

```
SELECT * FROM ORDER_ITEM WHERE ExtendedPrice BETWEEN 100 AND 200;
SELECT * FROM ORDER_ITEM WHERE ExtendedPrice >= 100 AND ExtendedPrice <= 200;
```

    3. Wildcards in SQL WHERE Clauses SQL LIKE

```
SELECT * FROM SKU_DATA WHERE Buyer LIKE 'Pete%';
SELECT * FROM SKU_DATA WHERE SKU LIKE '%2__';
```

4. Combining the SQL WHERE Clause and the SQL ORDER BY Clause

```
SELECT * FROM ORDER_ITEM WHERE ExtendedPrice BETWEEN 100 AND 200 ORDER BY
OrderNumber DESC;
```

6.Performing Calculations in SQL Queries

1. Using SQL Built-in Functions SUM, AVG, MIN, MAX, COUNT

```
SELECT SUM(OrderTotal) AS OrderSum FROM RETAIL_ORDER;
SELECT SUM(ExtendedPrice) AS  OrderItemSum,
 AVG(ExtendedPrice)  AS  OrderItemAvg,
 MIN(ExtendedPrice)  AS  OrderItemMin,
 MAX(ExtendedPrice)  AS  OrderItemMax
FROM ORDER_ITEM;
SELECT COUNT(*) AS NumberOfRows FROM ORDER_ITEM;
SELECT COUNT(Department) AS DeptCount FROM SKU_DATA;
SELECT COUNT(DISTINCT Department) AS DeptCount FROM SKU_DATA;
```

> 1. You should be aware of two limitations to SQL built-in functions.
>
> ```
> #error
> SELECT Department, COUNT(*) FROM SKU_DATA;
> ```

> 2. The second problem with the SQL built-in functions that you should understand is that you cannot use them in an SQL WHERE clause
>
> ```
> SELECT * FROM RETAIL_ORDER WHERE OrderTotal > AVG(OrderTotal);
> ```

2. SQL Expressions in SQL SELECT Statements

```
SELECT Quantity * Price AS EP FROM ORDER_ITEM;
```

Another use for SQL expressions in SQL statements is to perform string manipulation.

```
SELECT Buyer+' in '+Department AS Sponsor FROM SKU_DATA;
#Eliminate these extra spaces
SELECT DISTINCT RTRIM(Buyer)+' in '+RTRIM(Department) AS Sponsor ROM SKU_DATA;
```

7.Grouping in SQL SELECT Statements

1. Example

```
SELECT Department,COUNT(*) AS Dept_SKU_Count
FROM SKU_DATA
GROUP BY Department;

SELECT SKU,AVG(ExtendedPrice) AS AvgEP
FROM ORDER_ITEM
GROUP BY SKU;

#Using more than one column
SELECT Department,Buyer,COUNT(*) AS Dept_Buyer_SKU_Count
FROM SKU_DATA
GROUP BY Department,Buyer;
```

> When using the GROUP BY clause, only the column or columns in the GROUP BY expression and the SQL built-in functions can be used in the expressions in the SELECT clause.

```
SELECT SKU,Department,COUNT(*) AS Dept_SKU_Count
FROM SKU_DATA
GROUP BY Department;
```

> In general, to be safe, always place the WHERE clause before the GROUP BY clause

2. Having The SQL HAVING clause restricts the groups that are presented in the result.

```
SELECT  Department,COUNT(*) AS Dept_SKU_Count
FROM SKU_DATA
WHERE SKU  <>  302000
GROUP BY Department
HAVING COUNT  (*)  >  1
ORDER BY  Dept_SKU_Count;
```

8.Looking for Patterns in NASDAQ Trading

1. Investigating the Characteristics of the Data
2. Searching for Patterns in Trading by Day of Week

9.Querying Two or More Tables with SQL

SQL provides two different techniques for querying data from multiple tables: subqueries and joins.

1. Querying Multiple Tables with Subqueries

```
SELECT SUM(ExtendedPrice) AS Revenue
FROM ORDER_ITEM
WHERE SKU  IN
(SELECT SKU
FROM SKU_DATA
WHERE Department='Water Sports');

SELECT  Buyer
FROM    SKU_DATA
WHERE   SKU  IN
(SELECT SKU
FROM    ORDER_ITEM
WHERE   OrderNumber  IN
(SELECT OrderNumber
FROM    RETAIL_ORDER
WHERE   OrderMonth='January'
AND OrderYear=2011));
#**you can use group by or order by**
```

2. Querying Multiple Tables with Joins

- The SQL join operator is used to combine two or more tables by concatenating (sticking together) the rows of one table with the rows of another table.

1. Equijoin or inner join

- Join:The table that is formed by concatenating two tables is called a join.
- The process of creating such a table is called joining the two tables, and the associated operation is called a join operation.
- And, you can use group by or order by.

- We can extend this syntax to join three or more tables.

```
SELECT * FROM RETAIL_ORDER,ORDER_ITEM;

SELECT   * FROM  RETAIL_ORDER,  ORDER_ITEM
WHERE RETAIL_ORDER.OrderNumber = ORDER_ITEM.OrderNumber;
```

3. Comparing Subqueries and Joins
    - As mentioned earlier, a subquery can only be used to retrieve data from the top table.
    - A join can be used to obtain data from any number of tables. Thus, a join can do everything a subquery can do, and more.
4. Summary
    - Structured Query Language (SQL) was developed by IBM and has been endorsed by the ANSI SQL-92 and following standards. SQL is a data sublanguage that can be embedded into full programming languages or submitted directly to the DBMS.
    - We are primarily interested in three categories of SQL statements: DML, DDL, and SQL/PSM statements.
    - The basic structure of an SQL query statement is SELECT/FROM/WHERE. The columns to be selected are listed after SELECT, the table(s) to process is listed after FROM, and any restrictions on data values are listed after WHERE. In a WHERE clause, character and date data valuesmust be enclosed in single quotes. Numeric data need not be enclosed in quotes.
    - By default, the WHERE clause is applied before the HAVING clause

Keyword	keyword
SELECT	DISTINCT
FROM	DESC
WHERE	ASC
ORDER BY	AND
GROUP BY	OR
HAVING	IN
NOT IN	BETWEEN
LIKE	%
SUM	AVG
MIN	MAX
COUNT	AS