Introduction to Database Systems(Lab)

Engr. Madeha Mushtaq
Department of Computer Science
Iqra National University

SQL NULL Values

- A field with a NULL value is a field with no value.
- If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field.
- Then, the field will be saved with a NULL value.
- It is not possible to test for NULL values with comparison operators, such as =, <, or <>.
- We will have to use the IS NULL and IS NOT NULL operators instead.

SQL NULL Values

- **IS NULL Syntax:**
- SELECT *column_names*
  FROM *table_name*
  WHERE *column_name* IS NULL;
- **IS NOT NULL Syntax:**
- SELECT *column_names*
  FROM *table_name*
  WHERE *column_name* IS NOT NULL;

SQL NULL Values

- **IS NULL Example:**
- SELECT CustomerName, ContactName, Address
  FROM Customers
  WHERE Address IS NULL;
- **IS NOT NULL Example:**
- SELECT CustomerName, ContactName, Address
  FROM Customers
  WHERE Address IS NOT NULL;

SQL UPDATE Statement

- The UPDATE statement is used to modify the existing records in a table.
- **UPDATE Syntax:**
- UPDATE *table_name*
  SET *column1 = value1*, *column2 = value2*, ...
  WHERE *condition*;
- **UPDATE Example:**
- The following SQL statement updates the first customer (CustomerID = 1) with a new contact person *and* a new city.
  - UPDATE Customers
    SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
    WHERE CustomerID = 1;

SQL UPDATE Statement

- **UPDATE Multiple Records:**
- It is the WHERE clause that determines how many records that will be updated.
- The following SQL statement will update the contactname to "Juan" for all records where country is "Mexico":
  - UPDATE Customers
    SET ContactName='Juan'
    WHERE Country='Mexico';

SQL UPDATE Statement

- Example:
- UPDATE Customers
  SET ContactName='Juan';
- Be careful when updating records. If you omit the WHERE clause, ALL records will be updated.

SQL DELETE Statement

- The DELETE statement is used to delete existing records in a table.
- **DELETE Syntax:**
- DELETE FROM *table_name* WHERE *condition*;
- **SQL DELETE Example:**
- The following SQL statement deletes the customer "Alfreds Futterkiste" from the "Customers" table:
  - DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

SQL DELETE Statement

- **Delete All Records:**
- It is possible to delete all rows in a table without deleting the table. This means that the table structure, attributes, and indexes will be intact:
- DELETE FROM *table_name*;
- The following SQL statement deletes all rows in the "Customers" table, without deleting the table:
  - DELETE FROM Customers;

SQL MIN() and MAX() Functions

- The MIN() function returns the smallest value of the selected column.
- The MAX() function returns the largest value of the selected column.
- **MIN() Syntax:**
- SELECT MIN(*column_name*)
  FROM *table_name*
  WHERE *condition*;
- **MAX() Syntax:**
- SELECT MAX(*column_name*)
  FROM *table_name*
  WHERE *condition*;

SQL MIN() and MAX() Functions

- **MIN() Example:**
- The following SQL statement finds the price of the cheapest product:
  - SELECT MIN(Price) AS SmallestPrice
    FROM Products;
- **MAX() Example:**
- The following SQL statement finds the price of the most expensive product:
  - SELECT MAX(Price) AS LargestPrice
    FROM Products;

SQL COUNT(), AVG() and SUM() Functions

- The COUNT() function returns the number of rows that matches a specified criteria.
- The AVG() function returns the average value of a numeric column.
- The SUM() function returns the total sum of a numeric column.
- **COUNT() Syntax:**
- SELECT COUNT(*column_name*)
  FROM *table_name*
  WHERE *condition*;

SQL COUNT(), AVG() and SUM() Functions

- **AVG() Syntax:**
- SELECT AVG(*column_name*)
  FROM *table_name*
  WHERE *condition*;
- **SUM() Syntax:**
- SELECT SUM(*column_name*)
  FROM *table_name*
  WHERE *condition*;
- **COUNT() Example:**
- The following SQL statement finds the number of products:
  - SELECT COUNT(ProductID)
    FROM Products;

SQL COUNT(), AVG() and SUM() Functions

- **AVG() Example:**
- The following SQL statement finds the average price of all products:
  - SELECT AVG(Price)
    FROM Products;
- **SUM() Example:**
- The following SQL statement finds the sum of the "Quantity" fields in the "OrderDetails" table:
  - SELECT SUM(Quantity)
    FROM OrderDetails;

SQL LIKE Operator

- The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.
- There are two wildcards often used in conjunction with the LIKE operator:
- % - The percent sign represents zero, one, or multiple characters
- _ - The underscore represents a single character.
- **LIKE Syntax:**
- SELECT *column1, column2, ...*
  FROM *table_name*
  WHERE *columnN* LIKE *pattern*;

SQL LIKE Operator

- Here are some examples showing different LIKE operators with '%' and '_' wildcards:

LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Finds any values that start with "a"
WHERE CustomerName LIKE '%a'	Finds any values that end with "a"
WHERE CustomerName LIKE '%or%'	Finds any values that have "or" in any position
WHERE CustomerName LIKE '_r%'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a_%_%'	Finds any values that start with "a" and are at least 3 characters in length
WHERE ContactName LIKE 'a%o'	Finds any values that start with "a" and ends with "o"

SQL LIKE Operator

- **SQL LIKE Examples:**
- The following SQL statement selects all customers with a CustomerName starting with "a":
  - SELECT * FROM Customers
    WHERE CustomerName LIKE 'a%';
- The following SQL statement selects all customers with a CustomerName ending with "a":
  - SELECT * FROM Customers
    WHERE CustomerName LIKE '%a';

SQL LIKE Operator

- The following SQL statement selects all customers with a CustomerName that have "or" in any position:
  - SELECT * FROM Customers
    WHERE CustomerName LIKE '%or%';
- The following SQL statement selects all customers with a CustomerName that have "r" in the second position:
  - SELECT * FROM Customers
    WHERE CustomerName LIKE '_r%';

SQL LIKE Operator

- The following SQL statement selects all customers with a CustomerName that starts with "a" and are at least 3 characters in length:
  - SELECT * FROM Customers
    WHERE CustomerName LIKE 'a_%_%';
- The following SQL statement selects all customers with a ContactName that starts with "a" and ends with "o":
  - SELECT * FROM Customers
    WHERE ContactName LIKE 'a%o';
- The following SQL statement selects all customers with a CustomerName that does NOT start with "a":
  - SELECT * FROM Customers
    WHERE CustomerName NOT LIKE 'a%';

End of Slides