**ЛАБОРАТОРНА РОБОТА №6**

**ПРЕДИКАТИ. ОПЕРАТОРИ ПОРІВНЯННЯ. СТРОКОВІ ФУНКЦІЇ**

Мета: ознайомлення студентів з предикатами, операторами порівняння та строковими функціями в SQL, навчання основам застосування цих інструментів для фільтрації та аналізу даних в реляційних базах даних. Розробка навичок використання предикатів і операторів порівняння для виконання ефективних запитів до бази даних, що дозволить студентам здійснювати точний вибір даних відповідно до заданих умов.

**ТЕОРЕТИЧНІ ОСНОВИ**

У базах даних, предикати та оператори порівняння є фундаментальними інструментами для визначення умов у запитах SQL, які дозволяють фільтрувати, оновлювати або видаляти дані згідно з заданими критеріями. Вони відіграють ключову роль у створенні ефективних і точних запитів до бази даних.

***Оператори порівняння***

Оператори порівняння дозволяють порівнювати два значення або вирази. Стандартні оператори порівняння включають:

- `=` (рівно)

- `>` (більше ніж)

- `<` (менше ніж)

- `>=` (більше ніж або дорівнює)

- `<=` (менше ніж або дорівнює)

- `<>` або `!=` (не дорівнює)

Ці оператори можна застосовувати для порівняння числових значень, рядків, дат та інших типів даних.

***Логічні предикати***

Логічні предикати використовуються для комбінування або інвертування умов порівняння, створюючи складніші умови фільтрації. Основні логічні предикати:

- `AND` - вибирає рядок, якщо обидві умови істинні.

- `OR` - вибирає рядок, якщо хоча б одна з умов істинна.

- `NOT` - інвертує результат умови порівняння.

***Спеціалізовані предикати***

- `BETWEEN` - перевіряє, чи значення лежить в межах вказаного діапазону.

- `IN` - перевіряє, чи значення входить до списку вказаних значень.

- `LIKE` - використовується для пошуку за шаблоном у рядкових даних.

- `IS NULL` - перевіряє, чи значення є NULL.

Ці інструменти можна застосовувати у вибірках (`SELECT`), оновленнях (`UPDATE`), видаленнях (`DELETE`) та інструкціях вставки (`INSERT` при використанні в `SELECT` для визначення значень), дозволяючи виконувати точний вибір та маніпуляцію даними згідно з бізнес-логікою і вимогами безпеки.

Правильне використання предикатів та операторів порівняння має вирішальне значення для оптимізації продуктивності запитів. Неоптимізовані умови можуть призвести до непотрібного сканування великих обсягів даних, що суттєво знижує швидкість виконання запитів і завантажує системні ресурси. Оптимізація запитів з використанням ефективних умов фільтрації та вибору індексованих колонок для порівнянь може значно підвищити продуктивність та забезпечити швидкий доступ до потрібних даних.

Оператори порівняння та логічні предикати є невід'ємною частиною мови SQL, що дозволяє розробникам та аналітикам ефективно маніпулювати даними. Розуміння та вміле використання цих інструментів сприяє створенню оптимізованих, продуктивних та безпечних запитів до баз даних. Водночас, знання принципів оптимізації запитів є критично важливим для підвищення ефективності роботи з реляційними базами даних та забезпечення швидкого доступу до потрібної інформації.

Строкові функції в T-SQL (Transact-SQL) є потужним інструментарієм для маніпуляції та аналізу текстових даних в базах даних Microsoft SQL Server. Вони дозволяють виконувати різноманітні операції з текстовими рядками, включаючи пошук, заміну, порівняння, конкатенацію та вилучення підрядків, а також перевірку формату і вмісту даних.

Основні строкові функції

1. `LEN` (довжина рядка)

- Опис: Повертає кількість символів у заданому рядку, не враховуючи кінцеві пробіли.

- Приклад: `SELECT LEN('Hello World') AS Length;`

- Результат: `11`

2. `SUBSTRING` (вилучення підрядка)

- Опис: Повертає частину рядка, починаючи з вказаної позиції та довжини.

- Приклад: `SELECT SUBSTRING('Hello World', 1, 5) AS ExtractedString;`

- Результат: `Hello`

3. `CHARINDEX` (індекс першого входження підрядка)

- Опис: Повертає позицію першого входження підрядка в рядку.

- Приклад: `SELECT CHARINDEX('World', 'Hello World') AS Position;`

- Результат: `7`

4. `REPLACE` (заміна підрядків)

- Опис: Замінює всі входження певного підрядка на інший рядок у заданому рядку.

- Приклад: `SELECT REPLACE('Hello World', 'World', 'T-SQL') AS ReplacedString;`

- Результат: `Hello T-SQL`

5. `CONCAT` (конкатенація рядків)

- Опис: Об'єднує два або більше рядків в один.

- Приклад: `SELECT CONCAT('Hello ', 'World') AS ConcatenatedString;`

- Результат: `Hello World`

6. `LTRIM` та `RTRIM` (видалення пробілів)

- Опис: Видаляють початкові (`LTRIM`) або кінцеві (`RTRIM`) пробіли з рядка.

- Приклад: `SELECT LTRIM(' Hello'), RTRIM('World ') AS TrimmedString;`

- Результат: `Hello` і `World`

7. `UPPER` та `LOWER` (зміна регістру)

- Опис: Конвертує всі символи рядка в верхній (`UPPER`) або нижній (`LOWER`) регістри.

- Приклад: `SELECT UPPER('Hello World') AS UpperCase, LOWER('Hello World') AS LowerCase;`

- Результат: `HELLO WORLD` і `hello world`

8. `PATINDEX` (пошук за шаблоном)

- Опис: Повертає позицію першого входження патерну, що заданий як регулярний вираз.

- Приклад: `SELECT PATINDEX('%World%', 'Hello World') AS PatternPosition;`

- Результат: `7`

Строкові функції відіграють ключову роль у роботі з даними, дозволяючи розробникам та аналітикам баз даних виконувати складні операції обробки тексту, необхідні для очищення даних, генерації звітів, імпорту та експорту даних, а також у багатьох інших задачах, пов'язаних з обробкою і аналізом інформації. Знання та вміння ефективно використовувати ці функції є невід'ємною частиною набору навичок будь-якого фахівця, що працює з SQL.

**ЗАВДАННЯ**

**Усі завдання необхідно виконувати на учбовій базі даних WebStore**

***Задача 1***

Напишіть запит, який поверне список офісів східного регіону з ціллю по продажах менше або рівної 350000.00.

- Використовується таблиця [dbo].[OFFICES]

- Результуючий набір даних містить: Ідентифікатор офісу, місто, ідентифікатор керівника офісу

***Задача 2.1***

Напишіть запит, що повертає список замовлень, які були проведені не в 2008 році.

Враховуйте тільки замовлення на товари з ідентифікатором товарів (PRODUCT), які містять «A» другим символом або «0» в будь-якому місці.

- Використовується таблиця [dbo].[ORDERS]

- Задійте предикат LIKE

- Задійте предикат NOT BETWEEN

- Результуючий набір даних містить усі стовпці

***Задача 2.2***

Напишіть запит, що повертає загальну суму проведених замовлень по ідент. виробника товарів (MFR),

які були проведені не в 2008 році. Враховуйте тільки замовлення на товари з ідентифікатором товарів (PRODUCT), які містять

A другим символом або 0 в будь-якому місці.

- Використовується таблиця [dbo].[ORDERS]

- Задійте предикат LIKE

- Задійте предикат NOT BETWEEN

- Задійте агрегатну функцію SUM

- Результуючий набір даних містить: Ідент. виробника товарів, кіль-ть унікальних замовлень

- Відсортуйте результат за загальною сумою (за зростанням)

***Задача 2.3***

Напишіть запит, що повертає ідент. виробника з найбільшою загальною сумою проведених замовлень.

Враховуйте ймовірність того, що одразу кілька виробників можуть мати одну і ту ж загальну суму.

Враховуйте тільки замовлення, які були проведені не в 2008 році.

Враховуйте тільки замовлення на товари з ідентифікатором товарів (PRODUCT), які містять

A другим символом або 0 в будь-якому місці.

- Використовується таблиця [dbo].[ORDERS]

- Задійте предикат LIKE

- Задійте предикат NOT BETWEEN

- Задійте агрегатну функцію SUM

- Задійте фільтр TOP і оператор WITH TIES

- Результуючий набір даних містить: Ідент. виробника товарів, загальна сума

***Задача 3***

Напишіть запит, який поверне ідент. офісу з найбільшою кількістю працівників.

Враховуйте ймовірність того, що відразу кілька офісів можуть мати одну й ту саму кількість працівників.

Враховуйте лише працівників на посаді Sales Rep та у віці 29, 45, 48.

- Використовується таблиця [dbo].[SALESREPS]

- Задіяти предикат IN

- Задіяйте агрегатну функцію COUNT

- Задійте фільтр TOP та оператор WITH TIES

- Результуючий набір даних містить: Ідент. офісу, кількість працівників

***Задача 4***

Напишіть запит, який поверне дані продукту з найбільшою довжиною (за кількістю символів) опису.

Враховуйте ймовірність того, що одразу кілька товарів можуть мати однакову кількість символів в описі.

- Використовується таблиця [dbo].[PRODUCTS]

- Застосуйте строкову функцію `LEN`

- Результируючий набір даних містить: Ідент. виробника, ідент. продукту, опис продукту, довжину опису.

***Задача 5***

Напишіть запит, що повертає дані продуктів, ідентифікатор яких містить лише цифри.

Склейте значення стовпців (конкатенація): MFR\_ID, PRODUCT\_ID, [DESCRIPTION]. Врахуйте, що між значеннями стовпців має бути лише один пробіл. При необхідності видаліть хвостові пробіли.

Порахуйте кількість символів у сконкатенованому рядку. Видаліть дублюючі рядки у результуючому наборі даних.

- Використовується таблиця [dbo].[PRODUCTS]

- Застосуйте предикат `NOT LIKE`

- Застосуйте строкові функції: `CONCAT\_WS`, `UPPER`, `LEN`, `RTRIM`

- Результируючий набір даних містить: сконкатенований рядок, довжина сконкатенованого рядка

- Відсортуйте результируючий набір даних за довжиною сконкатенованого рядка (за зменшенням)

***Задача 6***

Напишіть запит, який поверне список клієнтів з найменшим кредитним лімітом.

Враховуйте ймовірність того, що одразу кілька клієнтів можуть мати однаковий кредитний ліміт.

Результируючий набір даних містить: ідент. клієнта, найменування клієнта, кредитний ліміт.

Найменування клієнта необхідно відобразити як 2 букви на початку рядка і 2 букви в кінці рядка, всі інші букви замінити на символ \*.

- Використовується таблиця [dbo].[CUSTOMERS]

**КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Що таке оператор порівняння в SQL? Наведіть приклади операторів порівняння.

2. Як використовуються логічні оператори `AND`, `OR`, та `NOT` у SQL-запитах? Дайте приклади їх застосування.

3. Опишіть, як використовується предикат `LIKE` для пошуку за шаблоном. Як можна використати символи `%` та `\_` у шаблонах?\*\*

4. Що таке предикат `IN` і як він використовується в SQL? Наведіть приклад запиту з використанням `IN`.

5. Як використовується предикат `BETWEEN`? Чи включаються крайні значення діапазону, заданого в `BETWEEN`?

6. Опишіть, як застосовується оператор `IS NULL` у SQL-запитах. Чому не можна використовувати оператори `=` або `<>` для перевірки на `NULL`?

7. Яка різниця між `=` та `IS` при порівнянні значень з `NULL` у SQL?

8. Як можна використати предикат `NOT` для інвертування умови? Наведіть приклад.

9. Поясніть, як використовується `NOT BETWEEN` та які дані він вибирає.

10. Які стратегії можна використати для оптимізації запитів із складними умовами фільтрації?

11. Опишіть, як `TOP` і `WITH TIES` можуть бути використані разом. Які типи сценаріїв це дозволяє обробляти?

12. Як можна використовувати оператори порівняння та предикати для роботи з датами в SQL? Наведіть приклад.

13. Що таке строкові функції в T-SQL і для чого вони використовуються?

- Обговоріть призначення строкових функцій і дайте кілька прикладів їхнього застосування.

14. Як використовується функція `LEN` і чому вона може бути корисною?

- Дайте приклад використання `LEN` для визначення довжини рядка.

15. Опишіть, як застосовується функція `SUBSTRING`. Наведіть приклад.

- Як можна використовувати `SUBSTRING` для витягу певної частини рядка?

16. Чим відрізняються функції `CHARINDEX` та `PATINDEX`?

- Наведіть приклади використання обох функцій.

17. Як використовується функція `REPLACE` для заміни тексту у рядку?

- Приведіть приклад, де функція `REPLACE` може бути застосована для коригування даних.

18. Опишіть мету і використання функцій `LTRIM` та `RTRIM`.

- Чому важливо видаляти ведучі та завершальні пробіли у рядках даних?

19. Як функції `UPPER` та `LOWER` можуть бути застосовані у роботі з текстовими даними?

- Дайте приклад, коли нормалізація регістру рядка є корисною.

20. Чому важливо знати про функцію `CONCAT`? Які переваги вона пропонує порівняно з оператором `+`?

- Приведіть приклад використання `CONCAT` для об'єднання декількох рядків.

21. Як можна використовувати функцію `FORMAT` для форматування дат та чисел у T-SQL?

- Наведіть приклад, де `FORMAT` може бути особливо корисним.

22. Обговоріть, як валідація даних з використанням строкових функцій може вплинути на якість та надійність даних у базі даних.

- Чому важливо перевіряти та валідувати текстові дані перед їх обробкою або збереженням?

**ДОДАТКОВІ МАТЕРІАЛИ:**

<https://www.w3schools.com/sql/sql_where.asp>

<https://learn.microsoft.com/en-us/sql/t-sql/language-elements/comparison-operators-transact-sql?view=sql-server-ver16>

<https://learn.microsoft.com/en-us/sql/t-sql/queries/predicates?view=sql-server-ver16>

<https://learn.microsoft.com/en-us/sql/t-sql/functions/string-functions-transact-sql?view=sql-server-ver16>