LOGOS
IT ACADEMY

MySQL

Заняття #3

LOGOS
IT ACADEMY

- Оновлення даних
- Видалення даних
- Оператори фільтрації: in, between,
- Оператори like, regexp, is null

Інструкція зміни даних (UPDATE)

Інструкція **UPDATE** змінює значення стовпців.

Щоб змінити значення в одному стовпці таблиці, використовується синтаксис:

```
UPDATE <ім'я таблиці> SET <ім'я стовпця> = <значення>;
```

Приклад. Замінити рейтинг усіх замовників на 200:

```
UPDATE Customers SET rating = 200;
```

Якщо потрібно змінити значення стовпця у певному рядку, то використовується фраза **WHERE**.

Приклад. Змінити рейтинг на 200 усіх замовників для продавця James:

```
UPDATE Customers SET rating = 200 WHERE id = 1001;
```

Тут вказано поле **id**, а не поле **name**, оскільки використання первинних ключів надійно забезпечує видалення лише одного рядка.

LOGOS IT ACADEMY

Правила використання фрази SET

В одній команді можна модифікувати лише одну таблицю.

У фразі **SET** можна використовувати вирази.

Приклад. Продавцям у Лондоні збільшити їх комісійні у два рази:

```
UPDATE Sellers SET comm = comm * 2
WHERE city = 'London';
```

Вираз **SET** не є предикатом. Тому треба вводити **NULL** значення без синтаксису предикату.

Приклад. Встановити рейтинг замовників у Лондоні в NULL:

```
UPDATE Customers SET rating = NULL
WHERE city = 'London';
```

Інструкція видалення записів (DELETE)

Інструкція видаляє цілі рядки таблиці, а не індивідуальні значення полів.

Наприклад, щоб видалити все вмістиме таблиці Sellers, можна використати команду:

```
DELETE FROM Sellers;
```

Якщо необхідно видалити певні рядки з таблиці, використовують атрибут **WHERE**.

Наприклад, щоб видалити продавця Axelrod, можна ввести:

```
DELETE FROM Sellers WHERE id = 1003;
```

Тут вказано поле **id**, а не поле **name**, оскільки використання первинних ключів надійно забезпечує видалення лише одного рядка.

Примітка: Після видалення усіх рядків таблиці необхідно «обнуляти» автоінкементоване значення первинного ключа:

```
ALTER TABLE <ім'я таблиці> AUTO_INCREMENT = 1
```

Усунення надлишковості вибраних даних

Ключове слово DISTINCT (ВІДМІННІСТЬ) усуває повторювані значення з команди **SELECT**:

```
SELECT DISTINCT стовпець_1, … FROM таблиця;
```

DISTINCT слідкує за тим, які значення стовпець_1 були раніше, щоб вони не дублювались у результатній таблиці.

Приклад. Виведемо всіх виробників:

```
SELECT DISTINCT Manufacturer FROM Products;
```

Також ми можемо задавати вибірку унікальних значень за кількома стовпцями:

```
SELECT DISTINCT Manufacturer, ProductCount FROM Products;
```

LOGOS IT ACADEMY

Оператор IN

Оператор **IN** визначає набір значень, які повинні мати стовпці:

```
WHERE вираз [NOT] IN (вираз)
```

Вираз в дужках після **IN** визначає набір значень. Цей набір може обчислюватися динамічно на підставі, наприклад, ще одного запиту, або це можуть бути константні значення.

Наприклад, виберемо товари, у яких виробник або Samsung, або Xiaomi, або Huawei:

```
SELECT * FROM Products
WHERE Manufacturer IN ('Samsung', 'HTC', 'Huawei');
```

Оператор **NOT**, навпаки, дозволяє вибрати всі рядки, стовпці яких не мають певних значень:

```
SELECT * FROM Products
WHERE Manufacturer NOT IN ('Samsung', 'HTC', 'Huawei');
```

Альтернативою є поєднання предикатів порівняння з логічною операцією **OR**.

Оператор BETWEEN

Оператор **BETWEEN** визначає діапазон значень за допомогою початкового і кінцевого значення, яким має відповідати вираз:

```
WHERE вираз [NOT] BETWEEN початкове_значення AND кінцеве_значення
```

Наприклад, отримаємо усі товари, у яких ціна від 20 000 до 50 000 (початкове і кінцеве значення також включаються в діапазон):

```
SELECT * FROM Products WHERE Price BETWEEN 20000 AND 50000;
```

Якщо треба, навпаки, вибрати ті рядки, які не потрапляють в даний діапазон, то додається оператор **NOT**:

```
SELECT * FROM Products WHERE Price NOT BETWEEN 20000 AND 50000;
```

Приклад складніших виразів, наприклад, отримаємо товари за сукупною вартістю (ціна * кількість):

```
SELECT * FROM Products WHERE Price * ProductCount BETWEEN 90000 AND 150000;
```

На відміну від оператора **IN**, оператор **BETWEEN** є чутливим до порядку, тобто першим має бути менше значення (як символьне так і числове).

Оператор LIKE

Оператор **LIKE** приймає шаблон рядки, якому має відповідати вираз.

```
WHERE вираз [NOT] LIKE шаблон_рядка
```

Для визначення шаблону можуть застосовуватися ряд спеціальних символів підстановки:

- **%** - Відповідає будь-підрядку, яка може мати будь-яку кількість символів, при цьому подстрока може і не містити жодного символу

Наприклад, вираз **WHERE ProductName LIKE 'Galaxy%'** відповідає таким значенням як "Galaxy Ace 2" або "Galaxy S7"

- **_** - Відповідає будь-якому одиночному символу

Наприклад, вираз **WHERE ProductName LIKE 'Galaxy S_'** відповідає таким значенням як "Galaxy S7" або "Galaxy S8".

LOGOS IT ACADEMY

Оператор REGEXP

REGEXP дозволяє задати регулярний вираз, яким має відповідати значення стовпця. В цьому плані REGEXP представляє більш витончений і комплексний спосіб фільтрації, ніж оператор LIKE. REGEXP має схожий синтаксис:

WHERE вираз [NOT] REGEXP регулярний вираз

Регулярний вираз може приймати такі спеціальні символи:

- **^** - Вказує на початок рядка
- **$** - Вказує на кінець рядка
- **.** - Відповідає будь-якому одиночному символу
- **[Символи]** - відповідає будь-якому одиночному символу з дужок
- **[Начальний_сімвол-конечний_сімвол]** - відповідає будь-якому одиночному символу з діапазону символів
- **|** - Відокремлює два шаблони рядки, і значення має відповідати одну з цих шаблонів

LOGOS IT ACADEMY

Приклади REGEXP

Рядок повинен містити "Phone", наприклад, iPhone X, Nokia Phone N, iPhone:

```
WHERE ProductName REGEXP 'Phone';
```

Рядок повинен починатися з "Phone", наприклад, Phone 34, PhoneX, але не XPhone:

```
WHERE ProductName REGEXP '^Phone';
```

Рядок повинен закінчуватися на "Phone", наприклад, iPhone, Nokia Phone, але не PhoneX:

```
WHERE ProductName REGEXP 'Phone$';
```

Рядок повинен містити або iPhone 7, або iPhone 8:

```
WHERE ProductName REGEXP 'iPhone [78]';
```

Рядок повинен містити або iPhone 6, або iPhone 7, або iPhone 8:

```
WHERE ProductName REGEXP 'iPhone [6-8]';
```

Оператор IS NULL

Оператор **IS NULL** дозволяє вибрати всі рядки, стовпці яких мають значення **NULL**:

```
SELECT * FROM Products WHERE ProductCount IS NULL;
```

За допомогою додавання оператора **NOT** можна, наоборот, вибрати рядки, стовпчики яких не мають значення **NULL**:

```
SELECT * FROM Products WHERE ProductCount IS NOT NULL;
```

Додаткові джерела інформації:

Посилання

https://www.w3schools.com/sql/sql_update.asp
https://www.w3schools.com/sql/sql_delete.asp
https://www.w3schools.com/sql/sql_distinct.asp
https://www.w3schools.com/sql/sql_in.asp
https://www.w3schools.com/sql/sql_between.asp
https://www.w3schools.com/sql/sql_like.asp
https://www.w3schools.com/sql/sql_null_values.asp