**ЛАБОРАТОРНА РОБОТА №7**

**ТАБЛИЧНИЙ ОПЕРАТОР JOIN**

Мета: основна мета цієї лабораторної роботи полягає у вивченні та практичному застосуванні табличного оператора JOIN в SQL. Лабораторна робота охоплює наступні ключові аспекти:

1. Основи роботи з JOIN: Ознайомлення з концепцією табличного оператора JOIN та його важливістю в SQL для з'єднання таблиць за певними критеріями.

2. Види JOIN та їх застосування: Дослідження різних видів JOIN, включаючи CROSS JOIN, INNER JOIN, LEFT JOIN, RIGHT JOIN та FULL JOIN. Розуміння їх відмінностей та сценаріїв використання.

3. Практична робота з даними: виконання ряду завдань, що включають використання різних типів JOIN для з'єднання таблиць та аналізу даних. Це надасть студентам можливість поглиблено ознайомитися з принципами роботи JOIN.

4. Розвиток навичок SQL-запитів: поліпшення навичок студентів у формулюванні SQL-запитів, що включають JOIN, для вирішення практичних завдань з обробки та аналізу даних.

5. Аналіз реальних сценаріїв використання: застосування знань та навичок на практичних прикладах та реальних сценаріях використання, що допоможе студентам зрозуміти важливість і потенціал застосування JOIN в роботі з базами даних.

Ця лабораторна робота покликана не тільки надати студентам теоретичні знання, але й розвинути практичні навички роботи з однією з найбільш фундаментальних і потужних функцій у SQL, що є критично важливим для будь-якого спеціаліста в області обробки даних.

**ТЕОРЕТИЧНІ ОСНОВИ**

Види з'єднань (рис.1):

Декартовий добуток рядків

CROSS JOIN

Внутрішнє з'єднання

INNER JOIN

Зовнішні з'єднання

LEFT OUTER JOIN

RIGHT OUTER JOIN

FULL OUTER JOIN

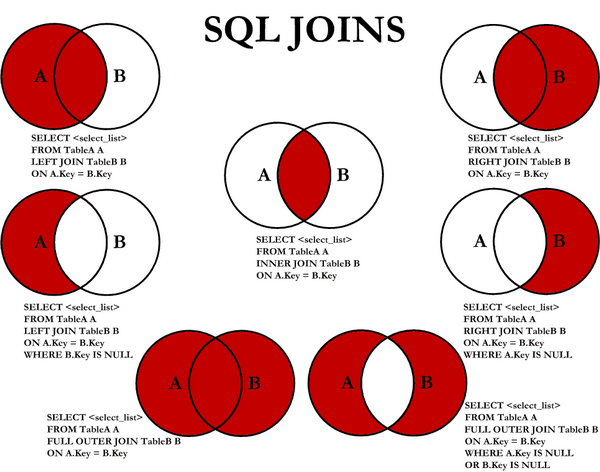


Рис. 1 Види з’єднань в SQL

Табличний оператор JOIN та його фази

CROSS JOIN -> INNER JOIN -> LEFT OUTER JOIN / RIGHT OUTER JOIN / FULL OUTER JOIN

\*\*Теоретичні Основи: Табличний Оператор JOIN та Його Фази\*\*

Табличний оператор JOIN у SQL використовується для об'єднання рядків з двох або більше таблиць на основі спільних полів між ними. Цей процес можна розглядати як послідовність фаз, кожна з яких відіграє певну роль у формуванні кінцевого результату запиту.

**1. CROSS JOIN**

- ***Опис:*** CROSS JOIN виконує декартів добуток двох таблиць, тобто поєднує кожен рядок однієї таблиці з кожним рядком іншої таблиці.

- ***Приклад:*** якщо маємо таблиці A і B, де A містить 3 рядки, а B – 2, результатом CROSS JOIN буде 6 рядків, які є всіма можливими комбінаціями рядків з обох таблиць.

- ***Синтаксис:***

SELECT \*

FROM TableA

CROSS JOIN TableB;

**2. INNER JOIN**

- ***Опис:*** INNER JOIN вибирає рядки з обох таблиць, які відповідають заданій умові. Якщо умова не виконується, рядки не включаються в результат.

- ***Приклад:*** об'єднання двох таблиць по спільній колонці, де включаються лише ті рядки, які мають відповідності в обох таблицях.

***- Синтаксис:***

SELECT \*

FROM TableA

INNER JOIN TableB

ON TableA.CommonColumn = TableB.CommonColumn;

**3. LEFT OUTER JOIN / RIGHT OUTER JOIN / FULL OUTER JOIN**

***- Опис:*** ці види JOIN розширюють можливості INNER JOIN, включаючи рядки, які не мають відповідностей в одній з таблиць.

- LEFT OUTER JOIN включає всі рядки з лівої таблиці, навіть якщо відповідностей у правій таблиці не знайдено.

- RIGHT OUTER JOIN включає всі рядки з правої таблиці, навіть якщо відповідностей у лівій таблиці не знайдено.

- FULL OUTER JOIN включає всі рядки з обох таблиць, навіть якщо відповідностей немає ні в одній з таблиць.

***- Приклади:***

- Якщо маємо таблиці A і B, LEFT OUTER JOIN включить всі рядки з A незалежно від того, чи є в них відповідності в B.

***- Синтаксис:***

SELECT \*

FROM TableA

LEFT OUTER JOIN TableB

ON TableA.CommonColumn = TableB.CommonColumn;

SELECT \*

FROM TableA

RIGHT OUTER JOIN TableB

ON TableA.CommonColumn = TableB.CommonColumn;

SELECT \*

FROM TableA

FULL OUTER JOIN TableB

ON TableA.CommonColumn = TableB.CommonColumn;

Розуміння цих основних фаз і принципів роботи JOIN є ключовим для ефективного використання SQL для обробки та аналізу даних, зібраних з різних джерел і збережених у різних таблицях.

**ЗАВДАННЯ**

**Завдання 1**

Напишіть запит, який поверне список унікальних ідентифікаторів виробників товару (MFR).

Враховуйте лише замовлення, які були проведені у 2008 році клієнтами зі словом CORP у назві (COMPANY). Для цього необхідно використовувати оператор LIKE та підстановочні знаки (wildcards).

- Використовуються таблиці: dbo.CUSTOMERS, dbo.ORDERS

- Застосуйте внутрішнє з'єднання (ANSI-92)

- Результуючий набір даних містить: Ідентифікатор виробника товару (без дублікатів)

***Додаткові (необов’язкові) умови завдання:***

Необхідно уникнути неявних перетворень типів даних, тобто зверніть увагу на

типи даних стовпців, які задіяні для фільтрації рядків. Виключіть обчислювані

вирази в умовах пошуку.

**Завдання 2**

Напишіть запит, який у розрізі ідентифікатора клієнта (CUST\_NUM) та місяця проведення замовлення (ORDER\_DATE) поверне кількість унікальних замовлень (для цього використовується GROUP BY). Враховуйте лише замовлення, які були проведені у 2008 році клієнтами зі словом CORP. у назві (COMPANY). У випадку, якщо у клієнта не було жодного замовлення, залиште такого клієнта у результуючому наборі даних.

Відсортуйте результативний набір даних за кількістю проведених замовлень (за спаданням)

- Використовуються таблиці: dbo.CUSTOMERS, dbo.ORDERS

- Застосуйте ліве зовнішнє з'єднання

- Результуючий набір даних містить: ідентифікатор клієнта, місяць проведення замовлення (для отримання місяця з дати замовлення використовуємо функцію MONTH), кількість унікальних замовлень (використовуємо COUNT)

***Додаткові (необов’язкові) умови завдання:***

Постарайтеся уникнути неявних перетворень типів даних, тобто зверніть увагу на типи даних стовпців, які задіяні для фільтрації рядків. Виключіть обчислювані вирази в умовах пошуку.

**Завдання 3**

Напишіть запит, який поверне список (без дублікатів) придбаних товарів.

Враховуйте лише замовлення, які були проведені у 2008 році клієнтами зі словом CORP у назві (COMPANY). У випадку, якщо у клієнта не було жодного замовлення, залиште такого клієнта у результуючому наборі даних.

- Використовуються таблиці: dbo.CUSTOMERS, dbo.ORDERS, dbo.PRODUCTS

- Застосуйте ліве зовнішнє з'єднання

- Результуючий набір даних містить: Ідентифікатор клієнта, назва компанії (у верхньому регістрі. Для цього використовуємо рядкову функцію (String Function) UPPER), опис товару ([DESCRIPTION])

***Додаткові (необов’язкові) умови завдання:***

Постарайтеся уникнути неявних перетворень типів даних, тобто зверніть увагу на типи даних стовпців, які задіяні для фільтрації рядків. Виключіть обчислювані вирази в умовах пошуку.

**КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Що таке табличний оператор JOIN у SQL?

2. Яка основна мета використання INNER JOIN?

3. Які відмінності між LEFT JOIN та RIGHT JOIN?

4. Що таке FULL OUTER JOIN та коли він використовується?

5. Як CROSS JOIN впливає на кількість рядків у результаті запиту?

6. Чому важливо використовувати умови в клаузі ON при виконанні JOIN?

7. Як з'єднати дані з трьох або більше таблиць за допомогою оператора JOIN?

8. У чому різниця між використанням WHERE та ON у контексті JOIN?

9. Що відбудеться, якщо при JOIN не вказати жодної умови з'єднання?

10. Як обробляються NULL значення в різних типах JOIN?

11. Як використати JOIN для фільтрації даних з однієї таблиці на основі даних іншої таблиці?

12. Як можна використовувати агрегатні функції разом з JOIN?

13. Як впорядкувати результати, отримані за допомогою JOIN?

**ДОДАТКОВІ МАТЕРІАЛИ:**

**JOIN**

https://www.w3schools.com/sql/sql\_join.asp

**LIKE**

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

**Підстановочні знаки (wildcards) у SQL**

<https://acode.com.ua/wildcards-sql/>

https://www.w3schools.com/sql/sql\_wildcards.asp

# Date and time data types and functions

<https://learn.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql?view=sql-server-ver16>

**GROUP BY**

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

# String Functions (Transact-SQL)

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

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