ВСТУП ДО БАЗ ДАНИХ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

VIEWs

Переклади, що зустрічаються в літературі: Представлення ? Подання ? Перегляд ? Розріз ? Зріз ? Вид ? Вигляд? Проекція? Уявлення ?

- ***VIEW*** – це віртуальна (тимчасова) таблиця, що є об'єктом бази даних, інформація в якому не зберігається постійно, а формується динамічно **при зверненні** до нього.

У яких випадках використовують VIEW

- для повторного використання запитів SQL
- для спрощення складних запитів
- для виведення фрагментів таблиці замість усієї таблиці
- для захисту даних: користувачам можна надати доступ до певної підмножини таблиць, а не до всіх таблиць
- для зміни форматування та способу відображення даних

Приклад 1. Співробітнику у відділі кадрів потрібно працювати з наступною інформацією про викладачів: ПІБ, адреса, посада, стать, вік та інше, але він не повинен переглядати і взагалі не мати доступ до заробітної плати викладача.

В такому випадку повинна бути можливість створити віртуальну таблицю, яка містить тільки потрібні атрибути з початкової таблиці.

Приклад 2. Співробітнику у відділі кадрів зручніше бачити ПІБ викладача в одному стовпчику , а не в трьох різних, а також потрібно, щоб перед номером телефону було написано «тел.».

Подібні налаштування також зручно зробити за допомогою VIEW.

VIEW – це фактично запит, який виконується кожного разу при зверненні до нього або при його використанні у певній команді. Результат виконання цього запиту в кожний момент часу стає змістом VIEW. У користувача створюється враження, що він працює зі справжньою, реально існуючою таблицею.

VIEW можна використовувати в команді так само, як будь-яку іншу таблицю. До VIEW можна будувати запит, модифікувати його (якщо він відповідає певним вимогам), з'єднувати з іншими таблицями тощо.

Зміст VIEW не фіксований і **оновлюється щоразу**, коли на нього посилаються у команді.

На відміну від інших об'єктів бази даних VIEW не містить власних даних і не займає дискової пам'яті (крім пам'яті, що потрібна на зберігання опису самого VIEW).

Створення VIEW

CREATE VIEW <view_name> [(column_name)] AS SELECT [query]
[WITH CHECK OPTION]

Зазвичай у VIEW використовують імена, отримані безпосередньо з імен полів початкових таблиць. У випадку використання агрегатних функцій або обчислюваних полів, необхідно дати стовпцям нові імена.

Параметр WITH CHECK OPTION вказує серверу виконувати перевірку змін, які відбуваються через VIEW на відповідність критеріям, визначеним в операторі SELECT. Якщо користувач намагається виконати зміни, що призводять до виключення рядка з VIEW, при заданому аргументі WITH CHECK OPTION сервер видасть повідомлення про помилку та зміни будуть відхилені.

```
create view CustomersTotal as
   select TRIM(cust_name), sum(item_price*quantity) as total_sum
   from customers join orders using(cust_id)
      join orderitems using (order_num)
   group by cust_id;

select * from CustomersTotal;
```

Видалення VIEW

DROP VIEW [IF EXISTS] <view_name>

Модифікація VIEW

В деяких випадках (але не у всіх) VIEW можна модифікувати.
Увага! При цьому будуть змінені й дані в початковій таблиці.

Відповідно до стандарту SQL, можна модифікувати VIEW, засновані на запитах:

1) лише по одній таблиці;
2) запит не повинен містити ключові слова DISTINCT, GROUP BY, HAVING;
3) не повинен містити вкладених запитів;
4) не повинен містити операцій UNION, INTERSECT, EXCEPT (MINUS).

ПЕРЕВАГИ VIEW

+ будь-які зміни в таблицях одразу ж відображаються у VIEW;

+ якщо початкова таблиця з якихось причин ділиться на дві або більше, можна переконструювати VIEW, щоб ці зміни були непомітні для користувача;

+ VIEW дозволяють надавати дані в тому вигляді (форматі), в якому вони є зручними для користувача;

+ складні багатотабличні запити зводяться до простих запитів до створеного VIEW;

+ за допомогою VIEW користувач може працювати лише з дійсно потрібною для нього частиною даних;

+ в результаті використання VIEW одні й ті самі таблиці можуть бути представлені користувачам в зовсім різному вигляді;

+ забезпечення цілісності даних (WITH CHECK OPTION).

НЕДОЛІКИ VIEW

- не всі VIEW можна модифікувати;
- якщо VIEW було створено за допомогою команди select * from table то при додаванні нових стовпців в початкову таблицю, вони не будуть відображатись у новому VIEW;
- якщо VIEW було побудовано на основі багатотабличного запиту, то його обробка може зайняти значних витрат часу.

Віртуальні та матеріалізовані VIEWs

Все, зазначене вище, відноситься до **віртуальних** VIEW.

Матеріалізовані VIEW – це фізичні копії базових таблиць (аналогія – знімок чи зображення вихідних базових таблиць).

```
CREATE MATERIALIZED VIEW [ IF NOT EXISTS ] table_name
    [ (column_name [, ...] ) ]
    [ USING method ]
    [ WITH ( storage_parameter [= value] [, ... ] ) ]
    [ TABLESPACE tablespace_name ]
    AS query
    [ WITH [ NO ] DATA ]

ALTER MATERIALIZED VIEW name
    action

REFRESH MATERIALIZED VIEW name

DROP MATERIALIZED VIEW name
```

MATERIALIZED VIEW, як і VIEW, також містить дані, отримані з виразу команди `CREATE MATERIALIZED VIEW`.

MATERIALIZED VIEW, на відміну від звичайного VIEW, попередньо обчислюється і зберігається на диску як об'єкт. Воно НЕ оновлюється при кожному використанні. Натомість матеріалізоване view має оновлюватися вручну командою REFRESH або за допомогою тригерів.

MATERIALIZED VIEW реагує швидше, порівняно зі звичайним VIEW. Це пояснюється тим, що воно попередньо обчислюється і, отже, не витрачається час на виконання запиту чи об'єднання таблиць у запиті, яким визначається MATERIALIZED VIEW.