Лабораторна робота 4

Тема: Основи перетворення даних засобами СУБД Oracle

Завдання

В лабораторній роботі використовується БД, створена у лабораторній роботі №2 відповідно до вашого варіанта.

За кожним етапом створити файл-скрипт *N.sql*, де *N* – номер етапу, в який під час виконання завдань вказувати:

1) умова завдання у вигляді багаторядкового коментаря

2) *SQL*-команда

3) рядки із відповіддю на запит (для *SELECT*-комнад) або реакція СУБД (для помилки) у вигляді багаторядкового коментаря

Для отримання рядків із відповіддю на *SQL*-команда зручно використовувати *SQLPlus*.

Приклад оформлення кожного рішення:

```
/* 1) отримання зарплат працівників за рік з урахуванням надбавки
*/

SELECT
    ename,
    sal,
    12*sal+100
FROM emp;

/*
ENAME             SAL 12*SAL+100
---------- ---------- ----------
KING              5000      60100
TOD               4000      48100
SCOTT             3000      36100
BLAKE             3500      42100
ALLEN             3000      36100
*/
```

Під час Online-заняття бажано встигнути виконати

1-2 завдання етапів 1-4 з урахуванням 2-х таблиць БД

Етап 1. Генератори унікальних послідовностей у СУБД Oracle

1.1. Для всіх таблиць нової БД створити генератори послідовності, щоб забезпечити автоматичне створення нових значень колонок, які входять у первинний ключ. Врахувати наявність рядків у таблицях. Виконати тестове внесення одного рядка до кожної таблиці.

1.2 Для всіх пар взаємопов'язаних таблиць створити транзакції, що включають дві INSERT-команди внесення рядка в дві таблиці кожної пари з урахуванням зв'язку між PK-колонкою першої таблиці і FK-колонкою 2-ї таблиці пари з використанням псевдоколонок NEXTVAL і CURRVAL.

1.3 Отримати інформацію про створені генератори послідовностей, використовуючи системну таблицю СУБД Oracle.

1.4 Використовуючи СУБД Oracle >= 12 для однієї з таблиць створити генерацію унікальних значень PK-колонки через DEFAULT-оператор. Виконати тестове внесення одного рядка до таблиці.

Етап 2. Вбудовані функції Oracle для перетворення даних

2.1 Для однієї з таблиць створити команду отримання символьних значень колонки з переведенням першого символу у верхній регістр, інших у нижній. При виведенні на екран визначити для вказаної колонки нову назву псевдоніму.

2.2. Модифікувати рішення попереднього завдання, створивши команду оновлення значення вказаної колонки у таблиці.

2.3 Для однієї з символьних колонок однієї з таблиць створити команду отримання мінімальної, середньої та максимальної довжин рядків.

2.4 Для колонки типу date однієї з таблиць отримати кількість днів, тижнів та місяців, що пройшли до сьогодні.

Етап 3. SQL-команди з операціями з'єднання таблиць. SQL-стиль

3.1 Для будь-яких двох таблиць створити команду отримання декартового добутку.

3.2 Для двох таблиць, пов'язаних через PK-колонку та FK-колонку, створити команду отримання двох колонок першої та другої таблиць з використанням екві-з’єднання таблиць. Використовувати префікси.

3.3 Повторити рішення попереднього завдання, застосувавши автоматичне визначення умов екві-з’єднання.

3.4 Повторити рішення завдання 3.2, замінивши еквіз'єднання на зовнішнє з'єднання (лівостороннє або правостороннє), яке дозволить побачити рядки таблиці з PK-колонкою, не пов'язані з FK-колонкою.

Етап 4. SQL-команди з операціями з'єднання таблиць. Oracle-стиль

4.1 Повторити рішення завдання 3.1

4.2 Повторити рішення завдання 3.2

4.3 Повторити рішення завдання 3.4

Етап 5. SQL-команди з операціями над множинами

5.1 Для однієї з таблиць створити команду отримання кількості рядків таблиці, згрупованих по одній з колонок, яка також повинна бути отримана, об'єднавши її з командою отримання загальної кількості рядків із зазначенням слова «Разом:», наприклад:

Колонка	*Кількість*
значення1	*10*
значення2	*15*
Разом:	*25*

При виконанні завдання необхідно врахувати, що підсумковий рядок завжди повинен бути останнім, незалежно від значень колонки у попередніх рядках.

Етап 6 Документування результатів роботи на Веб-сервісі *GitHub*

6.1 Розпочинаючи роботу над документуванням рішень лабораторної роботи, необхідно у вашому *GitHub*-репозиторії створити *Issue* з назвою «*tasks-of-laboratory-work-4*».

1) створити *Issue* з назвою «*tasks-of-laboratory-work-4*»;

2) підключити до *Issue* ваш *GitHub-project* (правий розділ «*Projects*» сторінки з *Issue*);

3) змінити статус *Issue* з «*Todo*» на «*In progress*», автоматично перевівши *Scrum*-картку з цим *Issue* на *Scrum*-дошку «*In progress*»;

4) створити нову *Git*-гілку з назвою, яка відповідає назві *Issue*, наприклад, «*tasks-of-laboratory-work-4*» (використовується посилання «*Create a branch*» у правому розділі «*Development*» сторінки з *Issue*).

6.2 Після створення *Git*-гілки перейти до цієї гілки для створення оновлень файлів *Git*-репозиторію.

6.3 У новій гілці *Git*-репозиторію створити каталог з назвою «*4-Simple-Data-Transformation*» (кнопка *«Add file» - «Create new file»*), при створенні якого одночасно створити файл *README.md* з першим рядком *«4 Основи перетворення даних засобами СКБД Oracle»* зі стилем «Заголовок 3-го рівня» мови розмітки *Markdown* (три символи решітка ###).

6.4 Розмістити в каталозі «*4-Simple-DQL-DML4-Simple-Data-Transformation*» *GitHub*-репозиторія файли 1.sql, 2.sql, 3.sql, 4.sql, 5.sql з рішеннями завдань відповідних етапів.

6.5 Виконати запит *Pull Request*, розпочавши процес *Code Review*.

Під час створення *Pull Request* необхідно вказати:

- *Reviewers = Oleksandr Blazhko, Maria Glava*;
- *Labels = enhancement (New feature or request)*;
- *Projects* = посилання на *GitHub-project*.

Завершення процесу *Code Review* відбудеться до початку нового заняття, після чого викладач закриє *Issue*, завершуючи процес виконання завдань з лабораторної роботи.

Під час консультації в понеділок ви зможете отримати більше коментарів щодо ваших рішень.