Национальный исследовательский ядерный университет «МИФИ» (Московский Инженерно–Физический Институт)
Кафедра №42 «Криптология и кибербезопасность»

Лабораторная работа №1-3: «Сложные запросы на выборку. Соединения»

Рекомендуемая дата защиты: 07.05.2022
Предельная дата защиты: 22.05.2022

Цель работы

Приобретение опыта написания более сложных запросов к базе данных Oracle, в частности — с применением аналитических функций и операций соединения.

Ход работы

1. Подготовить «легенду»: на какие вопросы требуется знать ответы сотрудникам организации в выбранной предметной области?
2. Разработать и проверить SQL-запросы, реализующие эти ответы. Необходимо разработать не менее 8 различных запросов с применением различных возможностей, предоставляемых стандартом SQL и Oracle Database. Необходимо уделить особое внимание различным случаям соединения таблиц.
3. Оформить отчёт.

Основные возможности сложных запросов на выборку

1. Подзапросы: вместо переменной или литерала в запросе можно использовать вложенный подзапрос. Этот подзапрос должен возвращать одно значение, либо один столбец значений, если оператор работает с диапазонами. Например, этот запрос вернёт список событий, запланированных в крупных городах:

```
SELECT * FROM events WHERE city_id IN ( SELECT city_id FROM cities WHERE pop > 1000000 );
```

Этот запрос вернёт стоимость изделий в долларах:

```
SELECT name, cost_rub / ( SELECT exchange_rate FROM stock WHERE currency = 'USD' ) FROM goods;
```

2. Соединение. Такой запрос сопоставляет друг другу ряды из двух таблиц. Без дополнительной условной фильтрации, каждая строка из одной таблиц соединяется с каждой строкой из другой. Операция соединения применяется для денормализации данных. Например, этот запрос позволяет извлечь список мероприятий с указанием города, где они проводятся:

```
SELECT cities.name, events.name FROM events, cities WHERE events.city_id = cities.city_id;
```

Таблицу можно соединять саму с собой. Например, этот запрос выводит список потенциальных пар по списку людей:

```
SELECT p1.name, p2.name FROM people p1, people p2 WHERE p1.gender > p2.gender;
```

3. Иерархические запросы (CONNECT BY, PRIOR, START WITH). Строятся к таблицам, где одни записи подчинены другим. Типичные примеры: таблица сотрудников, где для каждого указан непосредственный начальник; таблица обработки материалов, где для каждого типа сырья указан производимый из него продукт. При выполнении иерархического запроса создаются дополнительные столбцы LEVEL, CONNECT_BY_ISCYCLE, CONNECT_BY_ISLEAF, а также доступна функция CONNECT_BY_PATH. Например, этот запрос выводит уровень сотрудника в иерархии:

```
SELECT name, LEVEL FROM employees CONNECT BY PRIOR employee_id = boss_id;
```

Этот запрос выводит сотрудников, у которых есть люди в подчинении:

```
SELECT name FROM employees CONNECT BY PRIOR employee_id = boss_id
    WHERE CONNECT_BY_ISLEAF = 0;
```

4. Аналитические функции (OVER). Предназначены, для анализа того, как значения полей меняются с течением времени (или любого другого параметра). Позволяют извлечь значение заданного поля не

для текущей записи, а для предшествующей; вычислить скользящее среднее, и т.д. Так как аналитические функции оперируют понятиями «предыдущий», «следующий», «первый (лучший)», — их применение осмысленно только при наличии сортировки. Поэтому, параметр и порядок сортировки при использовании аналитической функции указывать обязательно. При этом, результаты запроса можно отсортировать по-другому. Пример: этот запрос вычисляет изменение дохода за месяц:

```
SELECT
        income,
        (income - lag(income,1,NULL) OVER (ORDER BY month)) as change
FROM income;
```

Этот запрос вычисляет баланс организации в каждом месяце по доходам и расходам:

```
SELECT
        month,
        sum(income) OVER
            (ORDER BY month
            ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
        - sum(expense) OVER
            (ORDER BY month
            ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
        AS balance
FROM income;
```

5. Теоретико-множественные операции (UNION, UNION ALL, INTERSECT, MINUS). Позволяют построить объединение, пересечение, разность множеств рядов из двух таблиц. Ценность этих инструкций состоит в возможности объединить две различных таблицы в одну. Оператор UNION устраняет возникающие при объединении дубликаты, UNION ALL этого не делает, поэтому работает существенно быстрее.

Оформление отчёта

1. Титульный лист: название института, название лабораторной работы, имя, фамилия, номер группы, год,...
2. Список выполненных запросов SQL с описаниями их назначения и смысла;
3. Приложение: Листинг использованных инструкций SQL. В тексте отчёта должна быть ссылка на приложение;
4. Заключение: краткое описание проделанной работы.