IT CAREER HUB

Урок 4.2. Представления View

Связь между Python, SQL и BI-инструментами

 **Важно!**
База данных с доступом на запись:
hostname: ich-edit.edu.itcareerhub.de
MYSQL_USER: ich1
MYSQL_PASSWORD: ich1_password_ilovedbs

В современной аналитике часто используется комбинация Python, SQL и BI (Business Intelligence) инструментов. Эти технологии работают вместе для эффективного анализа данных, автоматизации процессов и визуализации информации. Вот как они связаны и взаимодействуют друг с другом в контексте баз данных:

1. **Python: Загрузка данных в базу данных с помощью Python и API**
    - Сбор данных
    - Обработка данных
    - Загрузка данных
2. **SQL (Structured Query Language)**:
    - Хранение данных
    - Запрос данных
    - Обновление данных
3. **BI (Business Intelligence) инструменты**:
    - Визуализация данных
    - Анализ данных
    - Отчеты и дашборды

IT CAREER HUB

Процесс загрузки данных, создания витрины и подключения к BI-инструменту

1. **Загрузка данных в базу данных с помощью Python**
    - **Получение данных**: Python используется для автоматизированного сбора данных из внешних источников, таких как веб-API. Это может быть информация о продажах, клиентских данных, или других данных, которые нужно сохранить и проанализировать.
    - **Обработка данных**: После получения данных Python обрабатывает их. Это может включать очистку данных, преобразование форматов, фильтрацию ненужных записей и т.д.
    - **Загрузка в базу данных**: Обработанные данные затем загружаются в базу данных, такую как MySQL. Это происходит с помощью Python-скрипта, который подключается к базе данных и вставляет данные в соответствующие таблицы.

2. **Создание витрины данных в базе данных**
    - **Создание витрины**: После загрузки данных в базу данных создается витрина данных. Витрина данных – это специальная структура (например, представление или агрегированная таблица), которая организует данные для удобного анализа. Она может включать агрегированные данные, такие как суммы продаж по месяцам, средние значения и т.д.
    - **Агрегация и трансформация**: В витрине данных данные могут быть агрегированы и преобразованы, чтобы они были более удобными для анализа и визуализации. Например, создание сводной таблицы, которая суммирует продажи по каждому продукту или региону.

3. **Подключение BI-инструмента для визуализации**
    - **Подключение к базе данных**: BI-инструмент (например, Tableau) подключается к базе данных, где хранятся данные и витрины. Этот процесс включает указание адреса базы данных, учетных данных и названия таблиц или представлений, которые нужно использовать.
    - **Создание отчетов и визуализаций**: После подключения BI-инструмент извлекает данные из базы данных и предоставляет возможности для создания различных визуализаций, таких как графики, диаграммы и дашборды. Это позволяет пользователям легко анализировать данные и принимать решения на основе полученных результатов.

- **Обновление данных**: BI-инструмент может регулярно обновляться с новыми данными из базы данных, что обеспечивает актуальность отчетов и визуализаций.

Итоговый процесс:

1. **Python** автоматически собирает и обрабатывает данные, затем загружает их в **MySQL**.
2. В **MySQL** создается витрина данных для упрощенного анализа и доступа к агрегированным данным.
3. **BI-инструмент** (например, Tableau) подключается к базе данных, извлекает данные из витрины и создает визуализации, помогающие в принятии бизнес-решений.

IT CAREER HUB

Представления (Views) в SQL: Зачем нужны и как их создавать

Представление (View) в SQL — это виртуальная таблица, которая создается на основе результатов SQL-запроса.

Представление не хранит данные физически, вместо этого оно отображает результат запроса к базе данных каждый раз, когда вы обращаетесь к нему. Это позволяет абстрагироваться от физической структуры данных и сосредоточиться на их логическом представлении.

Зачем нужны представления

1. **Упрощение сложных запросов:**
   - Представления позволяют объединить сложные SQL-запросы в простые виртуальные таблицы. Это упрощает использование данных и делает запросы более читаемыми.
2. **Повышение безопасности:**
   - Вы можете предоставить доступ к определенным данным без раскрытия всей таблицы. Например, создать представление, которое показывает только определенные колонки или строки, и предоставить доступ к этому представлению вместо самой таблицы.
3. **Упрощение доступа к данным:**
   - Представления могут объединять данные из нескольких таблиц, создавая удобный интерфейс для пользователей, которые не должны знать внутреннюю структуру базы данных.
4. **Изоляция данных:**
   - Представления позволяют изолировать бизнес-логику от данных. Например, вы можете создать представление, которое автоматически вычисляет дополнительные значения на основе данных в таблице.
5. **Поддержка совместимости:**
   - Если структура баз данных меняется, представления могут служить промежуточным слоем, обеспечивая совместимость с устаревшими приложениями и запросами.

Как создавать представления

Для создания представлений используется команда CREATE VIEW. Вот общий шаблон и примеры создания представлений:

```
Unset
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Задание для закрепления

Создайте представление на основе всех записей таблицу Employees в Вашей базе данных

```
Unset
USE your_database;

CREATE VIEW firs_view AS

SELECT * FROM Employees;
```

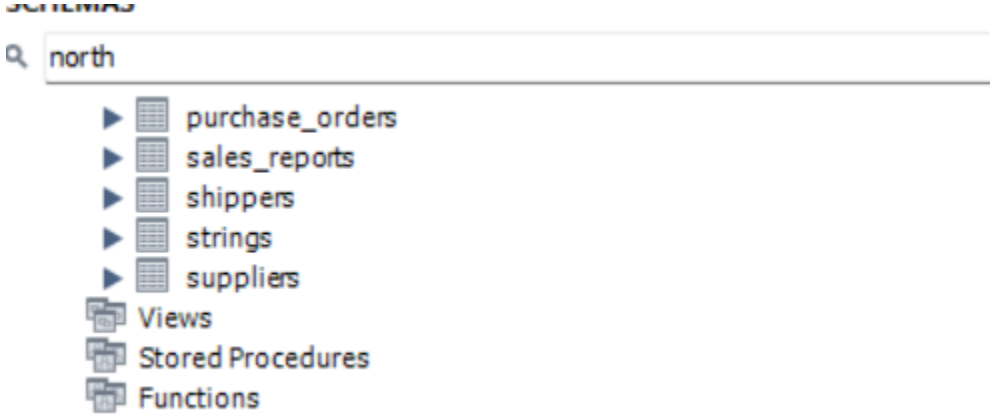
Создайте представление выбрав первые два столбца из таблицы Employees в Вашей базе данных

```
Unset

CREATE VIEW second_view AS

SELECT EmployeeID, FirstName

FROM Employees
```



Предварительно обновить схему данных