Базы данных и экспертные системы

Инструкции и операции на языке запросов SQL

В данной лекции будут рассмотрены средства SQL по определению базы данных, формирование запросов и операторы манипулирования данными.

- Тремя, наиболее важными языками, появившимися в результате публикации Коддом статей о реляционной модели и реляционных языках, стали SQL (Structured Query Language), QBE (Query by Example) и QUEL (Query Language).
- SQL стал результатом исследовательского проекта System R компании IBM.
- В конце 70-х SQL стал доступен для широкого использования, в качестве языка системы Oracle. В 1986 году был принят стандарт ANSI для SQL. Этот стандарт был пересмотрен в 1989 году и, в последствии, в 1992. К настоящему времени SQL является стандартом в системах «клиент-сервер».

- **SQL** — непроцедурный **язык**: серверу базы данных сообщается, что нужно сделать и каким образом.

- Для обработки запроса сервер базы данных транслирует команды **SQL** во внутренние процедуры.

- Благодаря тому, что **SQL** скрывает детали обработки данных, его легко использовать.

Что можно делать с помощью SQL?

- **SQL** позволяет
  - создавать таблицы данных,
  - хранить данные,
  - получать данные,
  - изменять данные,
  - изменять структуру таблиц,
  - объединять данные,
  - выполнять вычисления,
  - обеспечивать защиту данных.

Команды SQL

- **Команды языка определения данных — DDL (Data Definition Language).** Эти **SQL** команды можно использовать для создания, изменения и удаления различных объектов базы данных.
- **Команды языка управления данными — DCL (Data Control Language).** С помощью этих **SQL** команд можно управлять доступом пользователей к базе данных и использовать конкретные данные (таблицы, представления и т.д.).
- **Команды языка управления транзакциями — TCL (Transaction Control Language).** Эти **SQL** команды позволяют определить исход транзакции.
- **Команды языка манипулирования данными — DML (Data Manipulation Language).**

Стандартные SQL команды для взаимодействия с реляционными БД:

- CREATE
- SELECT
- INSERT
- UPDATE
- DELETE
- DROP

Данные команды разбиты на категории: **DDL – Data Definition Language**

Команда	Описание
CREATE	Создаёт новую таблицу, вид таблицы или другой объект в БД.
ALTER	Изменяет существующий объект БД (например, таблицу).
DROP	Удаляет указанную таблицу, вид таблицы или другой объект БД.

DML – Data Manipulation Language

Команда	Описание
SELECT	Получает определённые данные из одной или нескольких таблиц.
INSERT	Создаёт запись
UPDATE	Изменяет записи
DELETE	Удаляет записи

DCL – Data Control Language

Команда	Описание
GRANT	Даёт все привилегии пользователю
REVOKE	Отменяет привилегии отданные пользователю

Команда SQL Create Table

- Команда SQL Create Table предназначена для описания структуры таблицы. Команда SQL Create Table создает пустую таблицу (без строк).
- Пример 1
- CREATE TABLE Persons (P_Id int,
- LastName varchar(255),
- FirstName varchar(255),
- Address varchar(255),
- City varchar(255))

Команда SQL “INSERT”

- **Команда INSERT** добавляет строки в таблицу.
- Утверждение **INSERT** с фразой **VALUES** добавляет одиночную строку к таблице. Эта строка содержит значения, определенные фразой **VALUES**.
- Утверждение **INSERT** с **подзапросом** вместо фразы **VALUES** добавляет к таблице все строки, возвращенные **подзапросом**. Сервер обрабатывает **подзапрос** и вставляет каждую возвращенную строку в таблицу.
- Если подзапрос не выбирает никакие строки, сервер не вставляет никакие строки в таблицу.

Команда SQL “INSERT”

- **Подзапрос** может обратиться к любой таблице, включая целевую таблицу утверждения **INSERT**.
- Сервер назначает значения полям в новых строках, основанных на внутренней позиции столбцов в таблице и порядке значений фразы **VALUES** или в списке выбора запроса. Если какие-либо столбцы пропущены в списке столбцов, сервер назначает им значения по умолчанию, определенные при создании таблицы.
- Если любой из этих столбцов имеет **NOT NULL** ограничение то сервер возвращает ошибку, указывающую, что ограничение было нарушено и отменяет утверждение **INSERT**.

INSERT INTO

INSERT INTO Пример 2. Запишите:

Вставка новой строки в таблицу table_name с указанием вставки данных в нужные нам колонки.

```
INSERT INTO table_name
VALUES (’1',’165',’0',’name’)
```

В базе данных **MySQL** имеется возможность вставлять множество новых строк, используя одну команду **INSERT**.

INSERT INTO Пример 3.

Вставка несколько строк в таблицу table_name.

```
INSERT INTO table_name (tbl_id, chislo, chislotwo, name)
VALUES (’1',’159',’34',’name1')
       (’2',’14',’61',’name2')
       (’3',’356',’8',’name3')
```

“SELECT”

- **Select** – перечисляет столбцы, которые должны войти в результирующую таблицу.
- Это всегда столбцы одной или нескольких заранее определенных таблиц. Если результирующая таблица состоит из нескольких столбцов, они перечисляются через запятую. Кроме того, используется символ * который означает – “строка целиком”.

- **From** – задает одну или более таблиц, к которым обращается запрос. Все столбцы, перечисленные во фразах select и where должны существовать в одной из таблиц, перечисленных в данной команде.

- **Where** – данная фраза содержит условие, на основании которого выбираются строки таблиц. Данная фраза может содержать множество разнообразных условий.

“SELECT”

Рассмотрим пример простого запроса 1: “Кто работает аудитором?”.

```
Select name
From worker
Where skill_type = ‘аудитор’
```

- В нашем примере условие состоит в том, что значение столбца skill_type должно равняться “аудитор”.

“SELECT”

Запрос 2: У кого почасовая ставка от 8 до 10 ч?

```
Select *
From worker
Where hrly_rate >= 8 and hrly_rate <= 10
```

- В этом примере продемонстрировано использование операторов сравнения и булевого оператора. Всего в SQL используется 6 операторов сравнения: =, <>, <, >, >=, <= и три булевых оператора: and, or, not.
- Корме того в команде where могут использоваться операторы between и in. Для рассмотренного примера фразу where можно было записать следующим образом: Where hrly_rate between 8 and 10.

Команда DELETE

- **Команда DELETE** удаляет строки из таблицы или представления основной таблицы базы данных

- При выдаче утверждения **DELETE** включается любой **DELETE**-триггер, определенный на таблице.
  Команда DELETE Пример №3
  Удаление всех строк без исключения из таблицы:

```
DELETE FROM temp_assign
```

- **Команда DELETE Пример №4.**
  Удаляет из таблицы всех работников, у которых зарплата меньше 100 000 :

```
DELETE FROM emp WHERE JOB = ‘SALESMAN’ AND COMM < 100000
```

- В данном примере **команда DELETE** удаляет все строки, которые попадают под условие JOB = ‘SALESMAN’ AND COMM < 100000

Команда UPDATE

- **Команда UPDATE** — производит изменения в уже существующей записи или во множестве записей в таблице**SQL**. Изменяет существующие значения в таблице или в основной таблице представления.

- **WHERE** — определяет диапазон изменяемых строк теми, для которых определенное условие является **TRUE**; если опускается эта фраза, модифицируются все строки в таблице или представлении. При выдаче утверждения **UPDATE** включается любой **UPDATE-триггер**, определенный на таблице.

•

Подзапросы. Если предложение **SET** содержит **подзапрос**, он возвращает точно одну строку для каждой модифицируемой строки. Каждое значение в результате подзапроса назначается соответствующим столбцам списка в круглых скобках. Если подзапрос не возвращает никакие строки, столбцу назначается **NULL**. **Подзапросы** могут выбирать данные из модифицируемой таблицы. Предложение **SET** может совмещать выражения и **подзапросы**.

Команда UPDATE Пример 5
Изменение для всех покупателей рейтинга на значение, равное 200:

```
UPDATE Customers
SET rating = 200
```

Команда UPDATE Пример 6
Замена значения столбца во всех строках таблицы, как правило, используется редко. Поэтому в команде **UPDATE**, как и в команде **DELETE**, можно использовать предикат. Для выполнения указанной замены значений столбца rating, для всех клиентов, которые обслуживаются «Казстандарт» (snum = 1001), следует ввести:

```
UPDATE Clients
SET rating = 200
WHERE snum = 1001
```