Лекция №7 13.10.23

SQL (продолжение)

Создание таблиц

Для описания таблиц используется следующий синтаксис:

```
CREATE TABLE <имя таблицы> (<определение столбцов> [,…]);
```

Любая таблица должна иметь уникальное имя в пределах схемы и хотя бы 1 столбец.

Определение столбца имеет вид:

```
<определение столбца> = <имя столбца> <тип данных> [ограничения];
```

Определение ограничений задается как:

```
[CONSTRAINT <имя ограничения>] <описание ограничения определённого типа>;
```

Типы ограничений:

- `CHECK` - позволяет задать условие проверки корректности добавляемых значений
- `PRIMARY KEY` - первичный ключ
- `UNIQUE` - уникальные значения
- `NOT NULL` - не пустое значение
- ограничение внешнего ключа `[FOREIGN KEY (<имя столбца>)] REFERENCES <имя таблицы> […..]`

На один и тот же столбец может быть наложено несколько ограничений. На внешний ключ можно наложить правило поддержки ссылочной целостности, используя выражения `ON {DELETE | UPDATE} [NO ACTION | CASCADE | SET NULL | SET`

`DEFAULT]`. Где `CASCADE` - каскадное удаление. `SET NULL | SET DEFAULT` - замена внешнего ключа на неизвестное или по умолчанию. `NO ACTION` - удалить и обновить строки можно только, если с ними не связаны никакие значения.

Пример:

```
CREATE TABLE Сотрудник (
    Номер_договора INT PRIMARY KEY,
    Фамилия VARCHAR(20) NOT NULL,
    Имя VARCHAR(20) NOT NULL,
    Отчество VARCHAR(20) DEFAULT '-',
    Оклад NUMERIC(7,2) CHECK (Оклад >= 7500),
    UNIQUE (Фамилия, Имя, Отчество)
);
```

Изменение таблицы

```
ALTER TABLE <имя таблицы> <действия>;
```

1. Добавление ограничения уровня таблицы:

   ```
   ADD <ограничения>;
   ```

2. Добавление столбца:

   ```
   ADD <имя столбца, тип…>;
   ```

3. Изменение столбца:

   ```
   ALTER COLUMN;
   ```

4. Удаление столбца:

   ```
   DROP COLUMN;
   ```

Пример:

```
ALTER TABLE Сотрудник
    ADD Номер_отдела INT REFERENCES Отдел (Номер_отдела) ON DELETE SET NULL;
```

Удаление таблицы

```
DROP TABLE <имя таблицы>;
```

Вставка записей

```
INSERT INTO <имя таблицы> [<столбцы>] VALUES [<значения>];
```

1. После `VALUES` должны быть указаны значения для всех существующих столбцов таблицы
2. Значения должны быть перечислены в порядке указания столбцов при создании таблицы

Пример:

```
INSERT INTO отдел
VALUES (200 + 80, 'Education', NULL, DEFAULT)
```

или

```
INSERT INTO отдел(название, id)
VALUES ('Education', 280);
```

Вставка путём копирования

```
INSERT INTO <имя таблицы> [<столбцы>] SELECT...;
```

Обновление записей

```
UPDATE <имя таблицы> SET <имя столбца> = <значение> [,…..] [WHERE <проверок нет>];
```

Пример:

```
UPDATE Отдел SET id = id + 100;
```

Раздел `WHERE` позволяет ограничить множество затрагиваемых командой строкой.

В языке SQL оператор присваивания и оператор сравнения выглядят одинаково, однако запись вида `<имя столбца> = NULL` не допускается. Для сравнения с `NULL` используют операторы `IS NULL` или `IS NOT NULL`.

Возможные причины ошибок при вставке и изменении записи. При попытке вставки и обновления строк СУБД проверяет, соответствует ли тип данных вносимых значений, не нарушают ли новые значения ограничения целостности:

1. Вставка производится в идентификатор, который является первичным ключом, который не должен повторяться
2. Невозможность вставить пустые значения
3. Ограничения целостности

Удаление записей

```
DELETE FROM <имя таблицы> [WHERE <предмет>];
```

Если отсутствует раздел `WHERE`, из таблицы будут удалены все строки.

Альтернатива:

```
TRUNCATE TABLE <имя таблицы>;
```

Работает быстрее и потребляет меньше ресурсов.

При удалении также могут возникнуть ошибки. Если возникнет ошибка, то удаление завершится с ошибкой. Распространённая ошибка при удалении:

- Нарушение ограничения целостности

Составление простых запросов

Для составления запросов используется команда-субъект. Эта команда начинается с разделов:

```
SELECT… (5)
FROM… (1)
[WHERE… ] (2)
[GROUP BY… ] (3)
[HAVING…] (4)
[ORDER BY…] (6)
```

Сначала сервер смотрит (1). Затем фильтрует с помощью (2). Если требуется разделить раздел на группы, то (3). Потом (4) (как `WHERE`, только для групп). Потом (5) и (6), который осуществляет сортировку по определённому критерию.

При отправке любой команды на выполнение, СУБД производит несколько дополнительных проверок:

1. Проверка на правильное составление запросов (проверка синтаксиса)
2. Проверка на смысловую корректность запроса (проверка семантики)
3. Проверка полномочий

Для составления простых запросов хватает разделов `SELECT`, `FROM`, `WHERE`.

```
SELECT [DISTINCT] <имя столбцов, …….>
FROM <имя таблицы>
WHERE <предикат>;
```

`DISTINCT` служит для обхода уникальных записей. Если мы указали DISTINCT и получили строки-дубликаты, копии удалятся. `[DISTINCT]` указывается сразу после `SELECT` и действует на все столбцы, указанные в `SELECT`.

`*` - выбираем все имеющиеся столбцы в таблице `(SELECT *[DISTINCT])`

При отборе записей их можно сравнивать с шаблонами. Сравнение производится командой `LIKE` / `NOT LIKE`.

Проверка на попадание в диапазон служит оператор `BETWEEN`. Пишется как параметр `WHERE <перекат>`.

```
BETWEEN … AND …
≥ and ≤
```

Проверка на попадание в список значений `IN` и `NOT IN`.