

Управление данными

Лекция 3

Ссылочная целостность и начала DML

Свечников С.В.

Содержание занятия

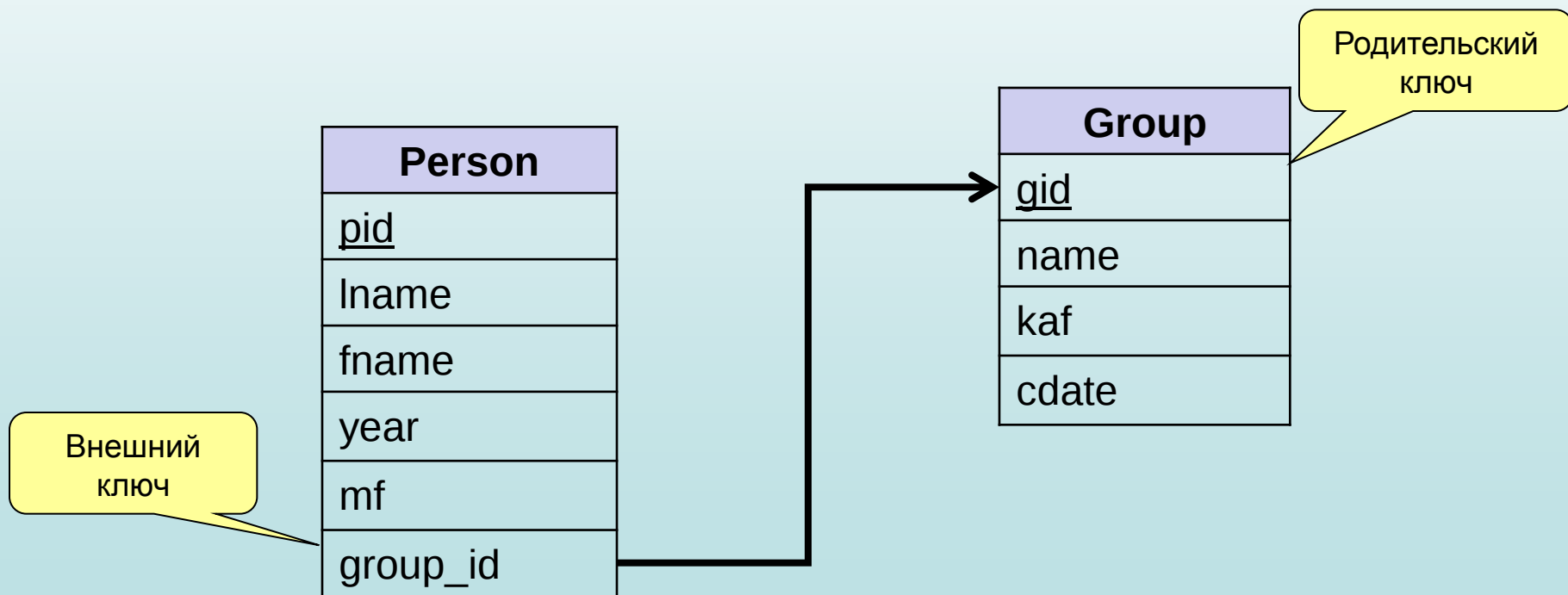
- 01. Поддержание ссылочной целостности
- 02. Ввод, изменение и удаление данных
- 03. Выборка информации из таблиц

01. Поддержание ссылочной целостности

Внешние и родительские ключи

Наличие внешнего ключа означает, что каждая строка таблицы ссылается на одну и только одну строку таблицы, содержащей родительский ключ.

Структура внешнего ключа должна соответствовать структуре родительского ключа.



Ограничение внешнего ключа

Ограничение FOREIGN KEY вводится при определении или изменении таблицы.

Синтаксис

FOREIGN KEY список столбцов REFERENCES таблица [список столбцов]

Список столбцов родительского и внешнего ключей должны быть совместимы:

- ☐ количество столбцов должно быть одинаково;
- ☐ первый, второй и остальные столбцы должны иметь одинаковые типы данных и размер.

Внешний ключ может содержать только те значения, которые присутствуют в родительском ключе, а также NULL.

Ограничение внешнего ключа

Пример

Ограничение FK на таблицу

```
CREATE TABLE person (  
    pid      integer PRIMARY KEY,  
    lname    varchar(255) NOT NULL,  
    fname    varchar(255),  
    city     varchar(100),  
    group_id integer,  
    FOREIGN KEY (group_id) REFERENCES gruppa(gid));
```

Предварительно должна быть создана таблица gruppa.

Действия, выполняемые по ссылке

В ограничении FOREIGN KEY можно указать, что должно происходить со значением внешнего ключа при удалении или изменении родительского ключа.

Операторы, изменяющие содержимое родительского ключа:

- ☐ UPDATE – изменяет существующие значения в БД
- ☐ DELETE – удаляет существующие строки из БД

Разрешается независимо изменять поведение операторов путем указания режимов обновления и удаления.

Всего существует 4 режима действий:

- ☐ CASCADE
- ☐ SET NULL
- ☐ SET DEFAULT
- ☐ NO ACTION

Действия, выполняемые по ссылке

CASCADE – означает, что внешний ключ будет приведен в соответствие с родительским ключом.

❑ UPDATE обновит FK в соответствие с PK

❑ DELETE удалит все строки FK

SET NULL – означает, что значения внешнего ключа будут установлены в NULL.

SET DEFAULT – означает, что для внешнего ключа будет установлено значение по умолчанию

NO ACTION – применяется по умолчанию и соответствует поведению системы.

Действия, выполняемые по ссылке

Синтаксис

```
[ ON UPDATE {CASCADE | SET NULL | SET DEFAULT | NO ACTION} ]  
[ ON DELETE {CASCADE | SET NULL | SET DEFAULT | NO ACTION} ]
```

Пример

```
CREATE TABLE person (  
    pid        integer PRIMARY KEY,  
    lname      varchar(255) NOT NULL,  
    fname      varchar(255),  
    year       int(3),  
    phone      int(11),  
    mf         varchar(10),  
    city       varchar(100),  
    group_id   integer,  
    FOREIGN KEY (group_id) REFERENCES grupp(gid)  
    ON UPDATE CASCADE ON DELETE SET NULL);
```

```
UPDATE grupp SET gid=111 WHERE gid=1;  
DELETE FROM grupp WHERE gid=111;
```

Внешние ключи в той же таблице

Родительский ключ для ограничения FOREIGN KEY может содержаться в той же самой таблице.

Пример

```
CREATE TABLE person (  
    pid            integer PRIMARY KEY,  
    lname          varchar(255) NOT NULL,  
    fname          varchar(255),  
    year           int(3),  
    mf             varchar(10),  
    manager_id     integer,  
    FOREIGN KEY (manager_id) REFERENCES person (pid)  
);
```

Person	
<u>pid</u>	
lname	
fname	
year	
mf	
manager_id	

02. Ввод, изменение и удаление данных

Управление содержимым таблицы

Значения помещаются в поля и удаляются из них с помощью трех операторов DML:

- ☐ INSERT
- ☐ UPDATE
- ☐ DELETE

Для них используется название – **операторы обновления**.

Оператор INSERT

Команда **INSERT** добавляет данные в таблицу

Синтаксис

```
INSERT INTO имя_таблицы  
VALUES (значение, ...);
```

Пример

```
INSERT INTO person  
VALUES(11, 'Пилюлькин', 'Евгений', 22, 5487455, 'муж', 'Одесса', 2);
```

```
SELECT * FROM person;
```

NULL в INSERT

NULL-значения можно добавить как и любое другое значение.

Пример

```
INSERT INTO person
```

```
VALUES(15, 'Ключев', 'Петр', 18, 4758425, 'муж', NULL, NULL);
```

```
SELECT * FROM person;
```

Указание столбцов

При добавлении строк можно явно указать имена столбцов для добавления значений.

Пример

```
INSERT INTO person (lname, fname, year, mf)  
VALUES('Ключев', 'Петр', 18, 'муж');
```

```
SELECT * FROM person;
```

Вставка результатов запроса

Команду INSERT можно использовать для перемещения значений из одной таблицы в другую.

Пример

```
INSERT INTO person3
```

```
SELECT pid, lname, fname, year, mf, group_id
```

```
FROM person
```

```
WHERE mf='муж';
```

```
SELECT * FROM person3;
```

При вставке результатов запроса должны выполняться условия:

- ☐ Таблица должна быть создана.
- ☐ Таблица должна иметь столбцы совпадающие по типу данных с таблицей, из которой добавляются значения.

Оператор UPDATE

Изменение всех или некоторых значений существующих строк выполняется с помощью команды UPDATE.

Синтаксис

```
UPDATE имя_таблицы  
SET присваивание, ...  
[WHERE условие]
```

Пример

```
UPDATE person SET lname='Шпунтик' WHERE pid=10;  
UPDATE person SET lname='Шпунтик';
```

```
SELECT * FROM person;
```

Важно:

Не разрешается обновлять несколько таблиц в одном операторе.

Оператор UPDATE

В предложении SET разрешается использовать выражения, включая те, что используют модифицируемый столбец.

Пример

```
UPDATE person SET year=year*12;
```

```
SELECT * FROM person;
```

В предложении SET можно изменять значения столбцов на NULL.

Пример

```
UPDATE person SET city=NULL
```

```
WHERE city='Ростов';
```

```
SELECT * FROM person;
```

Оператор DELETE

Для удаления строк из таблицы предназначена команда DELETE. Она удаляет строки целиком, а не отдельные значения.

Синтаксис

```
DELETE FROM имя_таблицы  
WHERE условие_отбора_записей
```

Пример

```
DELETE FROM person WHERE pid=11;  
DELETE FROM person WHERE city='Москва';  
DELETE FROM person;  
  
SELECT * FROM person;
```

03. Выборка информации из таблиц

Оператор SELECT

Команда SELECT выбирает данные из таблицы.

Синтаксис

```
SELECT [DISTINCT | DISTINCT ROW | ALL] select_expression,...  
[FROM table_references]  
[WHERE where_definition]  
[GROUP BY {unsigned_integer | col_name | formula}  
[HAVING where_definition]  
[ORDER BY {unsigned_integer | col_name | formula}  
[ASC | DESC], ...]
```

Примеры

```
SELECT * FROM person;
```

```
SELECT pid, lname, fname, year, phone, mf, city, group_id FROM person;
```

Оператор SELECT

Можно задать вывод определенных столбцов из таблицы.

Пример

```
SELECT lname, fname FROM person;
```

Дубликаты

Для устранения дубликатов используется аргумент DISTINCT.

Пример

```
SELECT DISTINCT city FROM person;
```

Важно:

- ☐ Если в таблице не содержится избыточных данных, от использования DISTINCT лучше отказаться.
- ☐ При отсутствии реальной потребности в DISTINCT система будет выполнять лишнюю работу, что замедлит выполнение запросов.
- ☐ DISTINCT можно применять к любому количеству столбцов.

Альтернативой DISTINCT является ALL.

Пример

```
SELECT ALL city FROM person;
```

Предложение WHERE

Для отбора только определенных строк можно использовать условия выборки, которые указываются в предложении WHERE.

Примеры

```
SELECT lname, city
```

```
FROM person
```

```
WHERE city='Москва';
```

```
SELECT * FROM person WHERE fname='Василий';
```

Использование неравенств

Оператор отношения или реляционный оператор – это математический символ обозначающий определенный тип сравнения между двумя значениями.

=	равно
>	больше, чем
<	меньше, чем
>=	больше или равно
<=	меньше или равно
<>	не равно

Пример

```
SELECT * FROM person WHERE year>20;
```

Логические операторы

Логические операторы или булевы операторы – выражения, относительно которых можно сказать: истинно оно или ложно.

В большинстве компьютерных языков результатом логического выражения является TRUE или FALSE. В SQL допустимо третье значение – UNKNOWN.

AND логическое «И» - принимает в качестве аргументов 2 логических выражения и выдает TRUE, если оба истинны.

OR логическое «ИЛИ» - принимает в качестве аргументов 2 логических выражения и выдает TRUE, если хотя бы одно истинно.

NOT логическое «НЕ» - принимает в качестве аргумента единственное логическое выражение и инвертирует его значение.

Таблица истинности для AND

AND показывает строки, которые удовлетворяют одновременно обоим условиям.

AND	TRUE	FALSE	UNKNOWN
TRUE	TRUE	FALSE	UNKNOWN
FALSE	FALSE	FALSE	FALSE
UNKNOWN	UNKNOWN	FALSE	UNKNOWN

Пример

```
SELECT * FROM person  
WHERE city='Москва'  
AND year>30;
```

Таблица истинности для OR

OR показывает строки, которые удовлетворяют хотя бы одному условию.

OR	TRUE	FALSE	UNKNOWN
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	UNKNOWN
UNKNOWN	TRUE	UNKNOWN	UNKNOWN

Пример

```
SELECT * FROM person  
WHERE city='Москва'  
OR year>30;
```

Таблица истинности для NOT

NOT инвертирует выбор.

ВЫРАЖЕНИЕ	ЗНАЧЕНИЕ
NOT TRUE	FALSE
NOT FALSE	TRUE
NOT UNKNOWN	UNKNOWN

Пример

```
SELECT * FROM person  
WHERE NOT city='Москва';
```

Использование логических операторов

Логические операторы можно комбинировать.

Пример

```
SELECT * FROM person WHERE NOT city='Москва' AND mf<>'муж';
```

NOT применяется только для к выражению следующему непосредственно за оператором.

Примеры

```
SELECT * FROM person WHERE NOT city='Москва' AND mf='муж';
```

```
SELECT * FROM person WHERE NOT city='Москва' AND NOT mf='муж';
```

Для изменения последовательности вычисления логических выражений используются скобки – ().

Пример

```
SELECT * FROM person WHERE NOT (city='Москва' AND mf='муж');
```

Вопросы



Домашнее задание

- ☐ Изучить главы 5, 6 и 7 книги М.Грабера Введение в SQL
- ☐ Подготовиться к контрольной работе по лекции
- ☐ Выполнить лабораторную работу №1

Контроль

Для выполнения контрольного теста используем ссылку

<http://app.startexam.com/Center/Web/student>

Пароль на тест **5428**