Базы данных

8 лк, 8 пз, контр.раб., экз.

ст.преп. Березенцева Т.Н.

Тема 1

ВВЕДЕНИЕ В STRUCTURED QUERY LANGUAGE (SQL)

История версий стандарта SQL

Начало 70-х годов компания IBM – экспериментальная СУБД System R на основе языка SEQUEL

1981 г. – SQL/DS

1987 г. – ANSI принят 1-й стандарт SQL (SQL level 1 или SQL-86 или 87)

1989 г. – SQL level 2 или SQL-89

1992 г. – ANSI SQL-92 или просто SQL2 (значительные **изменения**)

История версий стандарта SQL

1999 г. – SQL3 или SQL:1999

2003 г. – SQL: 2003

2006 г. – SQL: 2006

2008 г. – SQL: 2008

Подмножества языка SQL

- **SQL-DDL (Data Definition Language)** – язык определения структур баз данных
- **SQL-DML (Data Manipulation Language)** – язык манипулирования данными: добавление, изменение, удаление и извлечение данных

Подмножества языка SQL

- **SQL-DCL** (**Data Control Language**) – язык определения доступа к данным
- **SQL-TCL** (**Transaction Control Language**) – язык управления транзакциями

Тема 2

ТИПЫ ДАННЫХ SQL

Символьные типы данных

CHAR или **CHAR(n)** – символьные строки фиксированной длины

VARCHAR(n) – символьная строка переменной длины

Целые типы данных

INTEGER или **INT** – целое

SMALLINT – короткое целое

Вещественные типы данных

FLOAT и **SMALLFLOAT** – числа с плавающей точкой

DECIMAL(p) – тип данных аналогичный **FLOAT** с числом значащих цифр **p**

DECIMAL(p,n) – аналогично предыдущему

Денежные типы данных

MONEY(p,n) – всё аналогично типу **DECIMAL(p,n)**

Тип данных Дата и время

DATE – для хранения даты

TIME – для хранения времени

INTERVAL – для хранения временного интервала

DATETIME – для хранения моментов времени

Последовательные типы данных

SERIAL – тип данных на основе **INTEGER**

Двоичные типы данных

BINARY

BYTE

BLOB

Тема 3

ОПЕРАТОРЫ DDL ДЛЯ СОЗДАНИЯ СХЕМЫ БАЗЫ ДАННЫХ

Операторы базы данных

CREATE DATABASE <имя_базы_данных>

– создание базы данных

DROP DATABASE <имя_базы_данных>

– удаление базы данных

Оператор CREATE TABLE

```
CREATE TABLE <имя_таблицы>
  (<имя_столбца>
  <тип_столбца> [NOT NULL]
  [UNIQUE | PRIMARY KEY]
  [REFERENCES
  <имя_мастер_таблицы>
  [<имя_столбца>]] , ...)
```

Оператор создания таблицы CREATE

```
CREATE TABLE authors
   (au_id INT PRIMARY KEY,
    author VARCHAR(25) NOT NULL);
CREATE TABLE publishers
   (pub_id INT PRIMARY KEY,
    publisher VARCHAR(255) NOT
  NULL,
    url VARCHAR(255));
```

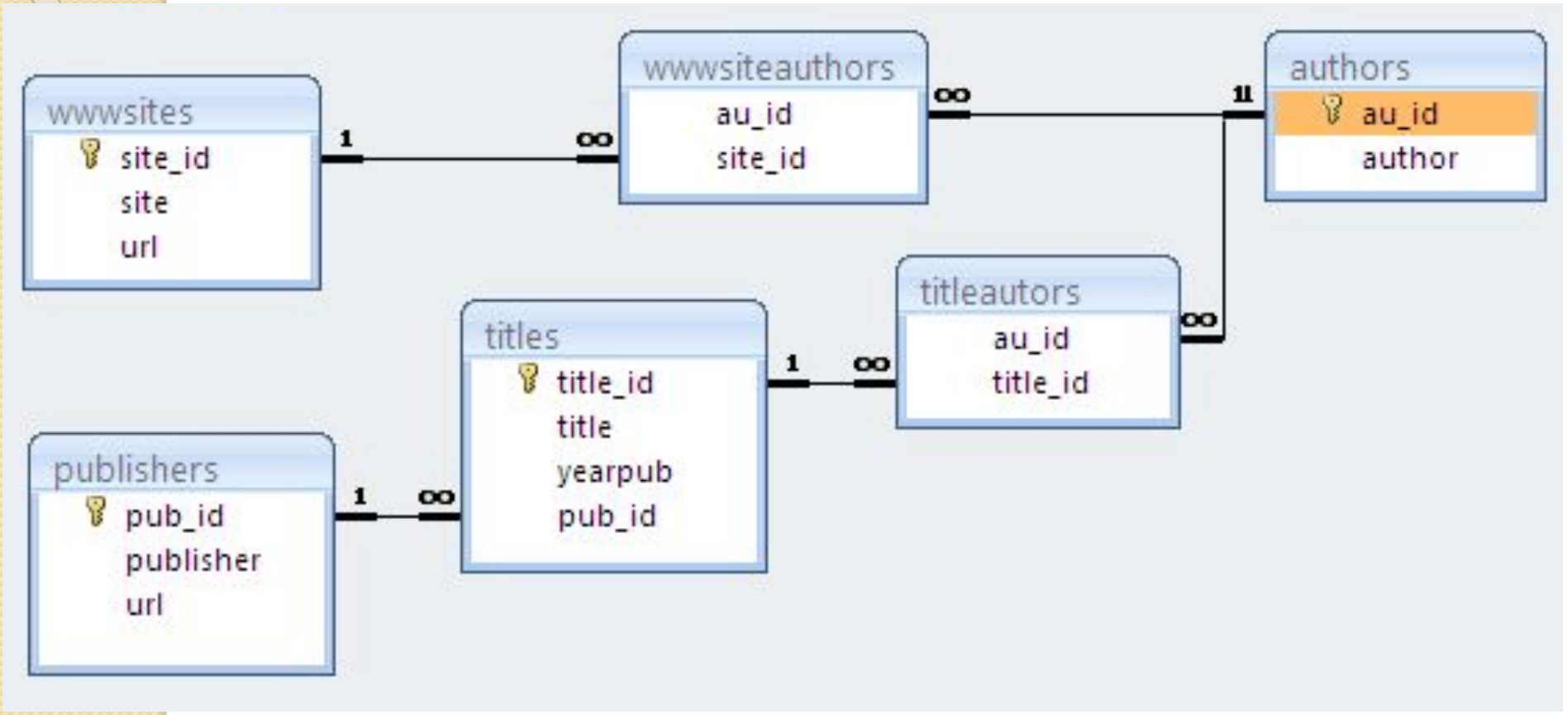
Оператор CREATE TABLE

```
CREATE TABLE titles
    (title_id INT PRIMARY KEY,
     title VARCHAR(255) NOT NULL,
     yearpub INT,
     pub_id INT REFERENCES
   publishers(pub_id));
CREATE TABLE titleautors
   (au_id INT REFERENCES authors(au_id),
   title_id INT REFERENCES titles(title_id));
```

Оператор CREATE TABLE

```
CREATE TABLE wwwsites
   (site_id INT PRIMARY KEY,
    site VARCHAR(255) NOT NULL,
    url VARCHAR(255));
CREATE TABLE wwwsiteauthors
    (au_id INT REFERENCES
  authors(au_id),
     site_id INT REFERENCES
  wwwsites(site_id));
```

Схема данных БД publications



Оператор DROP TABLE

DROP TABLE <имя_таблицы>

– удаление таблицы

Модификация таблицы ALTER

ALTER TABLE <имя_таблицы> ADD
(<имя_столбца> <тип_столбца>
[NOT NULL]
[UNIQUE | PRIMARY KEY]
[REFERENCES
<имя_мастер_таблицы>
[<имя_столбца>]]
,…) - добавить столбцы

Модификация таблицы ALTER

ALTER TABLE <имя_таблицы> DROP (<имя_столбца>,...) - удалить столбцы

Модификация таблицы ALTER

ALTER TABLE <имя_таблицы> MODIFY
(<имя_столбца> <тип_столбца>
[NOT NULL]
[UNIQUE | PRIMARY KEY]
[REFERENCES
<имя_мастер_таблицы>
<имя_столбца>]]
,…) – модификация характеристик столбцов

Тема 4

ОПЕРАТОРЫ DDL СОЗДАНИЯ ИНДЕКСОВ

Операторы создания индексов

CREATE [UNIQUE] INDEX
<имя_индекса> ON <имя_таблицы>
(<имя_столбца>,…)

Создание индексов для БД publications

```
CREATE INDEX au_names ON authors
  (author);
CREATE INDEX au_index ON authors
  (au_id);
CREATE INDEX title_index ON titles
  (title_id);
CREATE INDEX pub_index ON publishers
  (pub_id);
CREATE INDEX site_index ON wwwsites
  (site_id);
```

Оператор удаления индекса

DROP INDEX <имя_индекса>

Тема 5

ОПЕРАТОРЫ DCL ДЛЯ УПРАВЛЕНИЯ ПРАВАМИ ДОСТУПА

Оператор передачи прав на таблицу

GRANT <тип_права_на_таблицу>

ON <имя_таблицы>

[<список_столбцов>]

TO <имя_пользователя>

Тип права на таблицу

- **SELECT** – получение информации из таблицы
- **UPDATE** – изменение информации в таблице
- **INSERT** – добавление записей в таблицу
- **DELETE** – удаление записей из таблицы
- **INDEX** – индексирование таблицы
- **ALTER** – изменение схемы определения таблицы
- **ALL** – все права

Примеры

GRANT ALL ON publishers TO andy;

GRANT SELECT INSERT ON publishers TO peter;

GRANT SELECT ON publishers TO PUBLIC;

Оператор отмены прав на таблицу

REVOKE <тип_права_на_таблицу>
ON <имя_таблицы>
[<список_столбцов>] FROM
<имя_пользователя>

Назначение привилегий на БД в целом

GRANT

<тип_права_на_базу_данных> ON

<имя_базы данных>

TO <имя_пользователя>

Пример списка прав на БД

поддерживаемых СУБД *Informix:*

CONNECT

RESOURCE

DBA

Отмена прав на БД в целом

REVOKE
<тип_права_на_базу_данных>
FROM <имя_пользователя>

Тема 6

ОПЕРАТОРЫ DML ДЛЯ МОДИФИКАЦИИ ДАННЫХ

Добавление новой записи в таблицу

INSERT INTO <имя_таблицы>

[

(<имя_столбца>,<имя_столбца>,...)

]VALUES

(<значение>,<значение>,..)

Примеры

```
INSERT INTO publishers
  VALUES (16, "Microsoft
  Press","http://www.microsoft.com");

INSERT INTO publishers
   (publisher, pub_id)
  VALUES ("Super Computer
  Publishing",17);
```

Модификация записей

UPDATE <имя_таблицы>

SET <имя_столбца>=<значение>,…

[WHERE <условие>]

Например:

UPDATE publishers SET
url="http://www.superpub.com" WHERE
pub_id=17;

Операции допустимые в WHERE

- > , < , >= , <= , = , <> , !=
- проверки поля на нулевое значение: IS NULL, IS NOT NULL
- проверки на вхождение в диапазон: BETWEEN и NOT BETWEEN
- проверки на вхождение в список: IN и NOT IN
- проверки на вхождение подстроки: LIKE и NOT LIKE
- отдельные операции соединяются связями AND, OR, NOT и группируются с помощью скобок.

Пример

```
UPDATE publishers
   SET url="url not defined"
    WHERE url IS NULL;
```

Удаление записей

DELETE FROM <имя_таблицы>
[WHERE <условие>]

Например:

```
DELETE FROM publishers WHERE
  publisher = "Super Computer
  Publishing";
```

Тема 7

ОПЕРАТОР DML ДЛЯ ВЫБОРКИ ДАННЫХ

Оператор **SELECT** (полный синтаксис)

```
SELECT [ALL | DISTINCT]
   <список_выбора>
   FROM <имя_таблицы>, …
   [ WHERE <условие> ]
   [ GROUP BY <имя_столбца>,… ]
   [ HAVING <условие> ]
   [ORDER BY <имя_столбца>
   [ASC | DESC],… ]
```

Использование **ALL** и **DISTINCT**

- **DISTINCT** – исключает повторяющиеся записи из выборки,
- **ALL** указывает, что в результат необходимо включать все строки.

Примеры

```
SELECT author FROM authors;
SELECT * FROM authors;
SELECT title FROM titles
    WHERE yearpub>=1995 AND
     yearpub<=1997;
```

Примеры

```
SELECT title FROM titles WHERE
  yearpub BETWEEN 1995 AND 1997;
SELECT title FROM titles WHERE
  yearpub IN (1995,1996,1997);
```

Пример вложенных запросов

```
SELECT title FROM titles
    WHERE pub_id
    IN
   (SELECT pub_id
        FROM publishers
        WHERE publisher='Oracle Press');
```

Использование LIKE

WHERE <имя_столбца> LIKE <образец>
[ESCAPE <ключевой_символ>]

В шаблонах используются два символа:

- % (знак процента) – заменяет любое количество символов,
- _ (подчеркивание) – заменяет одиночный символ.

Использования LIKE

```
SELECT publisher, url FROM publishers
  WHERE publisher LIKE '%Wiley%';

SELECT title FROM titles WHERE title LIKE
  'SQL%';

SELECT site, url FROM wwwsites WHERE
  url LIKE '%my@_works%' ESCAPE '@';
```

Тема 8

ВЫБОРКА ИЗ НЕСКОЛЬКИХ ТАБЛИЦ

В операторе SELECT после ключевого слова **FROM** указывается список таблиц, по которым производится поиск данных.

После ключевого слова **WHERE** указывается условие, по которому производится слияние

Пример

```
SELECT titles.title, titles.yearpub,
  publishers.publisher
   FROM titles, publishers
   WHERE
  titles.pub_id=publishers.pub_id;
```

Пример

```
SELECT titles.title, titles.yearpub,
  publishers.publisher
  FROM titles,publishers
  WHERE titles.pub_id=publishers.pub_id
    AND titles.yearpub>1996;
```

Пример

```
SELECT authors.author,
  titles.title,titles.yearpub,
  publishers.publisher
 FROM titles, publishers, titleauthors
 WHERE
  titleauthors.au_id=authors.au_id AND
  titleauthors.title_id=titles.title_id AND
  titles.pub_id=publishers.pub_id AND
  titles.yearpub > 1996;
```

Тема 9

ВЫЧИСЛЕНИЯ ВНУТРИ SELECT

В конструкции <список_выбора> можно использовать константы, функции и их комбинации с арифметическими операциями и скобками.

Пример

```
SELECT title, yearpub-1986
  FROM titles
  WHERE yearpub > 1986;
```

В арифметических выражениях

допускаются операции:

- сложения (+),
- вычитания (-),
- деления (/),
- умножения (*),
- а также различные функции (COS, SIN, ABS и др.).

Также в запрос можно добавить строковую константу

Пример

```
SELECT 'the title of the book is',  title,
   yearpub-1992
  FROM titles
   WHERE yearpub > 1992;
```

Агрегатные функции

- **AVG(<имя поля>)** – среднее по всем значениям данного поля
- **COUNT(<имя поля>)** или **COUNT (*)** – число записей
- **MAX(<имя поля>)** – максимальное из всех значений данного поля
- **MIN(<имя поля>)** – минимальное из всех значений данного поля
- **SUM(<имя поля>)** – сумма всех значений данного поля

Пример

```
SELECT MIN(yearpub) FROM titles;
```

Пример

```
SELECT COUNT(*) FROM titles
  WHERE title LIKE '%SQL%';
```

Тема 10

ГРУППИРОВКА ДАННЫХ В ОПЕРАТОРЕ SELECT

GROUP BY <имя_столбца>,…

HAVING <условие>

Пример

```
SELECT publishers.publisher,
  count(titles.title)
 FROM titles, publishers
 WHERE
  titles.pub_id=publishers.pub_id
 GROUP BY publisher;
```

Пример

```
SELECT publishers.publisher,
  count(titles.title)
 FROM titles, publishers
 WHERE
  titles.pub_id=publishers.pub_id
 GROUP BY publisher
 HAVING COUNT(*)>1;
```

Пример

```
SELECT publishers.publisher,
  count(titles.title)
 FROM titles,publishers
 WHERE
  titles.pub_id=publishers.pub_id
 GROUP BY publisher
 HAVING publisher LIKE '%Press';
```

Тема 11

СОРТИРОВКА ДАННЫХ В ОПЕРАТОРЕ SELECT

ORDER BY <список_выбора>

- по возрастанию ASC или
- по убыванию DESC

Пример

```
SELECT author FROM authors
  ORDER BY author;
```

Пример

```
SELECT authors.author,
  titles.title,titles.yearpub,
  publishers.publisher
 FROM authors,titles, publishers,
  titleauthors WHERE
  titleauthors.au_id=authors.au_id AND
  titleauthors.title_id=titles.title_id AND
  titles.pub_id=publishers.pub_id
 ORDER BY authors.author ASC,
  titles.yearpub DESC;
```

Тема 12

ОПЕРАЦИЯ ОБЪЕДИНЕНИЯ

UNION

В SQL предусмотрена возможность выполнения операции реляционной алгебры "ОБЪЕДИНЕНИЕ" (UNION) над отношениями, являющимися результатами оператора SELECT.

Эти отношения должны быть определены по одной схеме.

Пример

```
SELECT publisher, url FROM publishers
    UNION
    SELECT site, url FROM wwwsites;
```

Тема 13

ИСПОЛЬЗОВАНИЕ ПРЕДСТАВЛЕНИЙ

Определение

Представление (view) – это таблица, содержимое которой берется из других таблиц посредством запроса.

Представления автоматически изменяются при изменении содержимого базовых таблиц

CREATE VIEW

CREATE VIEW
<имя_представления>
[<имя_столбца>,...]
AS <запрос>

Ограничения

- представление должно базироваться на единственном запросе (UNION не допустимо)
- выходные данные запроса, формирующего представление, должны быть не упорядочены (ORDER BY не допустимо)

Пример

```
CREATE  VIEW books AS SELECT
  authors.author, titles.title,
  titles.yearpub, publishers.publisher
  FROM authors, titles, publishers,
  titleauthors
  WHERE
  titleauthors.au_id=authors.au_id AND
  titleauthors.title_id=titles.title_id AND
  titles.pub_id=publishers.pub_id
```

Пример

Из представления можно осуществлять выборку данных

```
SELECT titles
    FROM books
    WHERE author LIKE '%Date‘
SELECT author, count(title)
    FROM books
    GROUP BY author
```

Преимущества представлений

- Сокращение времени выборки из нескольких таблиц
- Использование вычисляемых полей (столбцов)

Пример

```
CREATE VIEW amount (publisher,
  books_count)
  AS SELECT publishers.publisher,
  count(titles.title)
        FROM titles, publishers
       WHERE
           titles.pub_id=publishers.pub_id
       GROUP BY publisher;
```

Ограничения на изменение данных

- Если представление основано на одной таблице, изменения данных в нем допускаются. При этом изменяются данные в связанной с ним таблице.
- Если представление основано более чем на одной таблице, то изменения данных в нем не допускаются, т.к. в большинстве случаев СУБД не может правильно восстановить схему базовых таблиц из схемы представления.

Удаление представления

DROP VIEW <имя_представления>

Тема 14

ДРУГИЕ ВОЗМОЖНОСТИ SQL

Хранимые процедуры

- **процедурные** расширения SQL (например, PL/SQL компании Oracle и т. д.), содержащие логические операторы (IF ... THEN ... ELSE), операторы перехода по условию (SWITCH ... CASE ...), операторы циклов (FOR, WHILE, UNTIL) и операторы передачи управления в процедуры (CALL, RETURN)

Определение

Хранимые процедуры – это функциональные модули хранящиеся на сервере вместе с базой данных.

Могут быть вызваны с передачей параметров любым пользователем, имеющим на то соответствующие права.

Могут быть реализованы в виде внешних по отношению к СУБД модулей на языках общего назначения, таких как *C* или *Pascal*.

Пример для СУБД PostgreSQL

CREATE FUNCTION <имя_функции>
([<тип_параметра1>,…<тип_параметра2>])
RETURNS <возвращаемые_типы>
AS [<SQL_оператор> | <имя_объектного_модуля>]
LANGUAGE 'SQL' | 'C' | 'internal'

Триггеры

– это хранимая процедура без параметров, которая вызывается при выполнении оператора модификации этой таблицы (INSERT, UPDATE, DELETE).

Выполняются автоматически, независимо от причины модификации данных – действия человека-оператора или прикладной программы.

Синтаксис («усредненный")

```
CREATE  TRIGGER <имя_триггера>
  ON <имя_таблицы>
 FOR { INSERT | UPDATE | DELETE }
  [, INSERT | UPDATE | DELETE ] …
  AS <SQL_оператор>
```

Мониторы событий

- хранимые процедуры, которые непрерывно сканируют одну или несколько таблиц на предмет обнаружения тех или иных событий

Пример

- с токарным станком