Structured Query Language

Год	Название	Иное название	Изменения
1986	SQL-86	SQL-87	Первый вариант стандарта, принятый институтом ANSI и одобренный ISO в 1987 году.
1989	SQL-89	FIPS 127-1	Немного доработанный вариант предыдущего стандарта.
1992	SQL-92	SQL2, FIPS 127-2	Значительные изменения (ISO 9075); уровень *Entry Level* стандарта SQL-92 был принят как стандарт FIPS 127-2.
1999	SQL:1999	SQL3	Добавлена поддержка регулярных выражений, рекурсивных запросов, поддержка триггеров, базовые процедурные расширения, нескалярные типы данных и некоторые объектно-ориентированные возможности.
2003	SQL:2003		Введены расширения для работы с XML-данными, оконные функции (применяемые для работы с OLAP-базами данных), генераторы последовательностей и основанные на них типы данных.
2006	SQL:2006		Функциональность работы с XML-данными значительно расширена. Появилась возможность совместно использовать в запросах SQL и XQuery.
2008	SQL:2008		Улучшены возможности оконных функций, устранены некоторые неоднозначности стандарта SQL:2003

Подмножества команд SQL

операторы манипуляции данными (Data Manipulation Language, **DML**):

- SELECT считывает данные, удовлетворяющие заданным условиям,
- INSERT добавляет новые данные,
- UPDATE изменяет существующие данные,
- DELETE удаляет данные;

операторы определения данных (Data Definition Language, **DDL**):

- CREATE создает объект БД (саму базу, таблицу, представление, пользователя и т. д.),
- ALTER изменяет объект,
- DROP удаляет объект;

операторы управления транзакциями (Transaction Control Language, **TCL**):

- COMMIT применяет транзакцию,
- ROLLBACK откатывает все изменения, сделанные в контексте текущей транзакции,
- SAVEPOINT делит транзакцию на более мелкие участки.

операторы определения доступа к данным (Data Control Language, **DCL**):

- GRANT предоставляет пользователю (группе) разрешения на определенные операции с объектом,
- REVOKE отзывает ранее выданные разрешения,
- DENY задает запрет, имеющий приоритет над разрешением;

SELECT

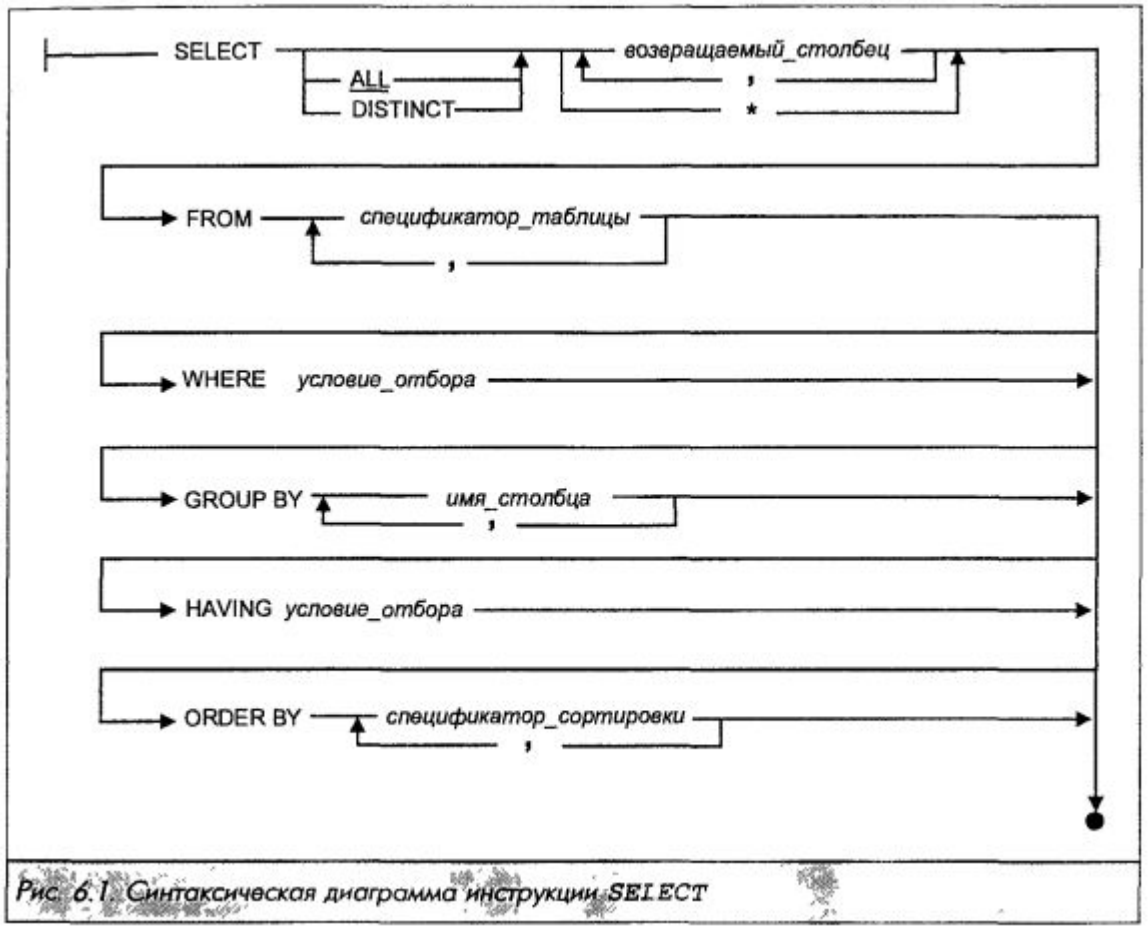

Рис. 6.1. Синтаксическая диаграмма инструкции SELECT

БД для примеров

foo
id
bar
name

item
id
foo_id
price

SELECT

```
SELECT * FROM foo;

SELECT foo.bar FROM foo;

SELECT foo.bar FROM foo WHERE foo.id = 1;
```

Условия в SQL

- AND, OR, NOT
- >, <, <=, >=, =, !=, <>
- IS NULL, IS NOT NULL

Сложные условия в SQL

- LIKE
- BETWEEN
- IN, NOT IN
- ANY, ALL
- EXISTS
- HAVING
- CASE … WHEN … THEN … ELSE … END

Экзотические условия в SQL

- COALESCE
- NULLIF
- …

Примеры с несколькими условиями

```
SELECT foo.bar
FROM foo
WHERE foo.id = 1 OR
    (foo.name LIKE ‘%th%’ AND
    foo.bar BETWEEN 10 AND 100);


SELECT foo.bar
FROM foo
WHERE foo.id = 1 AND
    (foo.bar IS NOT NULL OR id IN (4, 8, 15, 16, 413));
```

Фильтрация результатов и псевдонимы

```
SELECT DISTINCT * FROM foo;

SELECT DISTINCT f.bar FROM foo f;

SELECT CASE
        WHEN foo.id > 3 THEN ‘yep'
        WHEN n <= 3 THEN ‘nope'
        ELSE ‘WAT' END AS result
    FROM foo
    WHERE foo.id > 9 AND foo.id < 613;
```

Троичная логика

Таблица 6.1. Таблица истинности оператора AND

AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

Таблица 6.2. Таблица истинности оператора OR

OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

Таблица 6.3 Таблица истинности оператора NOT

NOT	TRUE	FALSE	NULL
	FALSE	TRUE	NULL

1. p AND q		p		
		True	**False**	**Unknown**
q	**True**	True	False	Unknown
	False	False	False	False
	Unknown	Unknown	False	Unknown

p OR q		p		
		True	**False**	**Unknown**
q	**True**	True	True	True
	False	True	False	Unknown
	Unknown	True	Unknown	Unknown

q	NOT q
True	False
False	True
Unknown	Unknown

p = q		p		
		True	**False**	**Unknown**
q	**True**	True	False	Unknown
	False	False	True	Unknown
	Unknown	Unknown	Unknown	Unknown

Схемы сложных условий

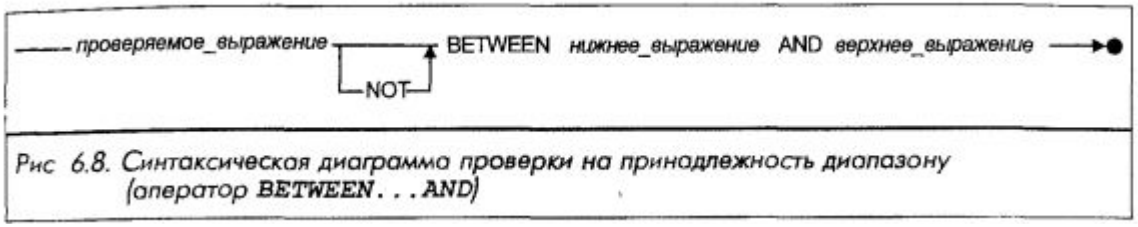

Рис 6.8. Синтаксическая диаграмма проверки на принадлежность диапазону (оператор **BETWEEN. . . AND**)

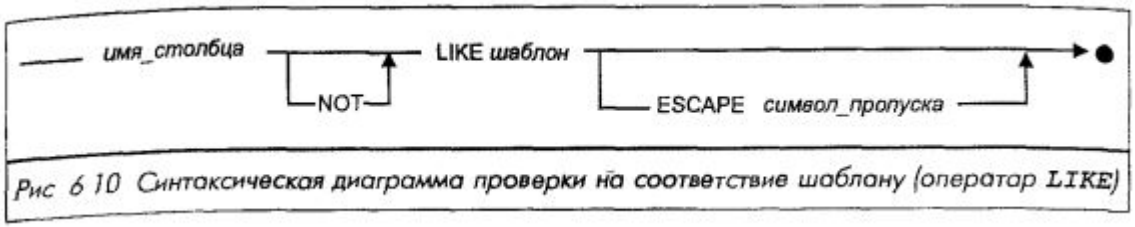

Рис 6.10 Синтаксическая диаграмма проверки на соответствие шаблону (оператор **LIKE**)

Рис 6.9 Синтаксическая диаграмма проверки на членства в множестве (оператор **IN**)

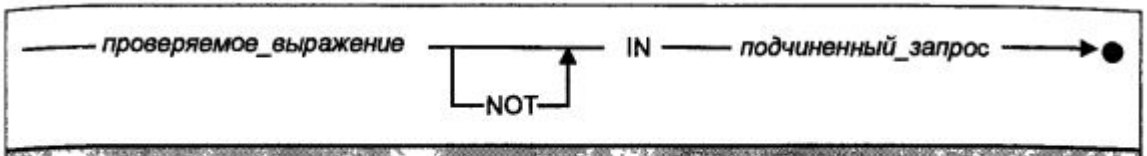

Рис. 9.4. Синтаксическая диаграмма проверки на принадлежность результатам подчиненного запроса (предикат IN)

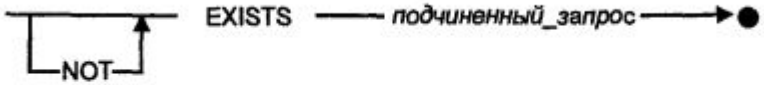

Рис. 9.5. Синтаксическая диаграмма проверки на существование (предикат EXISTS)

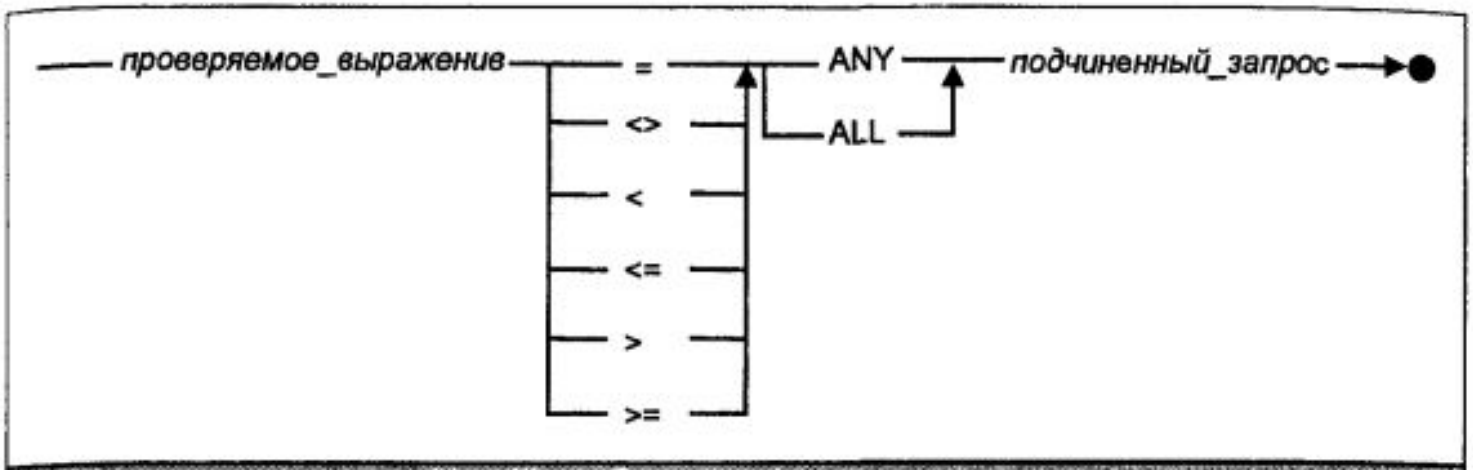

Рис. 9.6. Синтаксическая диаграмма многократного сравнения (предикаты ANY и ALL)

Рис. 9.10. Альтернативный синтаксис оператора CASE (SQL2)

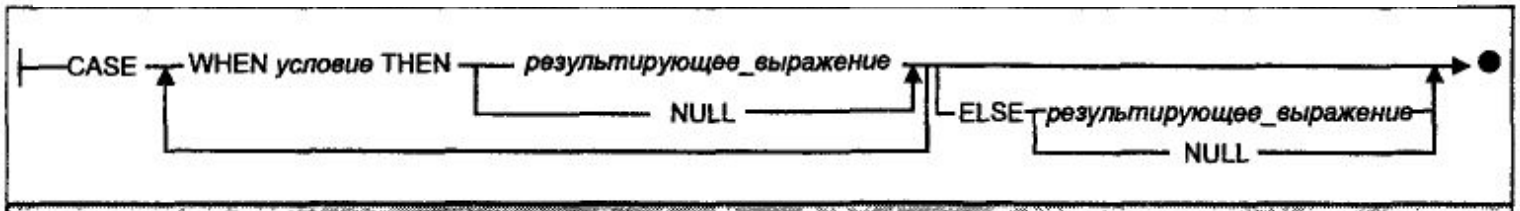

Рис. 9.8. Синтаксическая диаграмма оператора CAST (SQL2)

Сортировка выборки

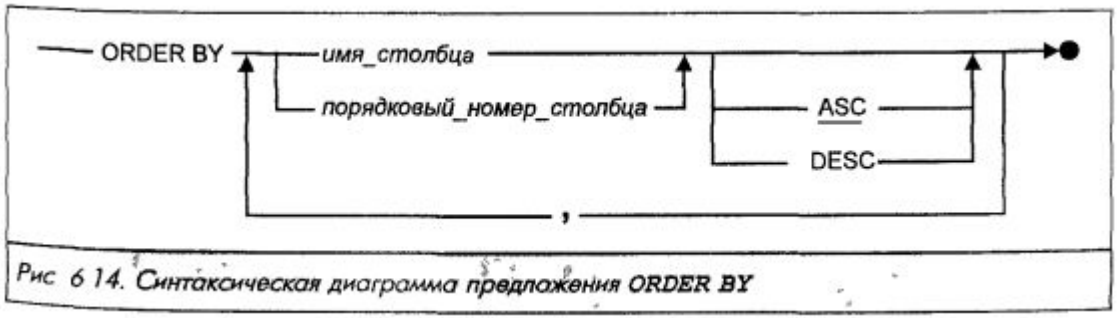

Рис 6 14. Синтаксическая диаграмма предложения ORDER BY

```
SELECT *
FROM foo f
WHERE f.id > 100
ORDER BY f.bar;

SELECT DISTINCT f.bar
FROM foo f
ORDER BY f.bar DESC;
```

Агрегатные функции

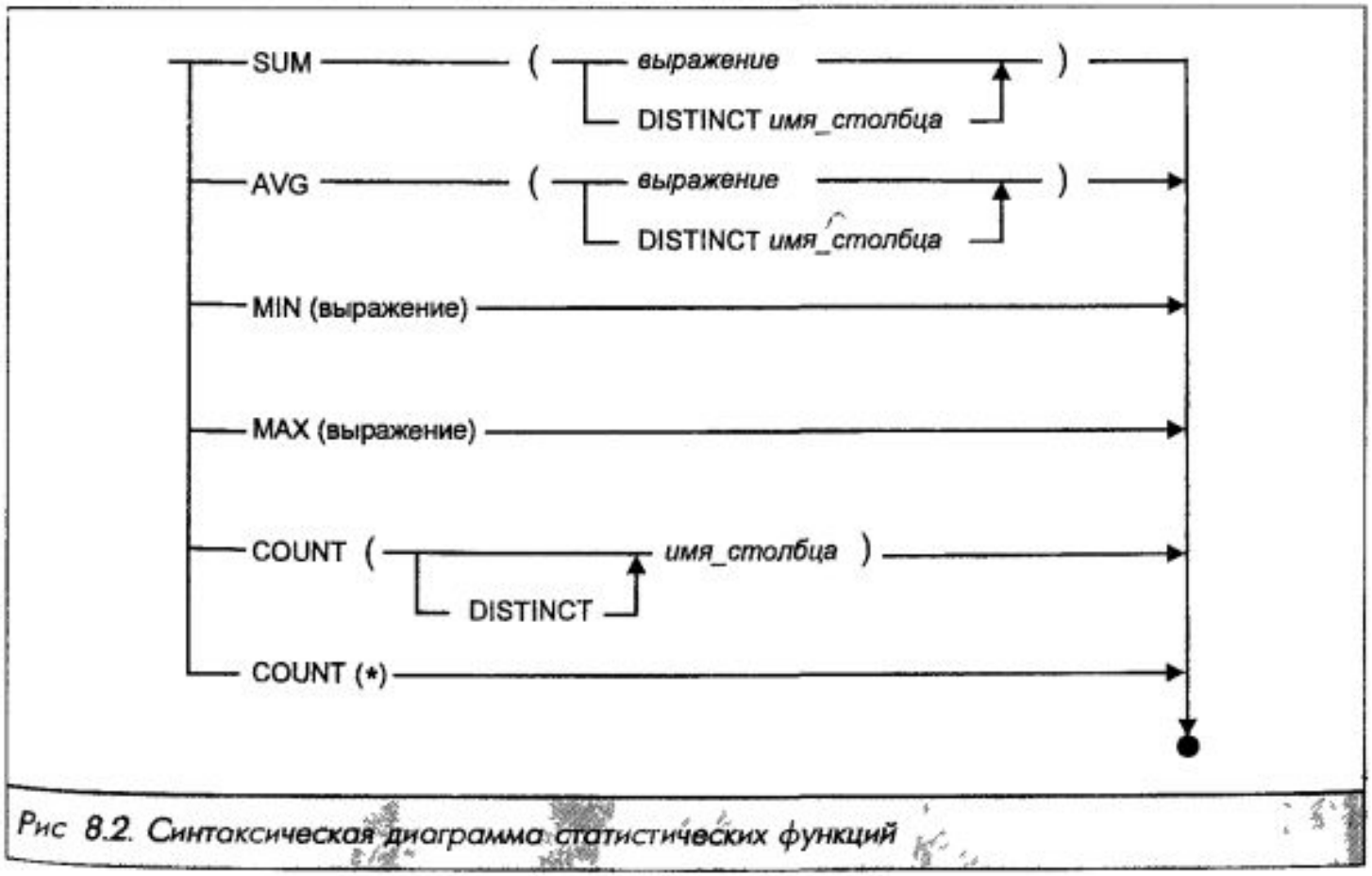

Рис 8.2. Синтаксическая диаграмма статистических функций

Агрегатные функции

```
SELECT count(*)
FROM foo f
WHERE f.id > 100;

SELECT MAX(f.id)
FROM foo f;

SELECT AVG(f.id)
FROM foo f
WHERE f.name LIKE ‘%bar’;
```

Группировка

```
SELECT f.bar “foo”, AVG(f.id) AS “average”
FROM foo f
WHERE f.name LIKE ‘%bar’
GROUP BY f.bar;
```

Объединение результатов запроса

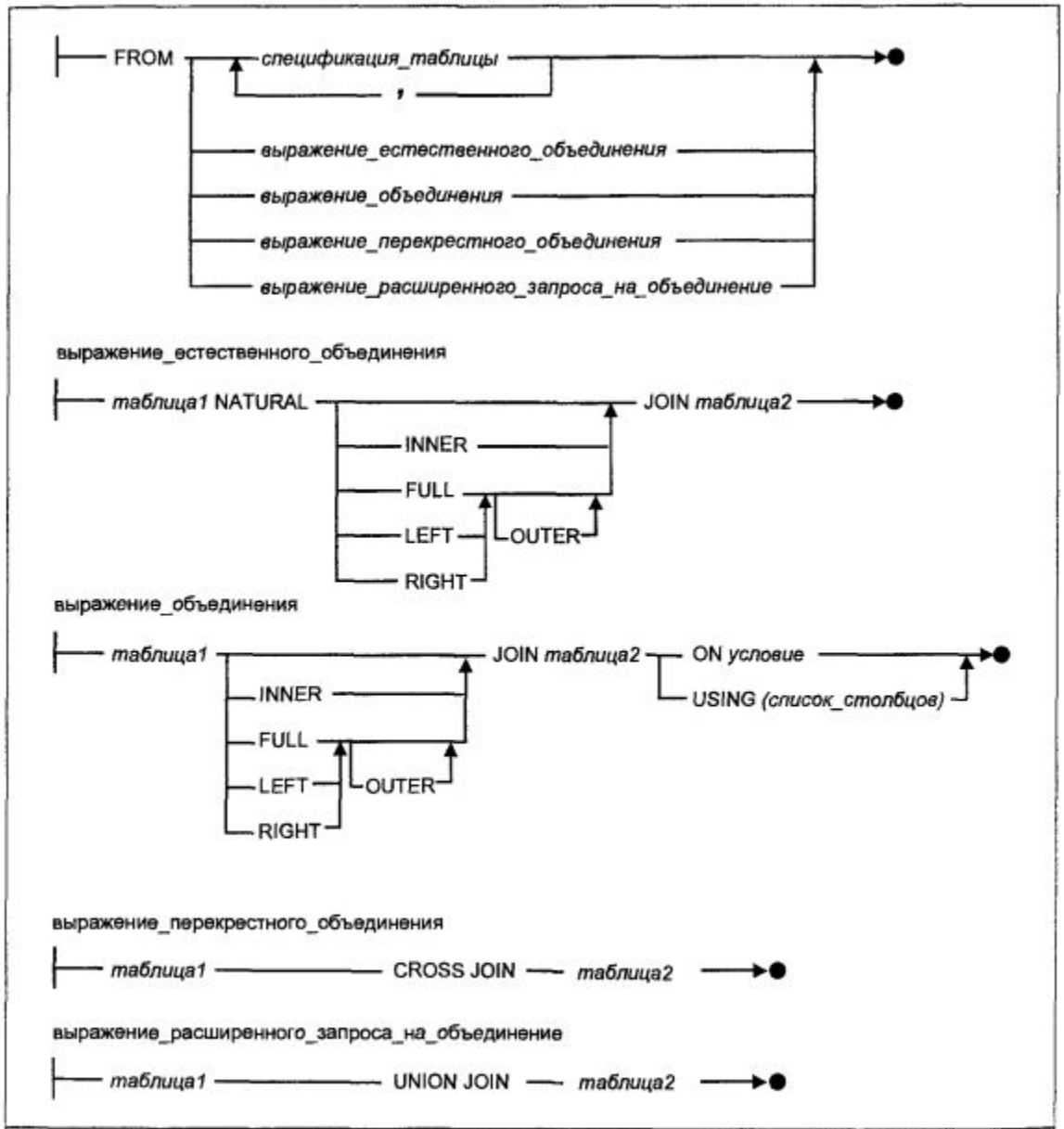

Рис 7 13 Расширенное предложение **FROM** в стандарте SQL2

Примеры объединения таблиц

```
SELECT *
FROM foo f
    INNER JOIN items i
    ON f.id = i.foo_id;

SELECT i.id, f.id, f.bar
FROM items i
    LEFT OUTER JOIN foo f
    ON f.id = i.foo_id;
WHERE i.id BETWEEN 100 AND 3011
    AND f.name IS NOT NULL;
```

Вложенные запросы

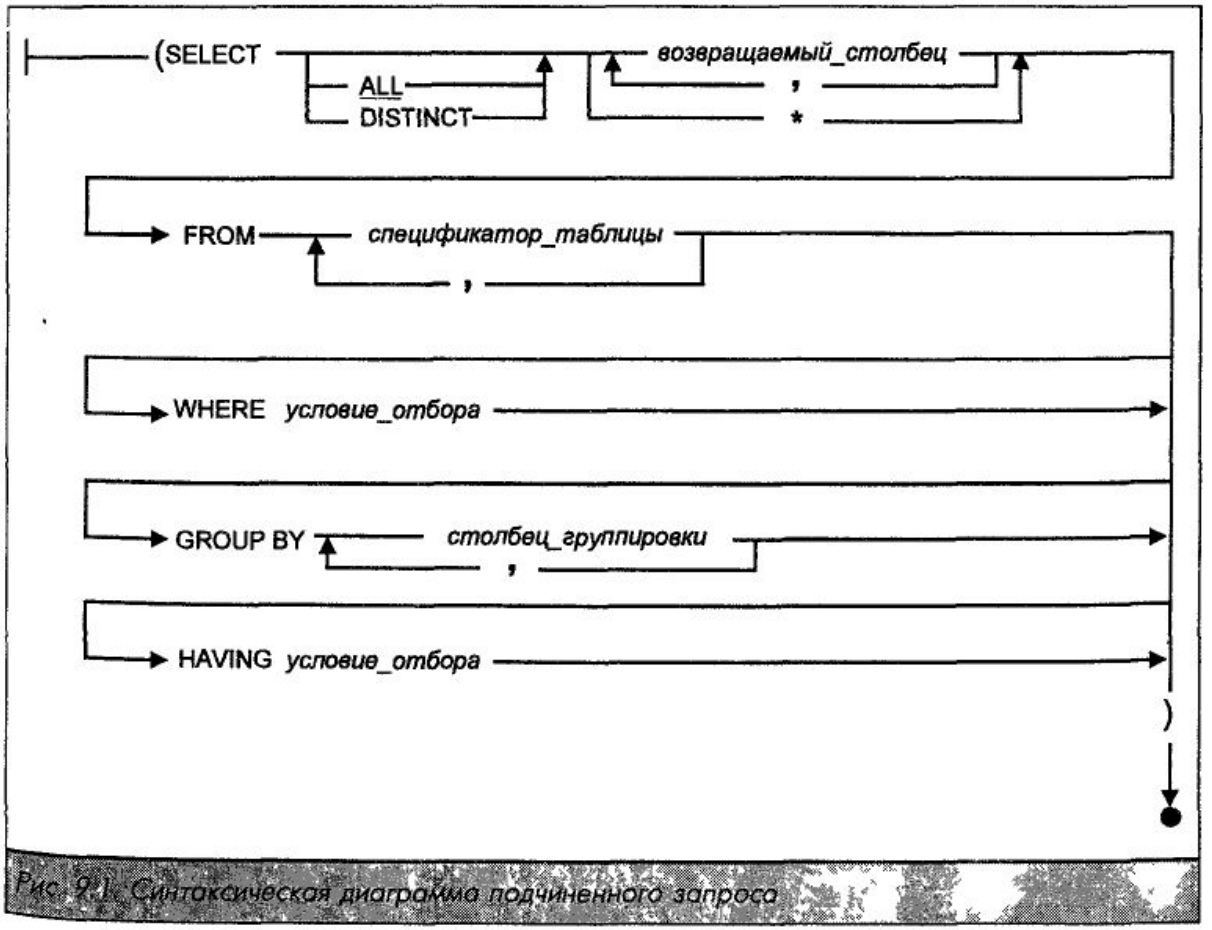

Рис. 9.1. Синтаксическая диаграмма подчиненного запроса

Примеры подзапросов

```
SELECT *
FROM items i
WHERE i.foo_id IN (
SELECT f.id
FROM foo f
WHERE f.bar = ‘foo’);

SELECT *
FROM foo f
WHERE f.id >
(SELECT MAX(i.price)
FROM item i
WHERE i.foo_id IS NOT NULL);
```

Примеры подзапросов

```
SELECT *
FROM items i
INNER JOIN
(SELECT f.id, f.bar, f.name
FROM foo f
WHERE f.name = ‘foo’) g
ON g.id = i.foo_id;
```

Предикаты ANY, ALL, EXISTS

```
SELECT *
FROM item i
WHERE i.price = ANY
(SELECT f.id
FROM foo f
WHERE f.bar > 100);

SELECT *
FROM item i
WHERE i.price > ALL
(SELECT f.id
FROM foo f
WHERE f.name IS NULL);
```

Примеры HAVING

```
SELECT SUM(i.price) FROM item I
GROUP BY i.foo_id
HAVING SUM(i.price) > 100;

SELECT SUM(i.price) FROM item I
GROUP BY i.foo_id
HAVING SUM(i.price) >
    (SELECT MAX(f.bar)
    FROM foo f
    WHERE name NOT LIKE ‘no%pe’);
```

Операторы UNION, INTERSECT, EXCEPT

```
SELECT f.id FROM foo f
WHERE f.name IS NOT NULL
UNION
SELECT i.foo_id
FROM item i;

SELECT i.foo_id
FROM item i
UNION ALL
SELECT f.id FROM foo f
WHERE f.name IS NOT NULL;
```

INSERT

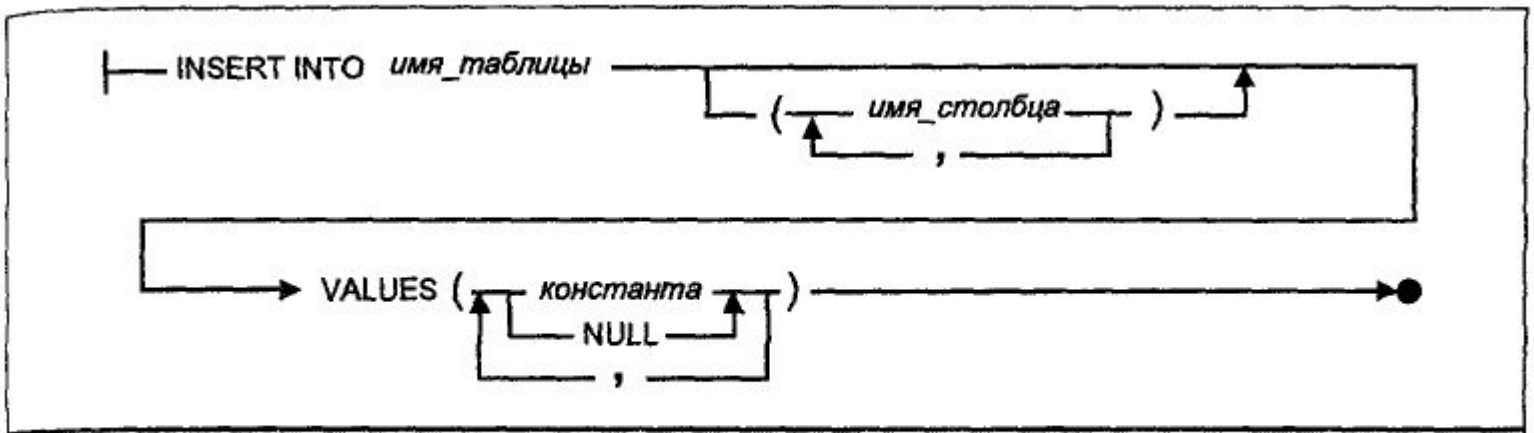

Рис 10.1 Синтаксическая диаграмма однострочной инструкции INSERT

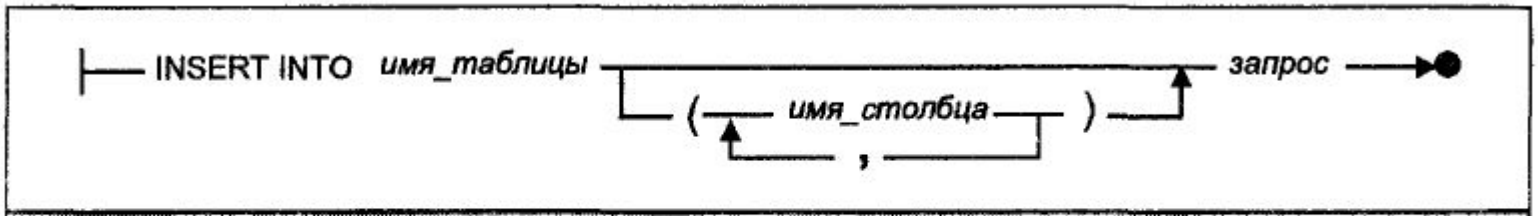

Рис. 10.3. Синтаксическая диаграмма многострочной инструкции INSERT

Примеры INSERT

INSERT INTO foo (id, name, bar) VALUES (42, ‘Nick Cage’, 100500);

INSERT INTO item (id, price) VALUES(12, -8);

INSERT INTO item VALUES (13, 42, 111);

UPDATE

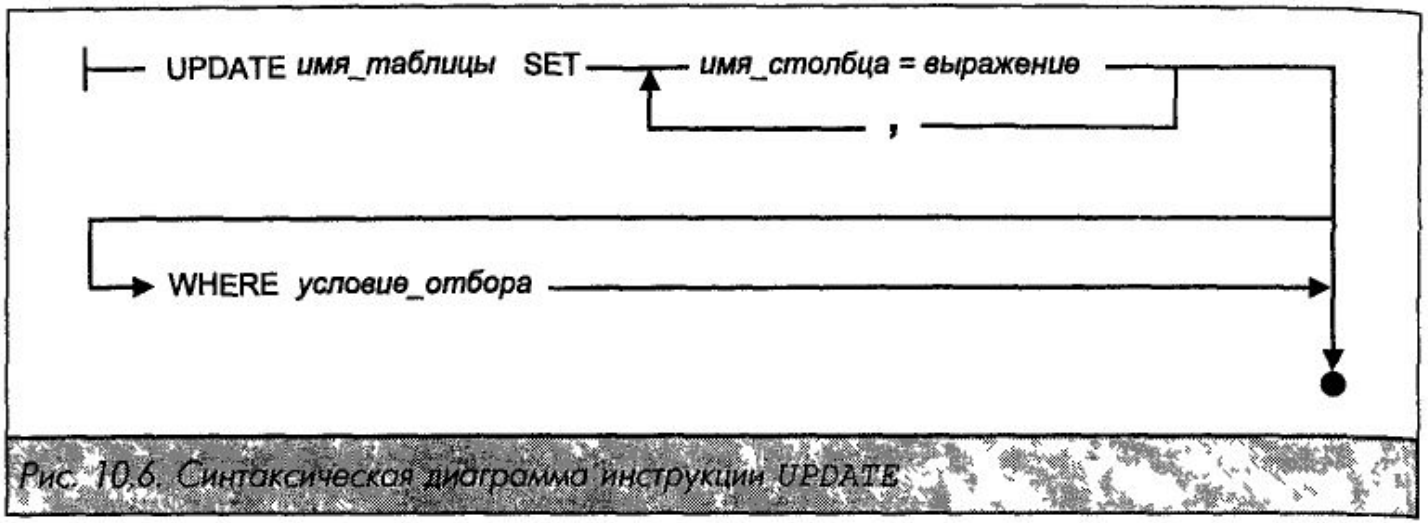

Рис. 10.6. Синтаксическая диаграмма инструкции UPDATE

```
UPDATE foo SET bar = NULL WHERE id = 42;

UPDATE item SET price = price * 2
WHERE foo_id IN (SELECT foo.id FROM
foo);
```

DELETE

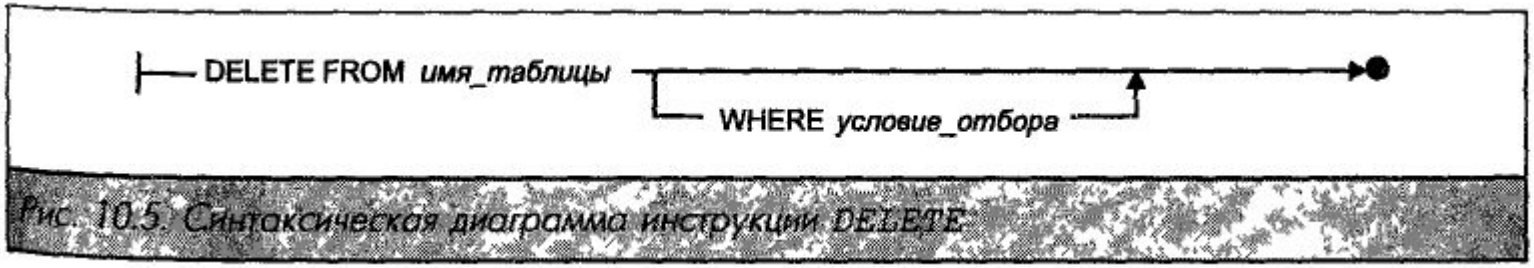

Рис. 10.5. Синтаксическая диаграмма инструкции DELETE

```
DELETE FROM foo WHERE foo.id < 100;

DELETE FROM foo
WHERE id IN
(SELECT i.foo_id FROM item i WHERE i.id > 0);

DELETE FROM item;
```

Транзакции

- BEGIN TRANSACTION; / BEGIN;
- SAVE TRANSACTION;
- COMMIT TRANSACTION; / COMMIT;
- ROLLBACK TO;
- ROLLBACK;


That's all Folks!