СУБД IBM System R

SEQUEL ***Structured English QUEry Language*** —
«структурированный английский язык запросов»

QBE (Query by Example) – «запрос по образцу»

SQL (Structured Query Language) –
язык структурированных запросов

Язык SQL

- **Интерактивный**
  - функционирование непосредственно в базе данных
- **Программный** - встраивание запросов в прикладную программу
  - **Статический SQL**
  - **Динамический SQL**
  - **API-интерфейсы**

Типы данных в SQL

- Целочисленный
- Вещественный
- Тип данных даты и времени
- Строковый

Целочисленный тип данных в SQL

Тип данных SQL	от	до
bigint	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
int	-2,147,483,648	2,147,483,647
smallint	-32,768	32,767
tinyint	0	255
bit	0	1
decimal	-10^38 +1	10^38 -1
numeric	-10^38 +1	10^38 -1
money	-922,337,203,685,477.5808	+922,337,203,685,477.5807
smallmoney	-214,748.3648	+214,748.3647

Вещественный тип данных в SQL

Тип	от	до
float	-1.79E + 308	1.79E + 308
real	-3.40E + 38	3.40E + 38

Тип данных даты и времени в SQL

Тип данных SQL	от	до
datetime	Jan 1, 1753	Dec 31, 9999
smalldatetime	Jan 1, 1900	Jun 6, 2079
date	Сохраняет дату как June 30, 1991	
time	Сохраняет время как 12:30 P.M.	

Строковый тип данных в SQL

Тип данных SQL	Описание
char	Максимальная длина 8000 символов (все значения в столбце имеют фиксированный размер, указанный при объявлении столбца). Обратите внимание: единица измерения SQL типа данных char - символ.
varchar	Максимальная длина 8000 символов (все значения в столбце имеют различный размер в зависимости от количества символов, но не более того размера, что был указан при объявлении столбца). Обратите внимание: единица измерения SQL типа данных varchar - символ.
varchar (max)	Максимальная длина 231 символ. Обратите внимание: единица измерения SQL типа данных varchar (max) - символ.
text	Максимальная длина 2,147,483,647 символов. Обратите внимание: единица измерения SQL типа данных text - символ.

Основные команды SQL

ALTER DOMAIN	CREATE CURSOR	FREE LOCATOR
ALTER TABLE	DECLARE TABLE	GET DIAGNOSTICS
CALL	DELETE	GRANT
CLOSE	DISCONNECT	HOLD LOCATOR
COMMIT	DROP ASSERTION	INSERT
CONNECT	DROP CHARACTER SET	OPEN
CREATE ASSERTION	DROP COLLATION	RELEASE SAVEPOINT
CREATE CHARACTER SET	DROP DOMAIN	RETURN
CREATE COLLATION	DROP ORDERING	REVOKE
CREATE DOMAIN	DROP ROLE	ROLLBACK
CREATE FUNCTION	DROP SCHEME	SAVEPOINT
CREATE METHOD	DROP SPECIFIC FUNCTION	SELECT
CREATE ORDERING	DROP SPECIFIC PROCEDURE	SET CONNECTION
CREATE PROCEDURE	DROP SPECIFIC ROUTINE	SET CONSTRAINTS
CREATE ROLE	DROP TABLE	SET ROLE
CREATE SCHEMA	DROP TRANSFORM	SET SESSION AUTHORIZATION
CREATE TABLE	DROP TRANSLATION	SET SESSION CHARACTERISTICS
CREATE TRANSFORM	DROP TRIGGER	SET TIME ZONE
CREATE TRANSLATION	DROP TYPE	SET TRANSACTION
CREATE TRIGGER	DROP VIEW	START TRANSACTION
CREATE TYPE	FETCH	UPDATE
CREATE VIEW		

Типы операторов SQL

- . DDL (Data Definition Language) – язык определения данных
  - CREATE
  - ALTER
  - DROP
  - RENAME

- DML (Data Manipulation Language)– язык манипуляции данными
  - INSERT
  - UPDATE
  - DELETE

Типы операторов SQL

- DQL (Data Query Language) – язык запросов к данным
  - SELECT
- DCL (Data Control Language) – язык управления данными
  - PASSWORD
  - GRANT
  - REVOKE

Типы операторов SQL

- DAL (Data Administration Language) – язык администрирования данных
  - START
  - AUDIT
  - STOP
- Команды управления транзакциями
  - SAVE
  - POINT
  - SET

Операторы SQL

CREATE TABLE (создание таблицы):

CREATE TABLE <table–name>
(<column name> <data type>[(<size>)],
<column name> <data type>[(<size>)], ...);

Операторы SQL

КодПродавца	ИмяПродавца	Город	Комиссионные
11	Браун	Лондон	0.12
12	Герман	Прага	0.13
14	Смит	Лондон	0.11
17	Алекс	Барселона	0.15
13	Джон	Нью Йорк	0.10

```
CREATE TABLE Продавцы
(КодПродавца integer,
ИмяПродавца char (10),
Город char (10),
Комиссионные real);
```

Операторы SQL

DROP TABLE (удаление таблицы):

DROP TABLE < table name >

DROP TABLE Продавцы

Введение ограничений

CREATE TABLE с учетом ограничений:

CREATE TABLE < table name >
(<column name> <data type> <column constraint>,
<column name> <data type> <column constraint> ...;

CREATE TABLE Продавцы
(КодПродавца integer NOT NULL,
ИмяПродавца char (10),
Город char (10),
Комиссионные real);

Введение ограничений

CREATE TABLE с учетом ограничений:

CREATE TABLE < table name >
(<column name> <data type> <column constraint>,
<column name> <data type> <column constraint> ...;

CREATE TABLE Продавцы
(КодПродавца integer UNIQUE,
ИмяПродавца char (10),
Город char (10),
Комиссионные real);

Введение ограничений

CREATE TABLE с учетом ограничений:

CREATE TABLE < table name >
(<column name> <data type> <column constraint>,
<column name> <data type> <column constraint> ...;

CREATE TABLE Продавцы
(КодПродавца integer NOT NULL UNIQUE,
ИмяПродавца char (10),
Город char (10),
Комиссионные real);

Создание первичных ключей

```
CREATE TABLE Продавцы
(КодПродавца integer PRIMARY KEY,
ИмяПродавца char (10),
Город char (10),
Комиссионные real);
```

Ограничение на столбец

```
CREATE TABLE Заказы
(КодПокупки integer,
СуммаПокупки double,
ДатаПокупки date,
КодКлиента integer,
КодПродавца integer,
PRIMARY KEY (КодКлиента, КодПродавца));
```

Ограничение на таблицу

Создание внешних ключей

```
CREATE TABLE Заказы
(КодПокупки integer,
СуммаПокупки double,
ДатаПокупки date,
КодКлиента integer REFERENCES Клиенты (КодКлиента),
КодПродавца integer REFERENCES Продавцы (КодПродавца));
```

Ограничение на столбец

```
CREATE TABLE Заказы
(КодПокупкиinteger,
СуммаПокупки double,
ДатаПокупки date,
КодКлиента integer,
КодПродавца integer,
FOREIGN KEY (КодКлиента) REFERENCES Клиенты (КодКлиента),
FOREIGN KEY (КодПродавца) REFERENCES Продавцы (КодПродавца));
```

Ограничение на таблицу

Операторы SQL

INSERT (ввод значений):

INSERT INTO <table name >
[(column [,column] ...)]
VALUES (<value> [,<value>] ...);

INSERT INTO Продавцы
VALUES (11, 'Браун', 'Лондон', 0.12);

INSERT INTO Продавцы (ИмяПродавца, Комиссионные, КодПродавца, Город)
VALUES ('Браун',0.12, 11, 'Лондон');

Операторы SQL

DELETE (удаление строк):

DELETE FROM <table name>
[WHERE search–condition];

DELETE FROM Продавцы;

Удаление всего содержимого таблицы

DELETE FROM Продавцы
WHERE КодПродавца = 13;

Удаление строки

Операторы SQL

UPDATE (изменение значения поля):

Первый вариант:

UPDATE <table name | view name>
SET column = expression [, column = expression] ...
[WHERE search–condition]
где expression – это столбец | выражение | константа | переменная.

Второй вариант:

UPDATE <table name>
SET column = expression, ...
[FROM table–list]
[WHERE search–condition]

Операторы SQL

```
UPDATE Клиенты
SET Рейтинг = 200;
```

Изменить рейтинг всех клиентов на 200

```
UPDATE Клиенты
SET Рейтинг = 200
WHERE КодПродавца = 11;
```

Модифицирование определенных строк

```
UPDATE Продавцы
SET Комиссионные = Комиссионные * 2;
```

Использование арифметических выражений

Операторы SQL

SELECT (извлечение информации):

SELECT [DISTINCT| ALL] {* | [<СПИСОК СТОЛБЦОВ>]}
FROM <СПИСОК ТАБЛИЦ>
[WHERE <предикат-условие выборки или соединения;>]
[GROUP BY <список полей результата>]
[HAVING <предикат-условие для группы>]
[ORDER BY <список полей, по которым требуется упорядочить ВЫВОД>]

Операторы SQL

Задача. Вывести номера телефонов кафедр университета.

```
SELECT Name_kaf, Nom_telef
FROM kafedra;
```

Результат:

Name kaf	Nom telef
Физики	23-34-24
Общей математики	23-65-43
Истории	23-78-72
Графики	23-99-77
Прикладной математики	23-66-62

Операторы SQL

Задача. Вывести сведения о кафедре «Графики».

```
SELECT *
FROM kafedra
WHERE Name_kaf = 'Графики';
```

Результат:

Kod_kaf	Name_kaf	Nom_telef	Nom_Auditoria	Col_sotr	Zav_kaf
004	Графики	23-99-77	385	18	Фирсов С.С.

Операторы SQL

Задача. Вывести сведения о кафедрах университета, находящихся па первом этаже, учитывая тот факт, что номера аудиторий первого этажа лежат в диапазоне от 1 до 99.

```
SELECT * FROM kafedra
WHERE NonuAuditoria BETWEEN 1 AMD 99;
```

Результат:

Kod_kaf	Name_kaf	Nomtetef	Nom_Auditoria	Col_sotr	Zavjcaf
002	Общей математики	23-65-43	003	22	Махов
005	Прикладной	23-66-62	028	24	Ляхова

Операторы SQL

Задача. Вывести сведения о кафедрах университета в виде, отсортированном по столбцу Name_kaf в порядке возрастания.

```
SELECT *
FROM kafedra
ORDER BY Namejtaf ASC;
```

Результат:

Kod_kaf	Name_kaf	Nom_telef	Nom_Auditoria	Col_sotr	Zav__kal
004	Графики	23-99-77	385	18	Фирсов
003	Истории	23-78-72	465	16	Росс
002	Общей ма	23-65-43	003	22	Махов
005	Прикладной	23-66-62	028	24	Ляхова
001	Физики	23-34-24	132	25	Иванов Т.М.

Агрегатные функции языка SQL

COUNT — возвращает количество значений в указанном столбце;

SUM — возвращает сумму значений в указанном столбце;

AVG — возвращает усредненное значение в указанном столбце;

MIN — возвращает минимальное значение в указанном столбце;

MAX — возвращает максимальное значение в указанном столбце.

Агрегатные функции языка SQL

Задача. Определить среднее число сотрудников, работающих на кафедрах университета.

```
SELECT AVG(Col_sotr) AS avg
FROM kafedra;
```

Результат:

avg
21

Вложенные запросы SQL

Задача. Вывести список платежей, где величина единовременных начислении превысила среднее значение.

```
SELECT ФИО, Этап, Начисления
FROM r
WHERE Начисления > (SELECT avg(Начисления) FROM r);
```

Результат:

ФИО	Этап	Начисления (руб)
Просов С. М.	Этап 1	2000
Чемцов Я.Ю.	Этап 3	2000
Чемцов Я.Ю.	Этап 4	2000
Яров И.М.	Этап 4	3200

Многотабличные запросы SQL

SELECT *
FROM rl, r2;
и соответствует декартову произведению таблиц r1 и r2.

Выражение
SELECT rl.A, r2.B
FROM rl, r2;
соответствует проекции декартова произведения двух таблиц на два столбца A из таблицы r1 и B из таблицы r2.

База данных «НИР»

R1 = **(ФИО, Отдел);**
R2**= (Отдел, Этап);**
R3= **(ФИО, Этап, Начисления).**

r1

ФИО	Отдел
Семенов Т.Т.	03
Просов СМ.	03
Мехова И.И.	03
Чемцов Я.Ю.	04
Яров И.М.	04

r3

ФИО	Этап	Начисления (руб)
Семенов Т. Т.	Этап 1	1000
Просов СМ.	Этап 1	2000
Мехова И.И.	Этап 1	500
Семенов Т. Т.	Этап 2	500
Просов СМ.	Этап 2	500
Мехова И.И.	Этап 2	1000
Просов СМ.	Этап 3	1000
Мехова И.И.	Этап 3	1000
Чемцов Я.Ю.	Этап 3	2000
Чемцов Я.Ю.	Этап 4	2000
Яров И.М.	Этап 4	3000

r2

Отдел	Этап
03	Этап 1
03	Этап 2
03	Этап 3
04	Этап 3
04	Этап 4

Вложенные запросы SQL

Задача. Вывести список сотрудников отдела 03, которые участвовали в выполнении Этапа_3.

```
SELECT r3.ФИО, r3.Этап
FROM rl, r3
WHERE rl.Отдел = '03' AND
rl.ФИО = r3.ФИО AND
r.Этап = 'Этап_3';
```

Результат:

ФИО	Этап
ПросовС.М.	Этап_3
Мехова И.И.	Этап_3

База данных «Сессия»

S1 = (**ФИО, Дисциплина, Оценка**) —
содержит сведения о результатах сессии;

S2= (**ФИО, Группа**) —
содержит сведения о составе групп;

S3 = (**Группа, Дисциплина**) —
содержит перечень экзаменов, подлежащих сдаче.

База данных «Сессия»

s1

ФИО	Дисциплина	Оценка
МурС.М.	Физика	4
Цуканов Т.Т.	Физика	5
Думская М.Т.	Физика	3
Дрозд Г.Р.	Физика	4
МурС.М.	История	4
Цуканов Т.Т.	История	5
Думская М.Т.	История	3
Цуканов Т.Т.	Математика	5
Думская М.Т.	Математика	4
Дрозд Г.Р.	Математика "	5
Петрова СО.	Экономика	5
Часов И.И.	Электротехника	4
Иванова Я. С.	Электротехника	5
Крисе Р.О.	Электротехника	3
Часов И.И.	Иностр_язык	5
Иванова Я.С.	Иностр_язгк	4
Часов И.И.	Экономика	4
Иванова Я.С.	Экономика	4
Крисе Р.О.	Экономика	5
Фирсова Л.Р.	Экономика	3

s2

ФИО	Группа
МурС.М.	02-КТ-21
Цуканов Т.Т.	02-КТ-21
Думская М.Т.	02-КТ-21
Дрозд Г.Р.	02-КТ-21
Петров С.О.	02-КТ-12
Часв И.И.	02-КТ-12
Иванова Я.С.	02-КТ-12
Крисс Р.О.	02-КТ-12
Фирсова Л.Р.	02-КТ-12

s3

Группа	Дисциплина
02-КТ-21	Физика
02-КТ-21	История
02-КТ-21	Математика
02-КТ-12	Экономика
02-КТ-12	Электротехника
02-КТ-12	Иностр. язык

Вложенные запросы SQL

Задача. Вывести группы, в которых по одной дисциплине на экзаменах получено больше одной пятерки.

```
SELECT s2.Группа
FROM s1, s2
WHERE s1.ФИО = s2.ФИО AND
s1.Оценка = 5
GROUP BY s2.Группа , s1.Дисциплина
HAVING count (*)> 1;
```

Результат:

Группа
02-КТ-21
02-КТ-12

Вложенные запросы SQL

Задача. Вывести список тех студентов, кто должен был сдавать экзамен по истории, но пока еще не сдавал.

```
SELECT ФИО
FROM s2,S3
WHERE s2.Группа=s3.Группа AND
Дисциплина = 'История' AND NOT EXISTS (SELECT ФИО
FROM SI
WHERE ФИО = a.ФИО AND
Дисциплина = 'История');
```

Результат:

ФИО
Дрозд Г. Р.