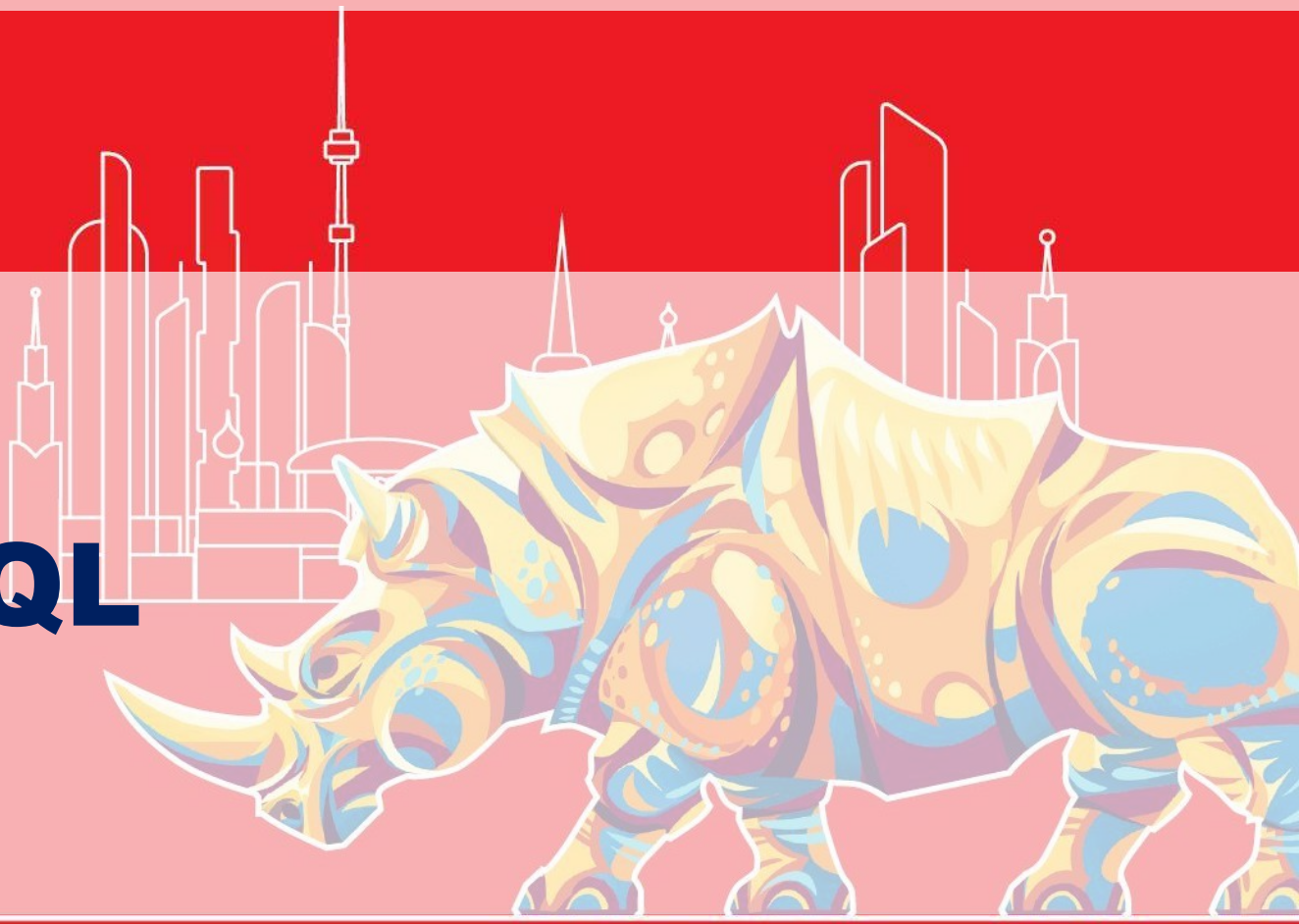
УНИВЕРСИТЕТ СИНЕРГИЯ



SQL

Лавров Сергей

Понятия

Основные понятия по управлению данными

Понятия БД и СУБД

База данных (БД)

Database (DB)

это упорядоченный набор структурированной информации

Система управления базой данных (СУБД)

Database Management System (DBMS)

это комплекс программ, позволяющих создать базу данных и манипулировать данными

Реляционная база данных

Реляционная база данных

это совокупность взаимосвязанных таблиц, содержащих информацию об объектах определенного типа

Таблица

это совокупность связанных данных, хранящихся в структурированном виде в базе данных

SQL (Structured Query Language)

Структурированный язык запросов

язык управления базами данных для реляционных баз данных.

Состав таблицы

Таблица состоит из **столбцов, строк**, а так же **ячеек** на пересечении каждого столбца и строки

Каждый **столбец** имеет уникальное имя и имеет определённый **тип данных** (число, текст, булево и т.д.)

Состав столбцов определяется при проектировании БД

Строки добавляются при использовании базы данных путём установки значений в ячейках строки.

Столбец, однозначно определяющий строку – называется **первичным ключом** (Primary Key PK)

Для связи таблиц используются **внешние ключи** (Foreign Key FK)

SQL

Основные команды SQL

SQL: Управление базой данных

Перечисленные ниже команды недоступны в MS Access

CREATE DATABASE имя_базы_данных;

Создание базы данных

DROP DATABASE имя_базы_данных;

Удаление базы данных

USE имя_базы_данных;

Выбор базы данных для использования

SQL: Управление таблицей

CREATE TABLE имя_таблицы (
имя_столбца_1 тип_столбца_1,
имя_столбца_2 тип_столбца_2,
PRIMARY KEY (имя_столбца),
FOREING KEY (имя_столбца) REFERENCES имя_другой_таблицы (имя_столбца_другой_таблицы)
);

Создание таблицы, со столбцами, а также можно указать первичный ключ и внешние ключи

RENAME TABLE старое_имя_таблицы TO новое_имя_таблицы

Переименование таблицы

DROP TABLE название_таблицы;

Удаление таблицы

SQL: Изменение таблицы

ALTER TABLE имя_таблицы ADD имя_поля тип_поля;

Добавление поля в таблицу

ALTER TABLE имя_таблицы ALTER COLUMN (имя_поля новый_тип_поля);

Изменение типа поля в таблице

ALTER TABLE имя_таблицы DROP COLUMN имя_поля;

Удаление столбца из таблицы

SQL: Изменение ключей

ALTER TABLE имя_таблицы PRIMARY KEY (имя_поля)

Добавление первичного ключа к таблице (если его не было)

ALTER TABLE имя_таблицы DROP PRIMARY KEY;

Удаление первичного ключа

ALTER TABLE имя_таблицы ADD CONSTRAINT имя_составного_ключа
PRIMARY KEY (имя_поля_1, имя_поля_2, …)

Добавление составного первичного ключа к таблице (если его не было)

ALTER TABLE имя_таблицы ADD CONSTRAINT имя_внешнего_ключа
FOREIGN KEY (имя_поля) REFERENCES имя_таблицы(имя_поля);

Добавление внешнего ключа к таблице

ALTER TABLE имя_таблицы DROP CONSTRAINT имя_ключа;

Удаление любого ключа из таблицы (ключ должно было предварительно указано)

SQL: Заполнение таблицы

INSERT INTO имя_таблицы (имя_столбца_1, имя_столбца_2, …)

VALUES (значение_1, значение_2, …);

Добавление данных в таблицу

UPDATE имя_таблицы

SET имя_столбца_1 = значение_1, имя_столбца_2 = значение_2, ...

WHERE условие;

Обновление данных в таблице по условию

DELETE FROM имя_таблицы;

Удаление всех данных из таблицы

SQL: Получение данных

SELECT имя_столбца_1, имя_столбца_2, …

FROM имя_таблицы;

Получение данных из таблицы

SELECT * FROM имя_таблицы;

Получение все данных из таблицы

SELECT DISTINCT имя_столбца_1, имя_столбца_2, …

FROM имя_таблицы;

Получение не повторяющихся данных из таблицы

SQL: Фильтрация данных

Все следующие запросы начинаются с:

SELECT имя_столбца_1, имя_столбца_2, …

FROM имя_таблицы

WHERE условие;

Получение данных с произвольным условием

WHERE имя_столбца BETWEEN значение_1 AND значение_2;

Получение данных, где значение в столбце имя_столбца находится между значение_1 и значение_2

WHERE имя_столбца IN значение_1, значение_2, …);

Получение данных, где значение в столбце имя_столбца может быть равно значение_1, или значение_2, или …

SQL: Сортировка и группировка

Все следующие запросы начинаются с:

SELECT имя_столбца_1, имя_столбца_2, …
FROM имя_таблицы

ORDER BY имя_столбца ASC|DESC;

Сортировка по возрастанию|убыванию значений в столбце

GROUP BY имя_столбца_1, имя_столбца_2, …;

Группировка значений по указанном столбцам

SQL: Агрегатные функции

При использовании группировки на остальные поля можно наложить функции:
COUNT (имя_столбца) — считает количество строк
SUM (имя_столбца) — считает сумму значений в данном столбце
AVG (имя_столбца) — считает среднее значение данного столбца
MIN (имя_столбца) — считает наименьшее значение данного столбца
MAX (имя_столбца) — считает наибольшее значение данного столбца

Так же можно делать отбор с условием на агрегатные функции

```
SELECT имя_столбца_1, имя_столбца_2, …
FROM имя_таблицы
GROUP BY имя_столбца
HAVING условие_на_значения_агрегатных_функций;
```

SQL: Соединение таблиц

Все следующие запросы начинаются с:

SELECT имя_столбца_1, имя_столбца_2, …

FROM имя_таблицы_1

INNER JOIN имя_таблицы

ON имя_таблицы_1. имя_столбца_n1 = имя_таблицы_2. имя_столбца_n2;

Внутреннее соединение (INNER писать не обязательно) – все строки, которые есть в 1й и 2й таблицах

LEFT OUTER JOIN имя_таблицы

ON имя_таблицы_1. имя_столбца_n1 = имя_таблицы_2. имя_столбца_n2;

Левостороннее соединение (OUTER писать не обязательно) – все строки из 1й таблицы, даже если их нет во 2й

RIGHT OUTER JOIN имя_таблицы

ON имя_таблицы_1. имя_столбца_n1 = имя_таблицы_2. имя_столбца_n2;

Правостороннее соединение (OUTER писать не обязательно) – все строки из 2й таблицы, даже если их нет во 1й

Примеры

Для лучшего понимания основных команд SQL

Пример: Управление базой данных

Перечисленные ниже команды недоступны в MS Access

CREATE DATABASE mydb;

Создание базы данных mydb

DROP DATABASE mydb;

Удаление базы данных mydb

USE mydb;

Выбор базы данных mydb для использования

Пример: Управление таблицей

```
CREATE TABLE students (
id INT,
groupId INT,
name CHAR(50),
PRIMARY KEY (id),
FOREING KEY (groupId) REFERENCES groups (id)
);
```

Создание таблицы students, с 3 столбцами: идентификатор студента(id), идентификатор группы(groupId), имя студента(name)

В таблице указан первичный ключ – идентификатор студента(id)

В таблице указан внешний ключ идентификатор группы(groupId), связанный с полем идентификатор(id) таблицы groups

```
RENAME TABLE students TO users;
```

Переименование таблицы students в users

```
DROP TABLE users;
```

Удаление таблицы users

Пример: Изменение таблицы

ALTER TABLE students ADD birthday DATE;

Добавление поля день рождения (birthday) с типом дата в таблицу students

ALTER TABLE students ALTER COLUMN (birthday CHAR(20));

Изменение типа поля день рождения (birthday) в таблице на строку из 20 символов

ALTER TABLE students DROP COLUMN birthday;

Удаление столбца день рождения (birthday) из таблицы students

Пример: Изменение ключей

ALTER TABLE groups PRIMARY KEY (id);

Добавление первичного ключа – идентификатор(id) к таблице groups

ALTER TABLE groups DROP PRIMARY KEY;

Удаление первичного ключа из таблицы groups

ALTER TABLE students ADD CONSTRAINT PK_id
PRIMARY KEY (id, groupId);

Добавление составного первичного ключа из идентификатора студента(id) и идентификатора группы(groupId) к таблице students

ALTER TABLE students ADD CONSTRAINT PK_id
FOREIGN KEY (groupId) REFERENCES groups(id);

Добавление внешнего ключа groupId к таблице students, связанного с полем идентификатор(id) таблицы groups

ALTER TABLE students DROP CONSTRAINT PK_id;

Удаление любого внешнего ключа PR_id из таблицы students

Пример: Заполнение таблицы

INSERT INTO students (id, groupId, name)

VALUES (355, 112, "Иванов Иван Иванович");

Добавление в таблицу students строки для студента с идентификатором 355, группой 112 и именем Иванов Иван Иванович

UPDATE students

SET id = 356, name = "Петров Петр Петрович"

WHERE id = 355;

Для строки с идентификатором студента 355 меняется идентификатор на 356 и меняется имя на Петров Петр Петрович

DELETE FROM students;

Удаление всех данных из таблицы students

Пример: Получение данных

SELECT name, groupId

FROM students;

Получение из таблицы students со столбцами: группа и имя студента

SELECT * FROM students;

Получение всех данных из таблицы students

SELECT DISTINCT groupId

FROM students;

Получение списка идентификаторов групп из таблицы студента без повторов

Пример: Фильтрация данных

```
SELECT name
FROM students
WHERE groupId = 112;
```

Получение списка всех студентов из группы 112

```
SELECT name, groupId
FROM students
WHERE groupId BETWEEN 110 AND 115;
```

Получение списка всех студентов с номерами групп из групп 110-115

```
SELECT name
FROM students
WHERE groupId IN 112, 114;
```

Получение списка всех студентов с номерами групп из групп 112 и 114

Пример: Сортировка и группировка

```
SELECT name, groupId
FROM students
ORDER BY name ASC, groupId DESC;
```

Вывод студентов и номеров групп, где имя студента упорядочено по возрастанию, а идентификаторы групп по убыванию

```
SELECT name, groupId
FROM students
GROUP BY groupId;
```

Вывод студентов и номеров групп сгруппированных по группе

Пример: Агрегатные функции

```
SELECT COUNT(id), groupId
FROM students
GROUP BY groupId;
```

Выводим список групп с количеством студентов в каждой группе

```
SELECT groupId
FROM students
GROUP BY groupId
HAVING COUNT(id) > 10;
```

Выводим список групп, в которых более 10 студентов

Пример: Соединение таблиц

```
SELECT students.name, users.phone
FROM students
INNER JOIN users
ON students.name = users.login;
```

Показывает список всех студентов, у которых заполнен телефон в таблице users

```
SELECT students.name, users.email, users.phone
FROM students
LEFT OUTER JOIN users
ON students.name = users.login;
```

Показывает список всех студентов, и у некоторых указаны адреса электронной почты и телефоны

```
SELECT students.name, users.email
FROM students
RIGHT OUTER JOIN users
ON students.name = users.login;
```

Показывает список всех адресов электронной почты, и у некоторых будет указано имя студента если нашлось соответствие