STRUCTURED QUERY LANGUAGE (SQL)

DBM 110 SQL Database Management

STRUCTURED QUERY LANGUAGE (SQL)

SQL is a standardized set of keyword commands, a kind of language, used to interact with a relational database management system RDBMS

STRUCTURED QUERY LANGUAGE (SQL)

1970 – SQL developed by IBM researchers Raymond Boyce and Donald Chamberlain

1986 – adopted by the American National Standards Institute (ANSI)

1987 – adopted by the International Standards Organization (ISO)

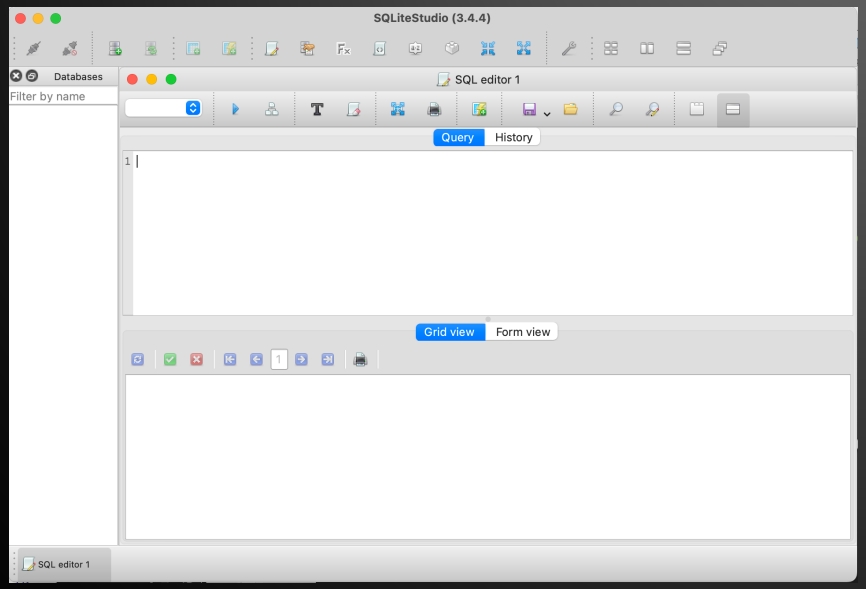
SQLite

SQLite is a C-language library that implements a full-featured, transactional SQL database engine (DBMS).

SQLite is self-contained, serverless*, public-domain and available as a library in many high-level programming languages.

** Because it is a standalone DBMS, it does not include the server-related security features of standard SQL.*

SQLite Studio



SQLite Studio is a cross-platform, open-source utility program that provides an easy-to-use interactive interface for SQLite.

BASIC SQL STATEMENTS

C CREATE TABLE – Add a table to the database

INSERT INTO – Insert data into an existing table

R SELECT FROM – Locate the data that matches the criteria

U UPDATE – Update the indicated records with new data

D DELETE FROM – Remove the indicated records from the table

WHAT IS A QUERY?

In database terminology, a Query is a means of asking a question of the database. In response, the DBMS selects data that matches the criteria set forth in the query.

Queries

SELECT <field>, <field>, …<field> FROM <tablename>;

- Select specific columns of data (fields)

SELECT * FROM <tablename>;

- Select everything from the table

Queries with Criteria

Select specific records (rows), with criteria using WHERE:

```
SELECT (<field>, <field>, …<field>) FROM <tablename>
        WHERE <criteria>;
```

A WHERE clause is used in SELECT, UPDATE, and DELETE statements to limit the operation to records that match the given criteria.

Queries with Criteria

Building Criteria:

- LIKE : an SQL keyword used for pattern matching (*wildcards:* % *or* _)
- BETWEEN : selects values in a given range
- AND, OR, NOT : logical operators used to combine conditions
- IN : is or is not IN a list of values
- +, -, /, *, %, <, <=, >, >=, = or ==, <> or != : math and comparison operators
- EXISTS : true if subquery returns some result

Sorting results:

- ORDER BY <field>, <field>, … – optional clause for SELECT

Creating Tables

```
CREATE TABLE <tablename>
        (<field> INTEGER PRIMARY KEY NOT NULL,
         <field> <datatype>,
        . . . <field> <datatype>);
```

- If you designate an INTEGER field as PRIMARY KEY, SQLite will automatically assign a unique value.
- A field designated as NOT NULL must have a value supplied.

Creating Tables

```
CREATE TABLE <table_name>
        (<field> <data_type>),
         <field> <data_type>),
         . . .,
         <field> <data_type>);
```

Usually, one of the fields is designated as the PRIMARY KEY and NOT NULL.

In SQL, a PRIMARY KEY is implicitly NOT NULL.

Adding Data

INSERT INTO <tablename> (<field>, <field>, …<field>)

VALUES (<value>, <value>, …<value>);

Any fields missing from the INSERT will be assigned default values (or NULL). If any required (NOT NULL) fields are missing, the INSERT will fail.

Updating Data

```
UPDATE <tablename>
        SET <field> = <value>,
        <field> = <value>,
        . . .
        WHERE <criteria>;
```

NOTE: If you omit the WHERE clause, all records in the table will be updated!

Deleting Data

```
DELETE FROM <tablename>
        WHERE <criteria>;
```

NOTE: If you omit the WHERE clause, all records in the table will be deleted!