find table name and comumns definintion for SQL and variences

Standard SQL:
For databases that support the ANSI SQL standard and have the `INFORMATION_SCHEMA` views available, you can use the following query:

```
SELECT table_name, column_name
FROM information_schema.columns;
```

you can append if you want to specify where

```
WHERE table_schema = 'your_database_name';
```

if you want **Database-specific Queries:** If you are working with a specific database system and the standard SQL approach doesn't work, you can try the following methods:

MySQL/MariaDB:

```
SELECT table_name, column_name
FROM information_schema.columns;
```

or

```
SHOW TABLES;
DESCRIBE table_name;
```

PostgreSQL:

```
 SELECT table_name, column_name
FROM information_schema.columns;
```

SQLite:

```
SELECT name AS table_name, sql AS column_definition
FROM sqlite_master
WHERE type = 'table';
```

You would run this SQLite command when you want to list all the tables in your SQLite database along with their SQL schema.

SQLite keeps a system table, `sqlite_master`, where it stores metadata about the database. Each row of `sqlite_master` represents an object (table, index, etc.) in the database.

The columns are:

- type: the type of the database object, such as 'table' or 'index'.
- name: the name of the object.
- tbl_name: the name of the table to which the object is associated. For a table, it's the same as `name`.
- rootpage: the page number in the database file where the root B-tree page for the object is stored.
- sql: the SQL statement that created the object.

This command specifies `type = 'table'` in the `WHERE` clause, so it only selects tables, not other types of objects like indices. For each table, it selects the name (renamed as `table_name` for clarity) and the SQL statement that created the table (as `column_definition`).

So this command is useful when you need to know the structure of all tables in your SQLite database, such as the table names and their corresponding column definitions. It's a handy tool for exploring a database when you don't have the schema in front of you or when you've inherited a database and need to understand its structure.

Revision #2
Created 2023-07-09 16:48:48 UTC by naruzkurai
Updated 2023-07-09 17:05:09 UTC by naruzkurai