DATA 514 Section 1 Worksheet 1

Lecture 1

Question 1

command: sqlite> select * from urls limit 3; output: 1|https://idp.u.washington.edu/idp/profile/SAML2/Redirect/SSO?execution=e1s2|Duo Security - Two-Factor Authentication|1|0|13356226743137677|0 2|https://idp.u.washington.edu/idp/profile/Authn/Duo/2FA/authorize?conversation=e1s2|Duo Security - Two-Factor Authentication|1|0|13356226743137677|0 3|https://api-57f2a007.duosecurity.com/oauth/v1/authorize?scope=openid&nonce=8d1707cbc02977506207eabbd8ce20f83010&response_type=code&redirect_uri=https%3A%2F%2Fidp.u.washington.edu%2Fidp%2Fprofile%2FAuthn%2FDuo%2F2FA%2Fduo-callback&client_id=DI7EYFPSWACJ9YOHOFJU&request=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJkdW9fdW5hbWUiOiJqajgwIiwic2NvcGUiOiJvcGVuaWQiLCJyZXNwb25zZV90eXBlIjoiY29kZSIsInJlZGlyZWN0X3VyaSI6Imh0dHBzOi8vaWRwLnUud2FzaGluZ3Rvbi5lZHUvaWRwL3Byb2ZpbGUvQXV0aG4vRHVvLzJGQS9kdW8tY2FsbGJhY2siLCJzdGF0ZSI6ImZmOTBkM2JlNTVjN2E1MjRmMWUwYThlYWNhMTllZmExLjY1MzE3MzMyIiwiZXhwIjoxNzExNzU2NzQyLCJjbGllbnRfaWQiOiJESTdFWUZQU1dBQ0o5WU9IT0ZKVSJ9.Upfc1bJyqQ-PjP6DZvhTeq-abuBVKWdo1-Z_EQWnKtbmkXMzTcw3Ewk09inPhydmrRnlEt2gvQP7_ip9wSIx_A|Duo Security - Two-Factor Authentication|1|0|13356226743137677|0

The output is a snapshot from a database, showing three records related to user interactions with a web-based authentication process. Each entry captures the visit details, including URLs and visit counts, indicating steps taken during a secure login procedure.

Question 2

1|https://idp.u.washington.edu/idp/profile/SAML2/Redirect/SSO?execution=e1s2|Duo Security - Two-Factor Authentication|1|0|13356226743137677|0

ID: 1 URL: https://idp.u.washington.edu/idp/profile/SAML2/Redirect/SSO?execution=e1s2 Title: Duo Security - Two-Factor Authentication Visit Count: 1 Typed Count: 0 Last Visit Time: 13356226743137677 Hidden: 0

Question 3

Field (Column): "URL"

Question 4

command: sqlite> .schema urls output: CREATE TABLE urls(id INTEGER PRIMARY KEY AUTOINCREMENT,url LONGVARCHAR,title LONGVARCHAR,visit_count INTEGER DEFAULT 0 NOT NULL,typed_count INTEGER DEFAULT 0 NOT NULL,last_visit_time INTEGER NOT NULL,hidden INTEGER DEFAULT 0 NOT NULL); CREATE INDEX urls_url_index ON urls (url);

The schema for the `urls` table provides information on both the data type and domain of its columns. It details the kinds of data each column can hold—`INTEGER` for numerical values and `LONGVARCHAR` for textual data—and specifies constraints like `PRIMARY KEY AUTOINCREMENT` for uniquely identifying each record and `NOT NULL` for mandatory fields. This combination of data types and constraints shows what data the table stores and the rules governing that data, thereby covering both the structure and integrity aspects of the database design.

Question 5:

In the provided outputs, the `id` attribute is explicitly defined as a key, indicated by the `PRIMARY KEY AUTOINCREMENT` constraint in the table's schema. This makes it the primary unique identifier for each row in the `urls` table, essential for ensuring each record is distinct and can be efficiently queried and related to data in other tables within an SQL database.

Theoretically, if we were only considering the limited context of these three rows, attributes like `visit_count` or `typed_count` could also serve as keys because they contain unique values within this small dataset. However, in the broader scope of an entire database where more records exist or are added over time, relying on `visit_count` or `typed_count` as unique identifiers becomes highly unlikely due to the natural expectation of repeated values in these fields.

Lecture 2

Question 1:

command: sqlite> .tables output: cluster_keywords downloads segment_usage
cluster_visit_duplicates downloads_slices segments
clusters downloads_url_chains urls
clusters_and_visits history_sync_metadata visit_source
content_annotations keyword_search_terms visited_links
context_annotations meta visits

The "python" for-loop would be:

```
foreach row in visits:
    if row.visit_count > 10:
        output(row.id)
```

Question 2:

The equivalent SQL statement would be:

```sql
select count(visit_count) from urls where visit_count > 1;
```

Question 3:

When using the command from question 2 with visit_count > 10 I get 0. That makes sense because I don't usually use chrome that often. when visit_count > 1 I get 4.

Question 4:

```
SELECT urls.id, urls.visit_count, visits.url, visits.visit_time
FROM urls
    JOIN visits ON urls.id = visits.url;
```

Based on the schema details provided:

Key (Primary Key):

- The `id` attribute in the `urls` table serves as the primary key. This is because it uniquely identifies each URL in the table.

Foreign Reference (Foreign Key):

- The `url` attribute in the `visits` table serves as the foreign key. It references the `id` in the `urls` table, linking each visit record to a specific URL based on its unique identifier.

Lecture 3

Question 1:

Joining the `urls` table with the `visits` table on a shared attribute creates an **inner join**. This type of join is chosen to combine rows from both tables where there's a match—specifically, where `visits.url` matches `urls.id`. It's ideal for scenarios where the analysis requires data that exists in both tables, such as examining visit patterns for URLs. An inner join ensures the results include only those records that have corresponding entries in both tables, providing a focused dataset for analysis.

Question 2:

```
SELECT urls.id, COUNT(visits.id) AS visit_count
    FROM urls
        JOIN visits ON urls.id = visits.url
    WHERE urls.visit_count > 2 AND visits.visit_duration > 120
        GROUP BY urls.id
    HAVING COUNT(visits.id) > 1
    ORDER BY visit_count DESC;
```

Results: 15|3 22|3

These results are artificial as I recently created the account for the sake of this exercise. The visits to URLs with IDs `15` and `22`, each recorded three times with significant duration, were part of a controlled test rather than organic browsing. This behavior suggests the testing focused on specific interactions with these sites, possibly to examine browser tracking or performance under predefined conditions. The pattern reflects the test's objectives and parameters rather than genuine user interest, underscoring the role of context in interpreting such data.