Class 4 Spanner
Elements of Databases
Sept 17, 2021
Announcements

Preparing for Midterm 1:

• End-of-chapter exercises (requires Sakila sample database)
• Practice SQL on Hacker Rank
• Practice SQL on Leetcode

On the horizon:

• BigQuery starting next week (no setup needed)
• Review session for Midterm 1 (week of the 4th)
Instapoll on your Spanner setup


1. Connect to the span database you created during the setup (either from UI or spanner-cli).

2. Run this query: `SELECT count(*) FROM information_schema.tables;`

3. How many tables are in the output?
# A World without Transactions

<table>
<thead>
<tr>
<th>Time</th>
<th>Client 1</th>
<th>Client 2</th>
</tr>
</thead>
</table>
| $t_0$ | UPDATE account
SET balance = balance - 100
WHERE name = 'Alice'; | | |
| $t_1$ | | SELECT name, balance
FROM account
WHERE name IN ('Alice', 'Bob'); |
| $t_2$ | UPDATE account
SET balance = balance + 100
WHERE name = 'Bob'; | |
# A World without Transactions

<table>
<thead>
<tr>
<th>Time</th>
<th>Client 1</th>
<th>Client 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_0$</td>
<td><code>UPDATE playlist</code>&lt;br&gt;SET <code>count</code> = <code>count</code> + 1&lt;br&gt;WHERE user = 'Alice';</td>
<td><code>UPDATE playlist</code>&lt;br&gt;SET <code>count</code> = <code>count</code> + 1&lt;br&gt;WHERE user = 'Alice';</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$t_1$</td>
<td><code>SELECT count</code>&lt;br&gt;FROM playlist&lt;br&gt;WHERE user = 'Alice';</td>
<td><code>SELECT count</code>&lt;br&gt;FROM playlist&lt;br&gt;WHERE user = 'Alice';</td>
</tr>
</tbody>
</table>
Transaction Guarantees

• Atomicity
• Consistency
• Isolation
• Durability
Transaction Blocks

BEGIN TRANSACTION;
  {some SQL statement 1}
  {some SQL statement 2}
  {some SQL statement n}
COMMIT;

BEGIN TRANSACTION;
  {some SQL statement 1}
  {some SQL statement 2}
  {some SQL statement n}
ROLLBACK;
Database Indexes

- **Critical** for many databases
- At least one index per table
- DBA analyzes workload and chooses which indexes to create (no easy answers)
- Creating indexes can be an expensive operation
- They work “behind the scenes”
- Query optimizer decides which indexes to use during execution

```sql
CREATE INDEX empid_idx ON Employee(empid);
CREATE INDEX empid_idx ON Employee(empid, salary);
```
B-Trees

- Standard index implementation in relational databases
- Designed to speed up lookups and range queries
- One tree node maps to one disk page
- Nodes store index entries
- Index entry = (key, ref)
- Branching factor 100+
- Height is $O(\log n)$
- Search speed $\approx$ height of tree
Spanner Overview

- Distributed database system:
  1 Spanner instance == 1...1000’s nodes
- Regional and multi-regional configurations
- Implements relational model
- Standard SQL (+ table hierarchies)
- Implements ACID transactions
- TrueTime assigns globally consistent time
- Compute and storage are decoupled
- Data is split based based on load and volume
- Dynamic split assignments to nodes
- Massive scale (PBs, 1000+ nodes)
- Higher latency per QPS than MySQL, etc.
Spanner Code Lab

- Clone snippets repo
- Open spanner notebook
- Create shopify database
- Populate shopify tables
- Run transactions
- Create foreign key
- Create index
Practice Problem 1

Debug this query and create an index to try to speed up its runtime.

```
SELECT *, c.title
WHERE c.title = 'Productivity'
FROM categories c JOIN apps_categories
ON c.id = category_id
AND reviews_count >= 50
AND rating >= 4.0
JOIN apps ON id = app_id;
```
Practice Problem 2

1. Write a query that returns all records in `pricing_plans` whose `app_id` values don’t exist in the table `apps.id`.

2. If the above query returns NULL, create a Foreign Key on `pricing_plans.app_id` which references `apps.id`. 
Project 3