

2023

The screenshot shows a presentation slide titled "2. Missing messages". The slide contains a diagram illustrating a database schema with two tables: "Entry" and "AccountVersion". The "Entry" table has fields: id (int), account_id (int), amount (float), created_at (timestamp), and updated_at (timestamp). The "AccountVersion" table has fields: id (int), account_id (int), current_balance (float), previous_balance (float), previous_version (int), and updated_at (timestamp). Below the diagram is a SQL query:

```
SELECT id
FROM entries
WHERE account_version IS NULL
AND created_at < (
    CURRENT_TIMESTAMP -
    INTERVAL '2 minutes'
)
```

The slide also features the fintech devcon logo and social media links: @fintechdevcon, #fintechdevcon, and fintechdevcon.io.

Double Entry Accounting Solutions at Scale: [Link](#)

Other Key Points I learnt while developing this.

- Choosing between NodeJs' express & Go Lang's Gin:
 - I decided to move with Go, since it's concurrency feature is industry appreciated and easier for developers, minimal size package
 - Dealing with Queues as said in above video was made simple using a separate go worker.
- Running Internal Postgres container for development was fine, but scaling issue when we containerize using docker (Backend + Postgres)
- Prisma client for Go lang support is still not matured as much as node.js client but works well for our usecase.
- Generating prisma client for Go from schema.prisma while dockerizing conflicts with already existing files which should be ignored (check .dockerignore file) while copying command is run in Dockerfile
- Ensured concurrency and Deadlock states to be avoided using asynchronous queue background worker (video reference)
- Also tried testing using **Grafana K6**
- I did used AI to help me write this code as I'm new to using Go Lang since I come from MERN environment and time constraint. I took help of Google's Gemini to learn most of the concepts and nuances.