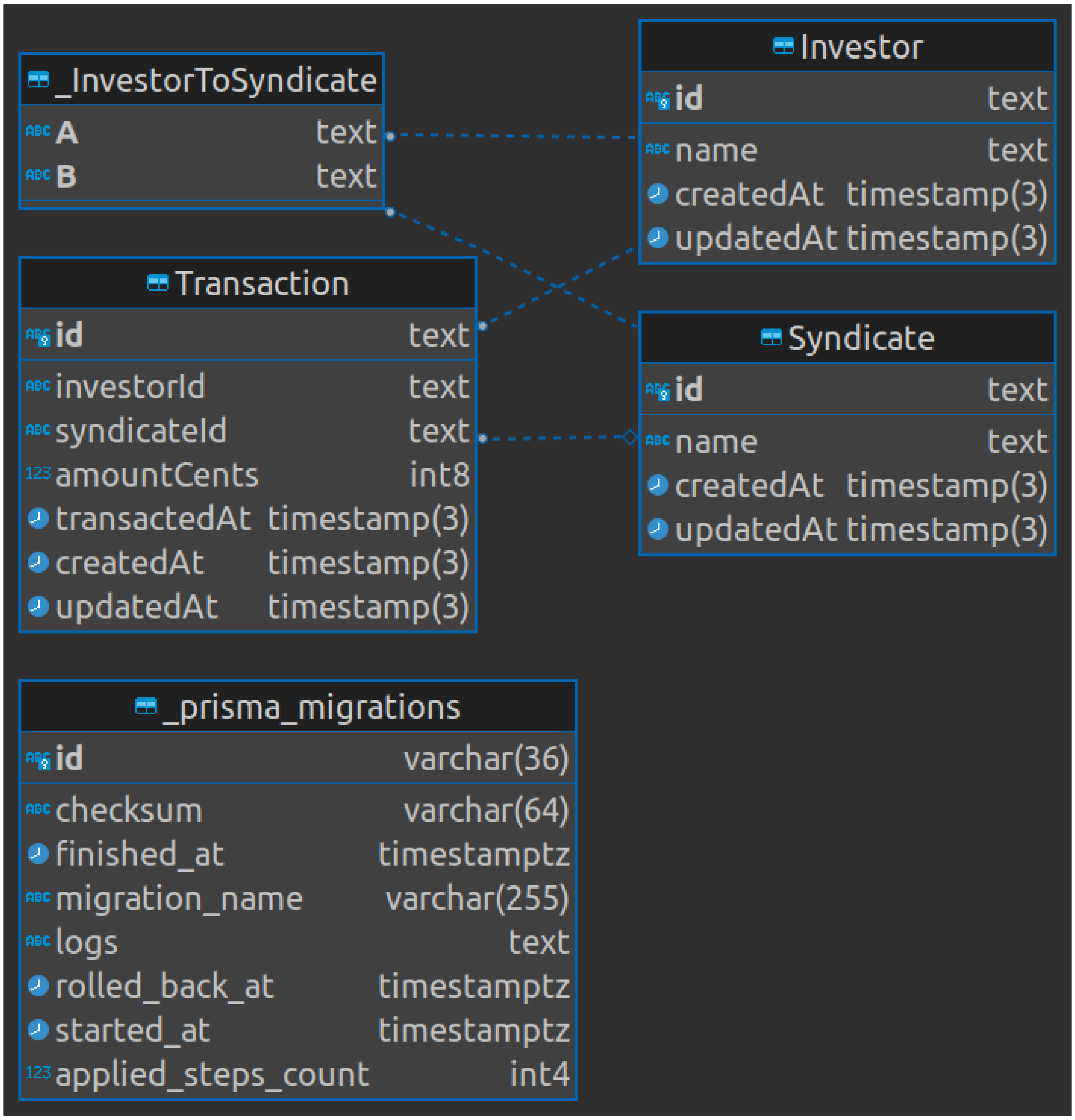
Task 2

For the real-time alert system, let's consider the PostgreSQL database schema used in Task 1.



_InvestorToSyndicate	
A	text
B	text

Investor	
id	text
name	text
createdAt	timestamp(3)
updatedAt	timestamp(3)

Transaction	
id	text
investorId	text
syndicateId	text
amountCents	int8
transactedAt	timestamp(3)
createdAt	timestamp(3)
updatedAt	timestamp(3)

Syndicate	
id	text
name	text
createdAt	timestamp(3)
updatedAt	timestamp(3)

_prisma_migrations	
id	varchar(36)
checksum	varchar(64)
finished_at	timestamptz
migration_name	varchar(255)
logs	text
rolled_back_at	timestamptz
started_at	timestamptz
applied_steps_count	int4

The alert system will trigger based on 2 different scenarios

1. When a single transaction amount exceeds a pre-defined threshold
2. When a sudden spike occurs in the number of transactions within a short period (e.g. 10x the average rate in the last hour)

Scenario 1
Each time a new Transaction is being created or the amount in an existing Transaction is updated, we will validate the input amount against the pre-defined threshold amount. If the input amount exceeds the threshold amount, we emit an event in the system. The system catches the event with a listener and sends a notification to the fund manager via WebScoket in real-time. We could extend the system in Task 1 and implement the solution in NestJS with EventEmitter2. Apache Kafka might be a good option here. Although I believe that socket.io would be a good enough option in this case for the real-time alert. Although it’s not mentioned in the problem statement, we might need to decide on whether the system should allow Transaction create/update after the alert.

Scenario 2
Let’s assume that we need to check the Transaction spike occurrence within a span of 10 minutes and we need to validate it against 10x the average of the number of transactions within each 10 minutes interval in the past hour. Whenever the system is about to create we need to do some calculations with the following queries.

1. ***Number of transactions in the last 10 minutes:***
   ```
   SELECT COUNT(*) as "recent_transaction_count"
   FROM "Transaction"
   WHERE "transactedAt" >= (NOW() - INTERVAL '10 minutes');
   ```
2. ***10x the average number of transaction rate of each 10 minute interval within the last hour:***
   ```
   WITH "intervals" AS (
           SELECT GENERATE_SERIES(
              NOW() - INTERVAL '1 hour',
              NOW(),
              INTERVAL '10 minutes'
           ) AS "interval_start"
   )
   SELECT (SUM("transaction_counts"."transaction_count") / COALESCE(COUNT(*), 0)) *
   10 AS "max_avg_transaction_rate"
   FROM (
           SELECT
                   "interval_start",
                   "interval_start" + INTERVAL '10 minutes' AS "interval_end",
                   COUNT("t"."id") AS "transaction_count"
           FROM "intervals"
           LEFT JOIN "Transaction" "t"
           ON "t"."transactedAt" BETWEEN "intervals"."interval_start" AND
   "intervals"."interval_start" + INTERVAL '10 minutes'
           GROUP BY "interval_start"
           ORDER BY "interval_start" DESC
   ) AS "transaction_counts";
   ```

If the Query 1 result is greater than Query 2 result, the system will fire an event. The listener will then in turn send an alert to the fund manager via WebSocket. Although it’s not mentioned in the problem statement, we might need to decide on whether the system should allow Transaction create after the alert.