```
import pytest
from uuid import uuid4
from swarms_cloud.loggers.log_api_request_to_supabase import (
    ModelAPILogEntry,
    log_to_supabase,
)
from supabase import Client


# Mocking Supabase Client
class MockSupabaseClient(Client):
    def __init__(self, *args, **kwargs):
        pass

    def table(self, table_name):
        return self

    def insert(self, entry):
        return self

    async def execute(self):
        return {"data": {}, "error": None}


# Test cases
@pytest.mark.asyncio
```

```
async def test_log_to_supabase():
    # Mocking the Supabase client
    supabase_client = MockSupabaseClient()

    # Creating a mock entry
    entry = ModelAPILogEntry(
        id=uuid4(),
        user_id=uuid4(),
        api_key_id=uuid4(),
        model_id=uuid4(),
        input_tokens=100,
        output_tokens=200,
        all_cost=1.0,
        input_cost=0.5,
        output_cost=0.5,
        messages={"message": "test"},
        status=200,
        temperature=0.1,
        top_p=0.1,
        echo=False,
        max_tokens=500,
        stream=False,
        repetition_penalty=1.0,
    )

    # Calling the function with the mock entry and client
```

```
    response = await log_to_supabase(
        table_name="swarms_cloud_api_key_activities",
        entry=entry,
        supabase=supabase_client,
    )

    # Asserting that the function returns a response with no error
    assert response["error"] is None


@pytest.mark.asyncio
async def test_log_to_supabase_exception():
    # Mocking the Supabase client to raise an exception
    class MockSupabaseClientException(MockSupabaseClient):
        async def execute(self):
            raise Exception("Test exception")

    supabase_client = MockSupabaseClientException()

    # Creating a mock entry
    entry = ModelAPILogEntry(
        id=uuid4(),
        user_id=uuid4(),
        api_key_id=uuid4(),
        model_id=uuid4(),
        input_tokens=100,
```

```
    output_tokens=200,
    all_cost=1.0,
    input_cost=0.5,
    output_cost=0.5,
    messages={"message": "test"},
    status=200,
    temperature=0.1,
    top_p=0.1,
    echo=False,
    max_tokens=500,
    stream=False,
    repetition_penalty=1.0,
)

# Calling the function with the mock entry and client
with pytest.raises(Exception) as e:
    await log_to_supabase(
        table_name="swarms_cloud_api_key_activities",
        entry=entry,
        supabase=supabase_client,
    )

# Asserting that the function raises an exception
assert str(e.value) == "Test exception"
```