🚀 Supabase Database Migratie Guide - WritGo

📊 Huidige Database Situatie

Database Type: PostgreSQL (hosted)
Aantal Models: 71 tabellen
Framework: Prisma ORM
Compatibiliteit: ✅ Volledig compatibel met Supabase

✨ Waarom Supabase?

Voordelen

- ✅ **PostgreSQL**: Direct compatibel met je huidige setup
- ✅ **Gratis Tier**: 500 MB database, 2 GB file storage
- ✅ **Auto Backup**: Dagelijkse backups (op betaalde plans)
- ✅ **Real-time**: Optionele real-time subscriptions
- ✅ **Built-in Auth**: Kan NextAuth.js aanvullen/vervangen
- ✅ **Storage API**: File upload alternatief voor S3
- ✅ **Admin UI**: Makkelijke database management
- ✅ **Edge Functions**: Serverless functies (optioneel)

Beperkingen Gratis Tier

- 500 MB database (upgrade naar Pro: $25/maand voor 8 GB)
- 2 GB file storage
- 50 MB file upload limit
- 2 GB bandwidth per maand
- 1 project

📋 Migratie Stappen

Fase 1: Supabase Setup (15 min)

1. Maak Supabase Project

1. Ga naar: https://supabase.com/dashboard
2. Sign up / Log in (gratis account)
3. Klik **New Project**
4. Vul in:
   - **Name**: `writgo-production`
   - **Database Password**: Kies een sterk wachtwoord (bewaar dit veilig!)
   - **Region**: `West EU (Ireland)` (dichtstbij Nederland)
   - **Pricing**: Free tier (upgrade later indien nodig)
5. Wacht 2-3 minuten tot project klaar is

2. Verkrijg Connection String

1. In je Supabase project:
   - Ga naar **Settings** (tandwiel icoon)
   - Klik **Database**
   - Scroll naar **Connection string**
   - Selecteer **Nodejs** (Prisma compatible)

2. Je ziet iets als:
   `postgresql://postgres:[YOUR-PASSWORD]@db.xxxxx.supabase.co:5432/postgres`

3. Vervang `[YOUR-PASSWORD]` met je database password

4. **Voeg connection pooling toe** voor betere performance:
   `postgresql://postgres:[YOUR-PASSWORD]@db.xxxxx.supabase.co:6543/postgres?pgbouncer=true`
   (Let op: poort 6543 i.p.v. 5432 voor pooling)

3. Update .env (Lokaal & Productie)

BELANGRIJK: Test eerst lokaal, daarna pas productie!

```
# .env
# OUDE DATABASE_URL (bewaar als backup)
# DATABASE_URL="postgresql://
role_660998b92:rtnUeIerDQmGCoPTTRSjuAGdgxVifMxH@db-660998b92..."

# NIEUWE SUPABASE URL
DATABASE_URL="postgresql://postgres:YOUR_PASSWORD@db.xxxxx.supabase.co:6543/postgres?
pgbouncer=true"

# Voor migraties (zonder pooling)
DIRECT_DATABASE_URL="postgresql://postgres:YOUR_PASSWORD@db.xxxxx.supabase.co:5432/
postgres"
```

Fase 2: Database Schema Migratie (10 min)

1. Update Prisma Schema

```
cd /home/ubuntu/writgo_planning_app/nextjs_space

# Open prisma/schema.prisma
nano prisma/schema.prisma
```

Voeg toe aan `datasource db` sectie:

```
datasource db {
    provider = "postgresql"
    url      = env("DATABASE_URL")
    directUrl = env("DIRECT_DATABASE_URL")  // Voeg deze regel toe!
}
```

2. Push Schema naar Supabase

Optie A: Nieuwe Database (Aanbevolen voor test)

```
# Update .env met Supabase URL
# Dan run:
yarn prisma db push
```

Dit creëert alle 71 tabellen in Supabase.

Optie B: Met Migratie Historie

```
# Genereer migratie
yarn prisma migrate dev --name init_supabase

# Deploy naar productie
yarn prisma migrate deploy
```

3. Verify Schema

```
# Check of alle tabellen zijn aangemaakt
yarn prisma studio

# Of via Supabase Dashboard:
# Project → Table Editor → Check tables
```

Fase 3: Data Migratie (30-60 min)

Optie 1: pg_dump/pg_restore (Aanbevolen voor grote datasets)

Stap 1: Dump Huidige Database

```
# Install PostgreSQL client tools
sudo apt-get update
sudo apt-get install postgresql-client -y

# Dump data (zonder schema, alleen data)
pg_dump "postgresql://
role_660998b92:rtnUeIerDQmGCoPTTRSjuAGdgxVifMxH@db-660998b92.db002.hosteddb.reai.io:
5432/660998b92" \
  --data-only \
  --no-owner \
  --no-privileges \
  --file=writgo_data_backup.sql

echo "✅ Data backup completed"
ls -lh writgo_data_backup.sql
```

Stap 2: Restore naar Supabase

```
# Restore data
psql "postgresql://postgres:YOUR_PASSWORD@db.xxxxx.supabase.co:5432/postgres" \
  -f writgo_data_backup.sql

echo "✅ Data restore completed"
```

Stap 3: Verify Data

```
# Check record counts
psql "postgresql://postgres:YOUR_PASSWORD@db.xxxxx.supabase.co:5432/postgres" -c "
  SELECT 'User' as table_name, COUNT(*) FROM \"User\" UNION ALL
  SELECT 'Client', COUNT(*) FROM \"Client\" UNION ALL
  SELECT 'Project', COUNT(*) FROM \"Project\" UNION ALL
  SELECT 'ContentPiece', COUNT(*) FROM \"ContentPiece\" UNION ALL
  SELECT 'Video', COUNT(*) FROM \"Video\";
"
```

Optie 2: Prisma Data Migration Script (Voor kleinere datasets)

Creëer `/home/ubuntu/writgo_planning_app/migrate_data.ts`:

```
import { PrismaClient } from '@prisma/client'

const oldDb = new PrismaClient({
  datasourceUrl: 'postgresql://role_660998b92:...@db-660998b92...'
})

const newDb = new PrismaClient({
  datasourceUrl: 'postgresql://postgres:...@db.xxxxx.supabase.co:5432/postgres'
})

async function migrateData() {
  console.log('Starting data migration...')

  // Migrate Users
  const users = await oldDb.user.findMany()
  console.log(`Migrating ${users.length} users...`)
  await newDb.user.createMany({ data: users, skipDuplicates: true })

  // Migrate Clients
  const clients = await oldDb.client.findMany()
  console.log(`Migrating ${clients.length} clients...`)
  await newDb.client.createMany({ data: clients, skipDuplicates: true })

  // Migrate Projects
  const projects = await oldDb.project.findMany()
  console.log(`Migrating ${projects.length} projects...`)
  await newDb.project.createMany({ data: projects, skipDuplicates: true })

  // Continue for all models...
  // (Add similar blocks for other important models)

  console.log('✅ Migration completed!')
}

migrateData()
  .catch(console.error)
  .finally(async () => {
    await oldDb.$disconnect()
    await newDb.$disconnect()
  })
```

Run migratie:

```
cd /home/ubuntu/writgo_planning_app/nextjs_space
yarn tsx ../migrate_data.ts
```

Fase 4: Testing (20 min)

1. Lokaal Testen

```
cd /home/ubuntu/writgo_planning_app/nextjs_space

# Update .env met Supabase URL
# Test de app
yarn dev

# Open http://localhost:3000
# Test:
# ✅ Login/signup
# ✅ Content generatie
# ✅ Project overzicht
# ✅ Admin dashboard
# ✅ Database queries
```

2. Verificatie Queries

```
# Check data integrity
yarn prisma studio

# Of via SQL:
psql "postgresql://postgres:YOUR_PASSWORD@db.xxxxx.supabase.co:5432/postgres" -c "
  -- Check totals
  SELECT
    (SELECT COUNT(*) FROM \"User\") as users,
    (SELECT COUNT(*) FROM \"Client\") as clients,
    (SELECT COUNT(*) FROM \"Project\") as projects,
    (SELECT COUNT(*) FROM \"ContentPiece\") as content_pieces,
    (SELECT COUNT(*) FROM \"CreditTransaction\") as transactions;
"
```

Fase 5: Productie Deployment (15 min)

1. Update Environment Variables

Op je productie server (writgoai.nl):

```
# SSH naar productie
ssh user@writgoai.nl

cd /pad/naar/app

# Backup oude .env
cp .env .env.backup.$(date +%Y%m%d)

# Update DATABASE_URL
nano .env
# Wijzig DATABASE_URL naar Supabase connection string
```

2. Deploy App met Nieuwe Database

```
# Rebuild app
yarn install
yarn build

# Restart app
pm2 restart writgo-app
# Of als je systemd gebruikt:
sudo systemctl restart writgo

# Check logs
pm2 logs writgo-app
# Of:
sudo journalctl -u writgo -f
```

3. Verify Productie

- Ga naar https://writgoai.nl
- Test alle kritieke flows:
- ✅ Login
- ✅ Content generatie
- ✅ Payment (Stripe)
- ✅ Admin functies

Fase 6: Cleanup & Optimalisatie

1. Database Indexing Check

Supabase heeft automatisch indexes, maar check custom indexes:

```
-- In Supabase SQL Editor:
CREATE INDEX IF NOT EXISTS idx_content_client_id ON "ContentPiece"("clientId");
CREATE INDEX IF NOT EXISTS idx_project_client_id ON "Project"("clientId");
CREATE INDEX IF NOT EXISTS idx_credit_transaction_client_id ON "CreditTransaction"("cl
ientId");
```

2. Backup Strategy

Automatische Backups (Supabase Pro):
- Dagelijkse backups (7 dagen bewaard)
- Point-in-time recovery

Handmatige Backups (Gratis tier):

```
# Wekelijkse cron job
crontab -e

# Voeg toe:
0 2 * * 0 pg_dump "postgresql://postgres:YOUR_PASSWORD@db.xxxxx.supabase.co:5432/post-
gres" | gzip > /backups/writgo_$(date +\%Y\%m\%d).sql.gz
```

3. Oude Database

BELANGRIJK: Behoud oude database 30 dagen als backup!

```
# Na succesvolle migratie en 30 dagen stabiele werking:
# 1. Maak final backup van oude database
# 2. Cancel oude database hosting
# 3. Update .env.backup met oude connection string (voor noodgevallen)
```

🔧 Troubleshooting

Error: “Too many connections”

Oplossing: Gebruik connection pooling:

```
DATABASE_URL="postgresql://postgres:pass@db.xxx.supabase.co:6543/postgres?pgboun-
cer=true"
DIRECT_DATABASE_URL="postgresql://postgres:pass@db.xxx.supabase.co:5432/postgres"
```

Update `prisma/schema.prisma`:

```
generator client {
  provider = "prisma-client-js"
  previewFeatures = ["postgresqlExtensions"]
}

datasource db {
  provider = "postgresql"
  url = env("DATABASE_URL")
  directUrl = env("DIRECT_DATABASE_URL")
}
```

Error: “Migration failed”

Oplossing: Reset en re-push schema:

```
# WAARSCHUWING: Dit wist alle data!
yarn prisma migrate reset
yarn prisma db push
```

Error: “Schema mismatch”

Oplossing: Regenerate Prisma Client:

```
yarn prisma generate
yarn build
```

Performance Issues

Oplossing: Enable pgvector en indexes:

```
-- In Supabase SQL Editor
CREATE EXTENSION IF NOT EXISTS pgvector;

-- Check slow queries
SELECT query, calls, total_time, mean_time
FROM pg_stat_statements
ORDER BY mean_time DESC
LIMIT 10;
```

📊 Kosten Vergelijking

Huidige Hosting

- **Provider**: Hosted DB (Reai.io?)
- **Geschatte kosten**: €10-50/maand?

Supabase

Free Tier (perfect voor start):
- 500 MB database
- 2 GB file storage
- 50k monthly active users
- Community support
- **Kosten**: €0/maand

Pro Tier (bij groei):
- 8 GB database
- 100 GB file storage
- 100k monthly active users
- Email support
- Daily backups
- **Kosten**: $25/maand (~€23)

Enterprise (bij schaal):
- Custom resources
- 99.99% SLA
- Dedicated support
- **Kosten**: Op aanvraag

✅ Pre-Migration Checklist

- [ ] Supabase account aangemaakt
- [ ] Project gecreëerd (EU region)
- [ ] Database password veilig opgeslagen
- [ ] Connection strings verkregen
- [ ] Backup van huidige database gemaakt
- [ ] .env.backup gemaakt
- [ ] Downtime window gecommuniceerd (indien applicable)
- [ ] Test environment opgezet
- [ ] Rollback plan klaar

✅ Post-Migration Checklist

- [ ] Alle tabellen gemigreerd (71/71)
- [ ] Data counts geverifieerd
- [ ] Login/signup werkt
- [ ] Content generatie werkt
- [ ] Stripe payments werken
- [ ] Admin functies werken
- [ ] Performance acceptable
- [ ] Backup strategie actief
- [ ] Monitoring setup (Supabase dashboard)
- [ ] Oude database bewaard als backup (30 dagen)

🚨 Rollback Plan

Als er problemen zijn na migratie:

```
# 1. Stop de app
pm2 stop writgo-app

# 2. Restore oude .env
cp .env.backup .env

# 3. Restart app
pm2 start writgo-app

# 4. Verify oude database werkt
# 5. Debug Supabase issue offline
```

📚 Handige Links

- **Supabase Docs**: https://supabase.com/docs
- **Prisma + Supabase**: https://www.prisma.io/docs/guides/database/supabase
- **Connection Pooling**: https://supabase.com/docs/guides/database/connecting-to-postgres#connection-pooler
- **Supabase Status**: https://status.supabase.com/

💡 Extra Features met Supabase

Na succesvolle migratie kun je overwegen:

1. **Supabase Storage**: Vervang AWS S3 voor file uploads
2. **Supabase Auth**: Optioneel alternatief voor NextAuth.js
3. **Real-time**: Live updates voor collaborative features
4. **Edge Functions**: Serverless API endpoints
5. **PostgREST**: Auto-generated REST API

🎯 Aanbevolen Timeline

Week 1: Setup & Testing
- Dag 1-2: Supabase account + schema migratie
- Dag 3-4: Data migratie + lokaal testen
- Dag 5-7: Uitgebreid testen

Week 2: Productie
- Dag 8: Production deployment (buiten kantooruren)
- Dag 9-10: Monitoring & optimalisatie
- Dag 11-14: Stabiliteit check

Week 3-6: Backup periode
- Oude database actief houden
- Wekelijkse backups
- Performance monitoring

Week 7: Cleanup
- Cancel oude database subscription
- Finalize backup strategy

Klaar om te starten? Begin met Fase 1 en neem contact op bij vragen!