BlogPost Migratie Fix Guide

🚨 Probleem

De originele BlogPost migratie (`20241217200000_writgo_blog.sql`) gaf de volgende error:

```
ERROR: column 'category' does not exist
```

Root Cause

Dit gebeurde omdat:
1. De `BlogPost` tabel mogelijk al bestond maar zonder alle kolommen
2. De index `BlogPost_category_idx` werd aangemaakt op de `category` kolom voordat deze bestond
3. Er was geen expliciete check of kolommen al bestonden voordat indices werden aangemaakt

✅ Oplossing

Een nieuwe, veilige migratie (`20241217220000_blogpost_fixed.sql`) die:
- **Stap voor stap** alle resources aanmaakt
- **Checkt of resources al bestaan** voordat ze worden aangemaakt
- **Idempotent** is - kan veilig meerdere keren worden uitgevoerd
- **Backwards compatible** is met bestaande data

📋 Wat de Fix Doet

Stap 1: Tabel Aanmaken

```
CREATE TABLE IF NOT EXISTS "BlogPost" (...)
```

- Maakt de basis tabel aan met alleen de essentiële kolommen
- Gebruikt `IF NOT EXISTS` om dubbele aanmaak te voorkomen

Stap 2: Kolommen Toevoegen

```
DO $$
BEGIN
  IF NOT EXISTS (SELECT ... WHERE column_name = 'category') THEN
    ALTER TABLE "BlogPost" ADD COLUMN "category" TEXT;
  END IF;
END $$;
```

- Voegt elke kolom **individueel** toe
- Checkt eerst of de kolom al bestaat
- Vermijdt “column already exists” errors

Stap 3: Constraints Toevoegen

```
ALTER TABLE "BlogPost" ADD CONSTRAINT "BlogPost_slug_key" UNIQUE ("slug");
```

- Voegt UNIQUE constraint toe aan slug
- Checkt eerst of constraint al bestaat

Stap 4: Indices Aanmaken

```
CREATE INDEX IF NOT EXISTS "BlogPost_category_idx" ON "BlogPost"("category");
```

- Maakt performance indices aan
- **Alleen NADAT alle kolommen bestaan!**
- Gebruikt `IF NOT EXISTS` om dubbele aanmaak te voorkomen

Stap 5 & 6: Relaties

- Voegt `blogPostId` kolom toe aan `PlannedArticle`
- Maakt foreign key constraint aan
- Alles met veilige checks

🚀 Uitvoeren van de Migratie

Optie 1: Supabase Dashboard (Aanbevolen voor productie)

1. **Open Supabase Dashboard**
   - Ga naar supabase.com (https://supabase.com)
   - Selecteer je project
   - Ga naar **SQL Editor**

2. **Kopieer de Migratie**
   `bash`
   `cat supabase/migrations/20241217220000_blogpost_fixed.sql`

3. **Plak en Run**
   - Plak de volledige SQL in de editor
   - Klik op **Run** (of gebruik `Ctrl+Enter`)

4. **Verify Success**
   - Kopieer queries uit `test_blogpost_migration.sql`
   - Run de verificatie queries

Optie 2: Supabase CLI (Voor development)

```
cd /home/ubuntu/writgoai_nl/nextjs_space

# Push alle pending migraties
supabase db push

# Of push specifiek bestand
supabase db push supabase/migrations/20241217220000_blogpost_fixed.sql
```

Optie 3: Direct SQL (Voor quick fix)

```
# Via psql (als je directe database toegang hebt)
psql $DATABASE_URL -f supabase/migrations/20241217220000_blogpost_fixed.sql

# Of via Supabase CLI
supabase db execute -f supabase/migrations/20241217220000_blogpost_fixed.sql
```

✓ Verificatie

Quick Check

Run deze query om te zien of alles werkt:

```
-- Moet 16 kolommen tonen
SELECT column_name
FROM information_schema.columns
WHERE table_schema = 'public' AND table_name = 'BlogPost'
ORDER BY ordinal_position;
```

Verwachte Output:

```
id
slug
title
content
status
createdAt
updatedAt
excerpt
coverImage
author
category        ← Deze moet er zijn!
tags
metaTitle
metaDescription
focusKeyword
publishedAt
```

Volledige Verificatie

Run **ALLE** queries uit `test_blogpost_migration.sql`:

```
# In Supabase Dashboard SQL Editor
# Kopieer en run alle queries uit test_blogpost_migration.sql
```

🔧 Troubleshooting

Error: “relation BlogPost already exists”

Oplossing: De migratie is al uitgevoerd. Skip deze stap.

Error: “column category still doesn’t exist”

Diagnose:

```
-- Check welke kolommen er WEL zijn
SELECT column_name FROM information_schema.columns
WHERE table_name = 'BlogPost';
```

Fix:

```
-- Voeg manueel toe
ALTER TABLE "BlogPost" ADD COLUMN "category" TEXT;
CREATE INDEX IF NOT EXISTS "BlogPost_category_idx" ON "BlogPost"("category");
```

Error: “constraint BlogPost_slug_key already exists”

Oplossing: Dit is OK! De constraint bestaat al. Ga verder.

Nuclear Option: Complete Reset (⚠️ VERLIEST ALLE DATA!)

Alleen gebruiken als development database!

```
-- VOORZICHTIG! Dit verwijdert ALLE BlogPost data!
DROP TABLE IF EXISTS "BlogPost" CASCADE;

-- Run dan de migratie opnieuw
-- Kopieer en run: 20241217220000_blogpost_fixed.sql
```

📊 Database Schema

BlogPost Tabel Structuur

```
interface BlogPost {
  id: string;                    // UUID primary key
  slug: string;                  // Unique, indexed
  title: string;                 // Required
  excerpt?: string;              // Optional teaser
  content: string;               // Full article content
  coverImage?: string;           // URL to cover image
  author?: string;               // Default: 'Writgo Team'
  category?: string;             // Indexed for filtering
  tags?: string[];               // Array of tags
  metaTitle?: string;            // SEO title
  metaDescription?: string;      // SEO description
  focusKeyword?: string;         // Main SEO keyword
  status: string;                // 'draft' | 'published'
  publishedAt?: Date;            // When published
  createdAt: Date;               // Auto-generated
  updatedAt: Date;               // Auto-generated
}
```

Indices

- `BlogPost_slug_idx` - Voor slug lookups
- `BlogPost_status_idx` - Voor filtering op status
- `BlogPost_publishedAt_idx` - Voor chronologische sorting
- `BlogPost_category_idx` - Voor category filtering

Relaties

```
PlannedArticle
  └─ blogPostId (TEXT, nullable)
      └─ FK → BlogPost.id (ON DELETE SET NULL)
```

🔄 Rollback Plan

Als je de migratie wilt terugdraaien:

Stap 1: Verwijder Foreign Key

```sql
ALTER TABLE "PlannedArticle"
DROP CONSTRAINT IF EXISTS "PlannedArticle_blogPostId_fkey";
```

Stap 2: Verwijder Kolom uit PlannedArticle

```sql
ALTER TABLE "PlannedArticle"
DROP COLUMN IF EXISTS "blogPostId";
```

Stap 3: Drop BlogPost Tabel

```sql
DROP TABLE IF EXISTS "BlogPost" CASCADE;
```

📝 Code Gebruik

Via Prisma Shim (Recommended)

```typescript
import { prisma } from '@/lib/prisma-shim';

// Create blog post
const post = await prisma.blogPost.create({
  data: {
    slug: 'mijn-eerste-post',
    title: 'Mijn Eerste Post',
    content: 'Dit is de inhoud...',
    category: 'Tutorial',
    status: 'draft',
  },
});

// Find by slug
const post = await prisma.blogPost.findUnique({
  where: { slug: 'mijn-eerste-post' },
});

// List published posts
const posts = await prisma.blogPost.findMany({
  where: { status: 'published' },
  orderBy: { publishedAt: 'desc' },
});
```

Via Direct Supabase

```
import { supabaseAdmin } from '@/lib/supabase';

// Create
const { data, error } = await supabaseAdmin
  .from('BlogPost')
  .insert({
    slug: 'mijn-eerste-post',
    title: 'Mijn Eerste Post',
    content: 'Dit is de inhoud...',
  })
  .select()
  .single();

// Query
const { data, error } = await supabaseAdmin
  .from('BlogPost')
  .select('*')
  .eq('status', 'published')
  .order('publishedAt', { ascending: false });
```

🎯 Next Steps

Na succesvolle migratie:

1. **Update API Routes**
   - Maak `/api/blog-posts` routes
   - Implementeer CRUD operations

2. **Create Blog UI**
   - Blog listing page
   - Individual post page
   - Admin interface voor posts

3. **Link met PlannedArticle**
   - Implementeer “Publish to Blog” feature
   - Auto-sync van gegenereerde content

4. **Add Blog Features**
   - Comments systeem
   - Related posts
   - RSS feed
   - Sitemap.xml

📞 Support

Als je nog steeds problemen hebt:

1. **Check Supabase Logs**
   - Dashboard → Logs → Database

2. **Run Diagnostic Query**
   `sql`
   `SELECT`

```
    table_name,
    column_name,
    data_type
  FROM information_schema.columns
  WHERE table_name IN ('BlogPost', 'PlannedArticle')
  ORDER BY table_name, ordinal_position;
```

3. **Contact Info**
   - Stuur query resultaten
   - Include error messages
   - Beschrijf wat je probeerde te doen

✨ Benefits van de Fix

- ✅ **Idempotent** - Kan veilig meerdere keren uitgevoerd worden
- ✅ **Backwards Compatible** - Werkt met bestaande data
- ✅ **Defensive** - Checkt alles voordat het wordt aangemaakt
- ✅ **Documented** - Elke stap is gedocumenteerd
- ✅ **Testable** - Komt met verificatie queries
- ✅ **Production Ready** - Veilig voor productie gebruik

📜 Change Log

v2 - Fixed Migration (20241217220000)

- ✅ Voegt kolommen individueel toe met existence checks
- ✅ Checkt constraints voordat ze worden aangemaakt
- ✅ Indices worden NADAT kolommen bestaan aangemaakt
- ✅ Volledige error handling
- ✅ Uitgebreide verificatie

v1 - Original (20241217200000)

- ❌ Error: column ‘category’ does not exist
- ❌ Index werd aangemaakt voordat kolom bestond

Status: ✅ Ready for Deployment
Last Updated: December 17, 2025
Version: 2.0 (Fixed)