


BELANGRIJKE DATABASE MIGRATIE VEREIST

Probleem

De `contentPlan` kolom bestaat niet in de database, waardoor content plan save faalt met de volgende fout:

 Failed to save content plan
Error: column "contentPlan" does not exist

Oplossing

Voer de SQL migratie uit in Supabase Dashboard om de `contentPlan` kolom toe te voegen.

Stap-voor-Stap Instructies

1. Open Supabase Dashboard

1. Ga naar <https://supabase.com/dashboard> (<https://supabase.com/dashboard>)
2. Log in met je account
3. Selecteer je Writgo project
4. Klik op **"SQL Editor"** in het linkermenu

2. Kopieer en Plak de SQL Migratie

Open het bestand `/supabase/migrations/20251215_add_contentplan_robust.sql` en kopieer de volledige SQL code.

Of gebruik deze directe code:


```

-- =====
-- ROBUST Content Plan Column Migration
-- Date: 2025-12-15
-- Purpose: Add contentPlan JSONB column with full error handling
-- =====

-- Step 1: Check if column exists, if not add it
DO $$
BEGIN
    IF NOT EXISTS (
        SELECT 1
        FROM information_schema.columns
        WHERE table_schema = 'public'
              AND table_name = 'Project'
              AND column_name = 'contentPlan'
    ) THEN
        -- Add contentPlan JSONB column
        ALTER TABLE "Project"
        ADD COLUMN "contentPlan" JSONB DEFAULT '[]'::jsonb;

        RAISE NOTICE '✅ Column contentPlan added successfully';
    ELSE
        RAISE NOTICE 'ℹ️ Column contentPlan already exists, skipping creation';
    END IF;
END $$;

-- Step 2: Add index if it doesn't exist
CREATE INDEX IF NOT EXISTS "Project_contentPlan_idx"
ON "Project" USING gin ("contentPlan");

-- Step 3: Add comment
COMMENT ON COLUMN "Project"."contentPlan"
IS 'Array of content plan topics generated by AI. Each topic has: title, description,
keywords, priority';

-- Step 4: Verify column exists and is properly configured
DO $$
DECLARE
    col_exists boolean;
    col_type text;
    col_default text;
BEGIN
    SELECT
        EXISTS (
            SELECT 1
            FROM information_schema.columns
            WHERE table_schema = 'public'
                  AND table_name = 'Project'
                  AND column_name = 'contentPlan'
        ),
        data_type,
        column_default
    INTO col_exists, col_type, col_default
    FROM information_schema.columns
    WHERE table_schema = 'public'
          AND table_name = 'Project'
          AND column_name = 'contentPlan';

    IF col_exists THEN
        RAISE NOTICE '✅ SUCCESS: contentPlan column exists and is ready to use';
        RAISE NOTICE '  - Data Type: %', col_type;
        RAISE NOTICE '  - Default: %', col_default;
    END IF;
END $$;

```



```

ELSE
    RAISE EXCEPTION '✗ ERROR: contentPlan column was not created';
END IF;
END $$;

-- Step 5: Test insert/update capability
DO $$
DECLARE
    test_project_id text;
    update_success boolean := false;
BEGIN
    -- Get first project ID for testing
    SELECT id INTO test_project_id FROM "Project" LIMIT 1;

    IF test_project_id IS NOT NULL THEN
        -- Test update
        UPDATE "Project"
        SET "contentPlan" = '["title":"Migration Test","description":"Testing contentPlan column","keywords":"test","priority":"high"}']::jsonb
        WHERE id = test_project_id;

        -- Verify update worked
        SELECT
            CASE
                WHEN "contentPlan" IS NOT NULL THEN true
                ELSE false
            END
        INTO update_success
        FROM "Project"
        WHERE id = test_project_id;

        IF update_success THEN
            RAISE NOTICE '✅ Test update successful for project: %', test_project_id;

            -- Rollback test data
            UPDATE "Project"
            SET "contentPlan" = '[]'::jsonb
            WHERE id = test_project_id;

            RAISE NOTICE '✅ Test data cleaned up';
        ELSE
            RAISE WARNING '⚠️ Test update failed verification';
        END IF;
    ELSE
        RAISE NOTICE 'ℹ️ No projects found for testing (empty database)';
    END IF;
END $$;

-- Final verification query
SELECT
    'contentPlan' as column_name,
    data_type,
    column_default,
    is_nullable,
    '✅ Column is ready for use!' as status
FROM information_schema.columns
WHERE table_schema = 'public'
    AND table_name = 'Project'
    AND column_name = 'contentPlan';

-- Success message
DO $$
BEGIN

```



```

RAISE NOTICE '=====';
RAISE NOTICE '✅ MIGRATION COMPLETE: contentPlan column is ready!';
RAISE NOTICE '=====';
END $$;

```

3. Voer de SQL uit

1. Plak de SQL code in de SQL Editor
2. Klik op de **“Run”** knop (of druk Ctrl/Cmd + Enter)

4. Controleer het Resultaat

Je zou het volgende moeten zien in de output:

```

✅ Column contentPlan added successfully
✅ SUCCESS: contentPlan column exists and is ready to use
  - Data Type: jsonb
  - Default: '[]'::jsonb
✅ Test update successful for project: [project-id]
✅ Test data cleaned up
=====
✅ MIGRATION COMPLETE: contentPlan column is ready!
=====

```

Als je dit ziet, is de migratie succesvol!

5. Optioneel: Voer ook de RPC Function uit

Voor extra robuustheid, voer ook deze SQL uit:


```
-- Create RPC function for updating content plan
CREATE OR REPLACE FUNCTION update_project_content_plan(
  p_project_id text,
  p_content_plan jsonb
)
RETURNS jsonb
LANGUAGE plpgsql
SECURITY DEFINER
AS $$
DECLARE
  v_updated_project jsonb;
BEGIN
  -- Update the project
  UPDATE "Project"
  SET "contentPlan" = p_content_plan,
      "updatedAt" = NOW()
  WHERE id = p_project_id
  RETURNING jsonb_build_object(
    'id', id,
    'name', name,
    'contentPlan', "contentPlan",
    'updatedAt', "updatedAt"
  ) INTO v_updated_project;


  -- Check if update was successful
  IF v_updated_project IS NULL THEN
    RAISE EXCEPTION 'Project not found: %', p_project_id;
  END IF;

  -- Return updated project data
  RETURN v_updated_project;
END;
$$;

-- Grant execute permission
GRANT EXECUTE ON FUNCTION update_project_content_plan(text, jsonb) TO authenticated;
GRANT EXECUTE ON FUNCTION update_project_content_plan(text, jsonb) TO service_role;
GRANT EXECUTE ON FUNCTION update_project_content_plan(text, jsonb) TO anon;
```

Test de App

Na het uitvoeren van de migratie:

1. Ga naar <https://writgo.nl/content-plan> (<https://writgo.nl/content-plan>)
2. Selecteer een project met een WordPress URL
3. Klik op **“Analyseer WordPress & Genereer Plan”**
4. Wacht 30-60 seconden
5.  Je zou nu 15-20 topics moeten zien zonder errors!

Nieuwe Features Beschikbaar!

Na de migratie heb je toegang tot **4 nieuwe content types** in het Content Genereren scherm:

1. **Standaard Artikel**

- SEO-geoptimaliseerd artikel (1500 woorden)
- Gebruik: algemene informatieve content
- Voorbeeld: “Beste fitness tips voor beginners”

2. **Productreview**

- Diepgaande review van één product
- Inclusief Voor-/Nadelen tabel
- Technische specificaties
- Voorbeeld: “iPhone 15 Pro Review”

3. **Beste Lijstje**

- Top X beste producten in categorie
- Vergelijkingstabel
- Per product: analyse, pluspunten, minpunten
- Voorbeeld: “Beste 5 Draadloze Oordopjes”

4. **Vergelijking**

- Product A vs Product B vergelijking
- Diepgaande feature-by-feature analyse
- Quick summary met aanbevelingen
- Voorbeeld: “Samsung QLED vs LG OLED”

Alle types volgen Writgo's strenge regels:

- 1500 woorden
- 100% menselijke score
- E-E-A-T geoptimaliseerd
- Conversationeel B1-niveau
- ‘je/jij’ aanspreekvorm
- Verboden woorden gefilterd

Hulp Nodig?

Als je problemen hebt met de migratie:

1. **Check of de SQL compleet is uitgevoerd:** scroll door de hele output
2. **Refresh de Supabase dashboard:** soms is een refresh nodig
3. **Check de “Project” tabel:** klik op “Table Editor” → “Project” → check of “contentPlan” kolom bestaat

Veelvoorkomende Fouten

Fout: “permission denied”

- Oplossing: Zorg dat je ingelogd bent als owner van het Supabase project

Fout: “column already exists”

- Oplossing: Dit is geen probleem! De migratie is al uitgevoerd. Ga door naar de test stap.

Fout: “table Project does not exist”

- Oplossing: Je database schema is anders. Controleer de tabel naam in je schema.

Bestand Locaties

- **SQL Migratie:** `/supabase/migrations/20251215_add_contentplan_robust.sql`
 - **RPC Function:** `/supabase/migrations/20251215_create_rpc_function.sql`
 - **API Route:** `/app/api/simplified/content-plan/analyze-wordpress/route.ts`
 - **Prompts:** `/lib/writgo-prompt.ts`
-

Klaar!

Na het voltooien van deze stappen zou je:

- ☒ Content plans kunnen opslaan zonder errors
- ☒ WordPress analyse kunnen uitvoeren
- ☒ 4 verschillende content types kunnen genereren
- ☒ Alle Writgo features kunnen gebruiken

Veel plezier met de nieuwe features! 🚀