# Database Seeder Tool Documentation

The Database Seeder Tool allows you to populate your database with default data in a controlled, flexible manner. You can choose which tables to seed and when to seed them.

## 🚀 Quick Start

# Show available options  
npm run db:seed --help  
  
# Seed all tables with default data  
npm run db:seed all --verbose  
  
# Seed specific tables  
npm run db:seed permissions roles

## 📋 Available Tables

| Table | Description | Default Data |
| --- | --- | --- |
| permissions | System permissions | 12 permissions across 5 categories |
| roles | User roles | admin, user, moderator, viewer |
| role-permissions | Role-permission assignments | Smart permission mapping |
| users | Default users | admin and test user accounts |
| all | All tables in correct order | Complete system setup |

## 🎯 Command Syntax

npm run db:seed <tables> [options]

### Tables

* permissions - Seed permissions table
* roles - Seed roles table
* role-permissions - Seed role-permission assignments
* users - Seed users table
* all - Seed all tables in dependency order

### Options

* --verbose, -v - Show detailed output
* --force, -f - Force recreate assignments (clears existing)
* --help, -h - Show help information

## 📖 Detailed Examples

### Example 1: Complete System Setup

npm run db:seed all --verbose

**Output:**

🌱 Starting database seeding...  
📋 Tables to seed: permissions, roles, role-permissions, users  
  
📝 Seeding permissions...  
 ✅ Created: users:read  
 ✅ Created: users:write  
 ✅ Created: users:delete  
 📊 Permissions: 12 created, 0 skipped  
  
🎭 Seeding roles...  
 ✅ Created: admin  
 ✅ Created: user  
 ✅ Created: moderator  
 📊 Roles: 4 created, 0 skipped  
  
🔗 Seeding role-permission assignments...  
 📊 Role-Permissions: 15 assignments created  
  
👤 Seeding users...  
 ✅ Created: admin (admin@rasdash.com)  
 🔑 Password: Admin123!  
 ✅ Created: testuser (user@rasdash.com)  
 🔑 Password: User123!  
 📊 Users: 2 created, 0 skipped  
 ⚠️ Please change default passwords after first login!  
  
🎉 Database seeding completed successfully!

### Example 2: Seed Specific Tables

npm run db:seed permissions roles --verbose

**Output:**

🌱 Starting database seeding...  
📋 Tables to seed: permissions, roles  
  
📝 Seeding permissions...  
 ⏭️ Exists: users:read  
 ⏭️ Exists: users:write  
 ✅ Created: reports:read  
 📊 Permissions: 1 created, 11 skipped  
  
🎭 Seeding roles...  
 ⏭️ Exists: admin  
 ⏭️ Exists: user  
 📊 Roles: 0 created, 4 skipped  
  
🎉 Database seeding completed successfully!

### Example 3: Force Recreate Role Permissions

npm run db:seed role-permissions --force --verbose

**Output:**

🌱 Starting database seeding...  
📋 Tables to seed: role-permissions  
⚠️ Force mode: Will recreate assignments  
  
🔗 Seeding role-permission assignments...  
 🧹 Cleared existing permissions for: admin  
 🧹 Cleared existing permissions for: user  
 🧹 Cleared existing permissions for: moderator  
 ✅ Assigned users:read to admin  
 ✅ Assigned users:write to admin  
 ✅ Assigned admin:dashboard to admin  
 📊 Role-Permissions: 15 assignments created  
  
🎉 Database seeding completed successfully!

## 🗂️ Default Data Details

### Permissions (12 total)

📂 USERS Category:  
- users:read - View user information  
- users:write - Create and update users   
- users:delete - Delete users  
  
📂 ROLES Category:  
- roles:read - View roles  
- roles:write - Create and update roles  
- roles:delete - Delete roles  
  
📂 PERMISSIONS Category:  
- permissions:read - View permissions  
- permissions:write - Create and update permissions  
  
📂 ADMIN Category:  
- admin:dashboard - Access admin dashboard  
  
📂 SYSTEM Category:  
- system:manage - Manage system settings  
  
📂 REPORTS Category:  
- reports:read - View reports  
- reports:write - Create and update reports

### Roles (4 total)

🎭 ADMIN Role:  
- Description: Full system access  
- System Role: Yes  
- Default Role: No  
- Permissions: ALL (\*)  
  
👤 USER Role:  
- Description: Basic user access   
- System Role: Yes  
- Default Role: Yes  
- Permissions: users:read  
  
🛡️ MODERATOR Role:  
- Description: Moderate content and users  
- System Role: No  
- Default Role: No   
- Permissions: users:read, users:write, roles:read, reports:read  
  
👁️ VIEWER Role:  
- Description: Read-only access  
- System Role: No  
- Default Role: No  
- Permissions: users:read, reports:read

### Users (2 default)

👑 Admin User:  
- Username: admin  
- Email: admin@rasdash.com  
- Password: Admin123!  
- Role: admin  
- Status: active  
- Assigned Roles: [admin]  
  
👤 Test User:  
- Username: testuser   
- Email: user@rasdash.com  
- Password: User123!  
- Role: user  
- Status: active  
- Assigned Roles: [user]

## 🔄 Seeding Order

When using all, tables are seeded in dependency order: 1. **permissions** - Base permissions first 2. **roles** - Roles that will use permissions 3. **role-permissions** - Assign permissions to roles 4. **users** - Users that will be assigned roles

## 🛠️ Advanced Usage

### Custom Seeding Workflow

# 1. Set up permissions and roles  
npm run db:seed permissions roles  
  
# 2. Verify setup  
npm run db:query permissions:by-category  
npm run db:query roles  
  
# 3. Configure role permissions  
npm run db:seed role-permissions --force  
  
# 4. Add users when ready  
npm run db:seed users

### Incremental Updates

# Add new permissions without affecting existing data  
npm run db:seed permissions  
  
# Update role assignments  
npm run db:seed role-permissions --force

### Development vs Production

# Development: Include test users  
npm run db:seed all --verbose  
  
# Production: Skip users, set up structure only  
npm run db:seed permissions roles role-permissions

## 🔧 Troubleshooting

### Duplicate Key Errors

If you get unique constraint violations:

# Check existing data first  
npm run db:query permissions  
npm run db:query roles  
  
# Use force mode to recreate assignments  
npm run db:seed role-permissions --force

### Permission Denied

Ensure your database user has INSERT permissions:

GRANT INSERT ON permissions, roles, role\_permissions, users, user\_roles TO your\_user;

### Foreign Key Violations

Seed tables in the correct order: 1. permissions (no dependencies) 2. roles (no dependencies)  
3. role-permissions (depends on roles + permissions) 4. users (no dependencies) 5. user-roles (created automatically with users)

## 🎯 Best Practices

### 1. Start Fresh

# For new installations  
npm run db:seed all --verbose

### 2. Incremental Updates

# Add new permissions  
npm run db:seed permissions  
  
# Update role assignments   
npm run db:seed role-permissions --force

### 3. Verify Results

# Check what was created  
npm run db:query role-permissions  
npm run db:query user-roles

### 4. Security

* Change default passwords immediately
* Remove test users in production
* Review permission assignments

## 📝 Customization

### Adding New Permissions

Edit /api/scripts/db-seed.js in the permissions seeder:

const defaultPermissions = [  
 // ... existing permissions ...  
 { name: 'custom:action', category: 'custom', description: 'Custom permission' },  
];

### Adding New Roles

Edit the roles seeder:

const defaultRoles = [  
 // ... existing roles ...  
 { name: 'custom', description: 'Custom role', isSystem: false, isDefault: false },  
];

### Modifying Role Permissions

Edit the role-permissions seeder:

const rolePermissionMap = {  
 admin: ['\*'], // All permissions  
 custom: ['users:read', 'custom:action'],  
 // ... other roles ...  
};

## 🔗 Integration

### With Other Tools

# 1. Seed database  
npm run db:seed all  
  
# 2. Verify with queries  
npm run db:query user-roles  
  
# 3. Clean up duplicates if needed  
npm run db:remove-duplicates --show

### With API Development

# Set up RBAC structure  
npm run db:seed permissions roles role-permissions  
  
# Start API server  
npm run dev  
  
# Test authentication with seeded users  
curl -X POST http://localhost:3001/api/v1/auth/login \  
 -H "Content-Type: application/json" \  
 -d '{"email":"admin@rasdash.com","password":"Admin123!"}'

This seeder provides a flexible, controlled way to set up your database with consistent, well-structured default data.