

# Índices de Base de Datos - Guía de Implementación

---

## Instrucciones

---

Esta guía contiene los índices optimizados que deben agregarse al archivo `prisma/schema.prisma`.

### Paso 1: Agregar índices a los modelos

Abre `prisma/schema.prisma` y agrega los siguientes bloques `@@index` a cada modelo **ANTES del cierre de llaves** `}`:

---

## User Model

---

```
model User {  
  // ... campos existentes ...  
  
  // Agregar al final, antes de }  
  @@index([email])  
  @@index([companyId])  
  @@index([role, companyId])  
  @@index([activo])  
  @@index([createdAt])  
  
  @@map("users")  
}
```

#### Beneficios:

- Búsquedas rápidas por email
  - Filtrado eficiente por compañía
  - Consultas optimizadas por rol
  - Filtrado por usuarios activos/inactivos
- 

## Building Model

---

```
model Building {  
  // ... campos existentes ...  
  
  // Agregar al final, antes de }  
  @@index([companyId])  
  @@index([tipo])  
  @@index([companyId, tipo])  
  @@index([createdAt])  
  
  @@map("buildings")  
}
```

**Beneficios:**

- Listado rápido de edificios por compañía
- Filtrado por tipo de edificio
- Consultas combinadas optimizadas

**Unit Model**

```
model Unit {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([buildingId])
  @@index([estado])
  @@index([buildingId, estado])
  @@index([tipo])
  @@index([companyId])

  @@map("units")
}
```

**Beneficios:**

- Unidades por edificio (consulta más frecuente)
- Filtrado rápido por estado (disponible, ocupada, etc.)
- Consultas combinadas edificio + estado

**Tenant Model**

```
model Tenant {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([companyId])
  @@index([email])
  @@index([dni])
  @@index([activo])
  @@index([createdAt])

  @@map("tenants")
}
```

**Beneficios:**

- Inquilinos por compañía
- Búsqueda rápida por email o DNI
- Filtrado por inquilinos activos

## Contract Model

```
model Contract {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([tenantId])
  @@index([unitId])
  @@index([estado])
  @@index([fechaInicio, fechaFin])
  @@index([companyId])
  @@index([estado, fechaFin])

  @@map("contracts")
}
```

### Beneficios:

- Contratos por inquilino
- Contratos por unidad
- Filtrado por estado
- Búsqueda por rangos de fechas
- Contratos activos y próximos a vencer

## Payment Model

```
model Payment {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([contractId])
  @@index([estado])
  @@index([fechaVencimiento])
  @@index([estado, fechaVencimiento])
  @@index([companyId])
  @@index([createdAt])

  @@map("payments")
}
```

### Beneficios:

- Pagos por contrato
- Filtrado por estado (pendiente, pagado, atrasado)
- Ordenamiento por fecha de vencimiento
- Pagos pendientes próximos a vencer

## MaintenanceRequest Model

```
model MaintenanceRequest {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([buildingId])
  @@index([estado])
  @@index([prioridad])
  @@index([buildingId, estado])
  @@index([estado, prioridad])
  @@index([companyId])
  @@index([createdAt])

  @@map("maintenance_requests")
}
```

### Beneficios:

- Mantenimientos por edificio
- Filtrado por estado y prioridad
- Consultas combinadas optimizadas

## Task Model

```
model Task {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([companyId])
  @@index([assignedToId])
  @@index([estado])
  @@index([prioridad])
  @@index([fechaVencimiento])
  @@index([companyId, estado])
  @@index([assignedToId, estado])

  @@map("tasks")
}
```

### Beneficios:

- Tareas por usuario asignado
- Filtrado por estado y prioridad
- Tareas próximas a vencer



## Notification Model

```
model Notification {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([userId])
  @@index([leida])
  @@index([userId, leida])
  @@index([createdAt])
  @@index([tipo])

  @@map("notifications")
}
```

### Beneficios:

- Notificaciones por usuario
- Filtrado rápido por leídas/no leídas
- Ordenamiento por fecha



## CalendarEvent Model

```
model CalendarEvent {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([companyId])
  @@index([userId])
  @@index([buildingId])
  @@index([fechaInicio, fechaFin])
  @@index([tipo])
  @@index([companyId, fechaInicio])

  @@map("calendar_events")
}
```

### Beneficios:

- Eventos por compañía
- Eventos por usuario o edificio
- Búsqueda eficiente por rangos de fechas

## Document Model

```
model Document {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([companyId])
  @@index([buildingId])
  @@index([unitId])
  @@index([tenantId])
  @@index([tipo])
  @@index([companyId, tipo])
  @@index([createdAt])

  @@map("documents")
}
```

### Beneficios:

- Documentos por entidad (building, unit, tenant)
- Filtrado por tipo de documento
- Búsqueda eficiente por compañía

## Expense Model

```
model Expense {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([companyId])
  @@index([buildingId])
  @@index([categoria])
  @@index([fecha])
  @@index([companyId, fecha])
  @@index([buildingId, categoria])

  @@map("expenses")
}
```

### Beneficios:

- Gastos por compañía o edificio
- Filtrado por categoría
- Búsqueda por rangos de fechas (reportes)



## AuditLog Model

```
model AuditLog {
  // ... campos existentes ...

  // Agregar al final, antes de }
  @@index([userId])
  @@index([companyId])
  @@index([createdAt])
  @@index([entityType, entityId])
  @@index([action])
  @@index([companyId, createdAt])

  @@map("audit_logs")
}
```

### Beneficios:

- Auditoría por usuario o compañía
- Búsqueda por tipo de entidad
- Consultas por rango de fechas (reportes de auditoría)



## Paso 2: Generar Migración

Después de agregar todos los índices:

```
# Generar migración
yarn prisma migrate dev --name add_performance_indexes

# Generar cliente de Prisma
yarn prisma generate
```



## Paso 3: Ejecutar Script de Optimización

```
yarn tsx scripts/optimize-database.ts
```

Este script ejecuta `ANALYZE` en PostgreSQL para actualizar las estadísticas del query planner.



## Paso 4: Verificar Impacto

### Antes de los índices:

```
EXPLAIN ANALYZE
SELECT * FROM units WHERE "buildingId" = 'xxx';
-- Seq Scan: 200ms
```

## Después de los índices:

```
EXPLAIN ANALYZE
SELECT * FROM units WHERE "buildingId" = 'xxx';
-- Index Scan: 5ms
```

## ⚠ Advertencias

1. **Espacio en disco:** Los índices ocupan espacio adicional (~10-20% del tamaño de las tablas)
2. **Escrituras:** Los índices ralentizan ligeramente las operaciones de INSERT/UPDATE (impacto mínimo)
3. **Mantenimiento:** PostgreSQL mantiene los índices automáticamente

## ✓ Checklist

- ☐ Agregar índices a User
- ☐ Agregar índices a Building
- ☐ Agregar índices a Unit
- ☐ Agregar índices a Tenant
- ☐ Agregar índices a Contract
- ☐ Agregar índices a Payment
- ☐ Agregar índices a MaintenanceRequest
- ☐ Agregar índices a Task
- ☐ Agregar índices a Notification
- ☐ Agregar índices a CalendarEvent
- ☐ Agregar índices a Document
- ☐ Agregar índices a Expense
- ☐ Agregar índices a AuditLog
- ☐ Ejecutar `yarn prisma migrate dev`
- ☐ Ejecutar `yarn tsx scripts/optimize-database.ts`
- ☐ Verificar mejoras de performance

## 📈 Mejoras Esperadas

- **Queries de listado:** 50-80% más rápidos
- **Búsquedas:** 70-90% más rápidas
- **Filtrados:** 60-85% más rápidos
- **Joins:** 40-60% más rápidos

¡Listo! Con estos índices, la base de datos estará optimizada para producción. 🚀