Apollo.ai Multi-Tenant SaaS Guide

🎯 Overview

Apollo.ai has been transformed into a **multi-tenant SaaS platform** that allows any business to connect their databases and start querying with natural language. This guide explains the architecture and how customers can get started.

🏗️ Architecture

How Multi-Tenancy Works

```
┌─────────────────────────────────────────┐
│         Apollo.ai Platform              │
│  (Your SaaS Application)                │
│                                         │
│  • User Authentication (Google SSO)     │
│  • Organization Management              │
│  • Database Connection Registry         │
│  • Query History & Learning             │
│  • Vector Search (Per-Tenant)           │
└─────────────────────────────────────────┘
         │          │          │
         ▼          ▼          ▼
     ┌────────┐ ┌────────┐ ┌────────┐
     │ Org A  │ │ Org B  │ │ Org C  │
     └────────┘ └────────┘ └────────┘
         │          │          │
     ┌───▼───┐  ┌───▼───┐  ┌───▼───┐
     │ DB 1  │  │ DB 2  │  │ DB 3  │
     │ DB 2  │  │ DB 3  │  │ DB 4  │
     └───────┘  └───────┘  └───────┘

    Company A  Company B  Company C
```

Key Components

1. **Organizations** - Each customer creates an organization
2. **Database Connections** - Organizations connect their databases
3. **Automatic Schema Discovery** - System learns each database structure
4. **Isolated Learning** - Query patterns are scoped per organization
5. **Secure Credentials** - All credentials encrypted at rest

---

🚀 Customer Onboarding Flow

Step 1: Sign Up

```
User signs up with Google SSO → Automatic organization created
```

Step 2: Add Database Connection

```
// Example: Adding a PostgreSQL database
{
  "name": "Production DB",
  "type": "postgresql",
  "host": "prod-db.company.com",
  "port": 5432,
  "database": "sales_db",
  "username": "readonly_user",
  "password": "encrypted_password",
  "ssl": true
}
```

Step 3: Automatic Schema Discovery

```
Apollo.ai automatically:
✓ Discovers all tables and columns
✓ Identifies relationships (foreign keys)
✓ Detects data types
✓ Caches schema for fast queries
✓ Syncs periodically for schema changes
```

Step 4: Start Querying

```
Natural Language → SQL → Results
"Show me top customers by revenue" → SELECT... → Data + Visualizations
```

📋 Supported Database Types

Database	Version	Status
PostgreSQL	9.6+	✅ Supported
MySQL	5.7+	✅ Supported
MariaDB	10.3+	✅ Supported
Oracle	11g+	✅ Supported
SQLite	3.x	✅ Supported
MongoDB	4.0+	🚧 Coming Soon

🔐 Security Features

Per-Tenant Isolation

- Each organization’s data is completely isolated
- Query history scoped by organization
- Vector embeddings tagged with `organizationId`
- No data leakage between tenants

Credential Encryption

```
// All database credentials encrypted at rest
const encryptedCredentials = encrypt({
  host, port, database, username, password, ssl
})
```

Role-Based Access Control

- **Owner**: Full control, can delete organization
- **Admin**: Manage members, databases, settings
- **Member**: Query databases, view history
- **Viewer**: Read-only access to queries and results

📊 Schema Learning System

How It Works

1. **Initial Discovery**
   `sql`
   ```
      -- Apollo.ai runs these queries on first connection
      SELECT table_name FROM information_schema.tables;
      SELECT column_name, data_type FROM information_schema.columns;
      SELECT constraint_name FROM information_schema.table_constraints;
   ```

2. **Relationship Detection**
   `typescript`
   ```
      // Automatically identifies foreign keys
      {
        "foreignKeys": [
          {
            "column": "customer_id",
            "referencedTable": "customers",
            "referencedColumn": "id"
          }
        ]
      }
   ```

3. **Sample Data Collection**
   `sql`

```
  -- Gets sample data for better query generation
  SELECT * FROM table_name LIMIT 3;
```

4. **Caching & Sync**
   - Schema cached in Apollo.ai database
   - Periodic refresh (daily) to catch schema changes
   - Manual refresh available via API

🎯 Query Intelligence (Per-Tenant)

Vector Database Integration

```
// Each query pattern is stored with organization context
{
  "organizationId": "org_abc123",
  "query": "Show me top customers",
  "sql": "SELECT * FROM customers ORDER BY total_spent DESC LIMIT 10",
  "embedding": [0.234, 0.567, ...], // Vector representation
  "confidence": 0.95
}
```

Learning Over Time

- Each organization builds its own "knowledge base"
- Similar queries get higher confidence
- Learns from corrections and feedback
- No cross-contamination between organizations

🛠️ API Reference

Organization Management

```
// Create Organization
POST /api/organizations
{
  "name": "Acme Corp"
}

// List User's Organizations
GET /api/organizations

// Get Organization Details
GET /api/organizations/{orgId}

// Add Team Member
POST /api/organizations/{orgId}/members
{
  "email": "colleague@company.com",
  "role": "MEMBER"
}
```

Database Connections

```
// Add Database Connection
POST /api/database-connections
{
  "organizationId": "org_abc123",
  "name": "Production DB",
  "type": "postgresql",
  "host": "db.company.com",
  "port": 5432,
  "database": "sales",
  "username": "readonly",
  "password": "secure_pass",
  "ssl": true
}

// Test Connection
POST /api/database-connections
{
  "action": "test",
  "organizationId": "org_abc123",
  ...connectionConfig
}

// List Connections
GET /api/database-connections?organizationId={orgId}

// Remove Connection
DELETE /api/database-connections?id={connectionId}
```

Query Execution

```
// Execute Query
POST /api/query
{
  "organizationId": "org_abc123",
  "query": "Show me sales by region",
  "database": "production_db"
}

// Response includes:
{
  "sql": "SELECT region, SUM(amount) ...",
  "results": [...],
  "summary": "Sales data grouped by region",
  "confidence": 0.92,
  "visualization": {...}
}
```

💼 Pricing Plans (Example)

Plan	Price	Features
Free	$0/mo	1 database, 100 queries/mo
Starter	$49/mo	3 databases, 1,000 queries/mo
Professional	$199/mo	10 databases, 10,000 queries/mo
Enterprise	Custom	Unlimited databases, custom SLA

🔄 Migration from Single-Tenant

If you have existing Apollo.ai installation:

```
# Run migration script
cd nextjs_space
yarn tsx scripts/migrate-to-multitenant.ts

# Results:
✅ Organizations created for all users
✅ All existing data linked to organizations
✅ System now multi-tenant ready!
```

📈 Scaling Considerations

Database Connection Pooling

- Each organization gets dedicated connection pools
- Configurable max connections per database
- Automatic cleanup of idle connections

Query Performance

- Schema cache reduces database roundtrips
- Vector search for fast query matching
- Result caching (optional)

Storage

- Query history: ~1KB per query
- Schema cache: ~100KB per database
- Vector embeddings: ~2KB per query pattern

🎨 UI Components (To Be Built)

Onboarding Wizard

```
Step 1: Create Organization → "Acme Corp"
Step 2: Add Database → Connection form
Step 3: Test Connection → ✅ Success
Step 4: Discover Schema → Automatic
Step 5: Start Querying → Dashboard
```

Organization Switcher

```
[Dropdown in header]
├─ Acme Corp (Current)
├─ Personal Workspace
└─ + New Organization
```

Database Management

```
Connected Databases
┌─────────────────────────────────┐
│ 🗄️ Production DB (PostgreSQL)   │
│    5 tables • Last sync: 2h ago│
│    [Refresh] [Edit] [Remove]    │
└─────────────────────────────────┘
```

🧪 Testing Multi-Tenancy

```
# Create two test organizations
curl -X POST /api/organizations \
  -H "Authorization: Bearer TOKEN_USER_1" \
  -d '{"name": "Company A"}'

curl -X POST /api/organizations \
  -H "Authorization: Bearer TOKEN_USER_2" \
  -d '{"name": "Company B"}'

# Add different databases to each
# Verify queries are isolated
# Confirm no data leakage
```

📞 Support & Documentation

- **API Docs**: `/docs/api`
- **Schema Discovery**: `/docs/schema-discovery`
- **Security**: `/docs/security`
- **Compliance**: `/docs/compliance`

🎉 Benefits of Multi-Tenant Apollo.ai

✅ **Easy Installation** - Just connect your database
✅ **No Code Required** - Natural language interface
✅ **Secure** - Enterprise-grade encryption
✅ **Scalable** - Supports multiple databases per org
✅ **Smart** - Learns your database over time
✅ **Compliant** - GDPR, SOC2 ready
✅ **Fast** - Cached schemas, optimized queries

🛣️ Roadmap

- [ ] Database connection wizard UI
- [ ] Organization switcher component
- [ ] Onboarding flow
- [ ] Billing integration (Stripe)
- [ ] Advanced role permissions
- [ ] Database connection health monitoring
- [ ] Schema change notifications
- [ ] Query optimization suggestions
- [ ] MongoDB support
- [ ] Real-time collaboration

Apollo.ai - Making databases accessible to everyone, everywhere.