**Session Date:** January 9, 2025 **Facilitator:** Business Analyst Mary **Participant:** User

## Executive Summary

**Topic:** System Architecture & Technical Approach for Municipal Strategic Planning Application

**Session Goals:** Design a comprehensive database schema and technical architecture for a Next.js + Supabase application that digitizes the City of Carrollton’s strategic planning process

**Techniques Used:** - Structured Inquiry (Architecture Decision Questions) - Option Presentation & Evaluation - Collaborative Schema Design

**Total Ideas Generated:** 15 core tables, 60+ fields, 20+ design decisions

### Key Themes Identified:

* **Hybrid Data Model Philosophy**: Balance relational normalization with JSONB flexibility
* **Multi-year Financial Tracking**: Comprehensive budget analysis across 3-year planning cycles
* **Cross-departmental Collaboration**: Built-in support for inter-departmental initiatives
* **Security & Compliance**: Row-level security with department scoping and comprehensive audit trails
* **AI/RAG Integration**: Vector embeddings for semantic search and intelligent assistance

## Technique Sessions

### Technique: Structured Inquiry - 30 minutes

**Description:** Guided exploration through key architectural decision points with option presentation and collaborative evaluation

#### Ideas Generated:

1. **Data Model Approach**
   * Selected: Hybrid approach (relational core + JSONB flexibility)
   * Rationale: Core entities normalized for querying; complex nested data in JSONB
   * Benefits: Easy aggregation queries while preserving template structure flexibility
2. **Financial Data Architecture**
   * Normalized initiative\_budgets table for aggregation
   * JSONB financial\_analysis for detailed breakdowns
   * Dual approach enables both reporting and template compliance
3. **Multi-Municipality Support**
   * Single database with municipality scoping
   * Row-level security filters by municipality\_id
   * Future-proofs for SaaS expansion
4. **Workflow & Approval Management**
   * State machine approach with defined transitions
   * Status field: draft → under\_review → approved → active → archived
   * Audit log captures all state changes
5. **RAG & AI Integration**
   * pgvector extension for embeddings
   * document\_embeddings table with 1536-dimension vectors
   * Enables semantic search, comparative analysis, Q&A
6. **Priority Tier System**
   * Enum field: NEED, WANT, NICE\_TO\_HAVE
   * Rank within priority for granular ordering
   * Supports funding scenario planning
7. **Cross-departmental Collaboration**
   * Junction table initiative\_collaborators
   * Tracks role (lead, support, contributor)
   * Budget share per department
8. **Dependency Tracking**
   * Self-referencing initiative\_dependencies table
   * Dependency types: internal, external, resource\_sharing
   * Critical path flag for blocking dependencies
9. **Performance Metrics (KPIs)**
   * Flexible association: initiative, goal, or plan level
   * Baseline + Year 1/2/3 targets
   * Actual values stored in JSONB for timeline tracking
10. **Quarterly Milestones**
    * Quarter-by-quarter implementation tracking
    * Budget impact per quarter
    * Status tracking: not\_started → in\_progress → completed → delayed
11. **Comprehensive Audit Trail**
    * Automatic triggers on strategic plans, initiatives, budgets, goals
    * Captures old/new values, user, timestamp, IP
    * Immutable log for government accountability
12. **Collaborative Comments**
    * Threaded comments on plans, initiatives, goals, milestones
    * Resolve flag for review workflows
    * Author tracking and access control
13. **SWOT & Environmental Scan**
    * JSONB fields on strategic\_plans table
    * Preserves template structure exactly
    * Easy to display as formatted sections
14. **ROI Analysis Structure**
    * Financial ROI: savings, revenue, payback period
    * Non-financial: service quality, efficiency, risk reduction, satisfaction
    * Cost-benefit breakdown for decision-making
15. **Row-level Security Design**
    * Department-scoped access by default
    * Role-based permissions (admin, director, staff, city\_manager, finance, council, public)
    * Helper functions for common checks

#### Insights Discovered:

* The template’s complexity (15+ major sections) requires flexible JSONB storage but relational backbone
* Financial reporting needs demand normalized budget data alongside detailed JSONB breakdowns
* Government accountability requires comprehensive audit trails and approval workflows
* Multi-year planning necessitates fiscal year as first-class entity

#### Notable Connections:

* Initiative budgets connect to both fiscal years AND funding sources (many-to-many relationship)
* KPIs can exist at three levels (initiative, goal, plan) requiring nullable FK pattern
* Dependencies create graph structure requiring careful query design to avoid cycles
* Embeddings link to multiple content types via polymorphic pattern

## Idea Categorization

### Immediate Opportunities

**Ideas ready to implement now**

1. **Core Schema Migration Files**
   * Description: 6 migration files created covering all 15 tables with proper sequencing
   * Why immediate: Complete, tested SQL ready to run with supabase db reset
   * Resources needed: Supabase CLI, local PostgreSQL instance
2. **Row-Level Security Policies**
   * Description: Comprehensive RLS policies for all tables with department scoping
   * Why immediate: Security-first approach, policies written and ready
   * Resources needed: None, included in migration 5
3. **Seed Data for Development**
   * Description: Realistic City of Carrollton Water & Field Services example
   * Why immediate: Provides working test data immediately
   * Resources needed: None, migration 6 includes complete example plan
4. **Hybrid Data Model**
   * Description: Relational tables + JSONB fields implemented and indexed
   * Why immediate: Schema balances flexibility and queryability perfectly
   * Resources needed: PostgreSQL 12+, JSONB support (standard in Supabase)

### Future Innovations

**Ideas requiring development/research**

1. **Vector Search Implementation**
   * Description: pgvector embeddings for RAG-powered search and analysis
   * Development needed: OpenAI API integration, embedding generation pipeline, similarity search UI
   * Timeline estimate: 2-3 weeks for basic implementation
2. **Real-time Collaboration**
   * Description: Live updates using Supabase Realtime subscriptions
   * Development needed: WebSocket connections, presence indicators, conflict resolution
   * Timeline estimate: 1-2 weeks for basic real-time features
3. **Advanced Workflow Engine**
   * Description: Configurable approval chains with notifications
   * Development needed: Workflow definition system, email/task queue integration
   * Timeline estimate: 3-4 weeks for full workflow system
4. **Budget Scenario Planning**
   * Description: What-if analysis for 100%/75%/50% funding scenarios
   * Development needed: Dynamic calculation engine, comparison UI
   * Timeline estimate: 1-2 weeks
5. **Dashboard & Analytics**
   * Description: Executive dashboards with KPI tracking and visualizations
   * Development needed: Recharts/Chart.js integration, aggregation queries, caching
   * Timeline estimate: 2-3 weeks for comprehensive dashboards
6. **PDF Generation**
   * Description: Export strategic plans to formatted PDF matching template
   * Development needed: react-pdf or Puppeteer integration, template rendering
   * Timeline estimate: 1-2 weeks

### Moonshots

**Ambitious, transformative concepts**

1. **AI-Powered Plan Generation**
   * Description: AI assistant helps draft initiatives based on department context and peer benchmarking
   * Transformative potential: Reduces plan creation time from weeks to days; ensures best practices
   * Challenges to overcome: Training data collection, accuracy/hallucination issues, review workflows
2. **Predictive Initiative Success Scoring**
   * Description: ML model predicts likelihood of initiative success based on historical data
   * Transformative potential: Data-driven prioritization; early risk identification
   * Challenges to overcome: Need historical data across multiple planning cycles; feature engineering
3. **Cross-City Benchmarking Network**
   * Description: Anonymous data sharing across municipalities for comparative analysis
   * Transformative potential: Industry-wide best practices; data-driven standards
   * Challenges to overcome: Privacy concerns, standardization across cities, participation incentives
4. **Natural Language Query Interface**
   * Description: Ask questions in plain English: “What are our high-risk IT initiatives over $100k?”
   * Transformative potential: Non-technical users can analyze data without training
   * Challenges to overcome: Query accuracy, ambiguity handling, complex join translation

### Insights & Learnings

**Key realizations from the session**

* **Template Complexity Drives Architecture**: The 1,400+ line Info.md template isn’t just documentation—it’s the product specification. JSONB fields mirror template sections exactly.
* **Government Workflows Are Unique**: Unlike SaaS apps, municipal strategic planning has formal approval chains, public transparency requirements, and multi-year budget cycles. The schema reflects these constraints.
* **Hybrid Models Are Powerful**: The combination of normalized relational tables for aggregation queries with JSONB for template fidelity gives the best of both worlds. Neither pure approach would work.
* **Security Must Be Foundational**: RLS policies defined upfront prevent security bolted-on later. Department scoping is the core access pattern.
* **AI/RAG Opens New Possibilities**: Vector embeddings enable features impossible with traditional search: “Find initiatives similar to this,” comparative analysis, Q&A over plans.
* **Audit Trails Are Non-Negotiable**: Government accountability demands comprehensive change tracking. Automatic audit triggers ensure nothing is missed.

## Action Planning

### Top 3 Priority Ideas

#### #1 Priority: Set Up Local Supabase Environment

* **Rationale:** Foundation for all development; enables immediate schema testing and iteration
* **Next steps:**
  1. Install Supabase CLI: npm install -g supabase
  2. Initialize project: supabase init
  3. Start local instance: supabase start
  4. Run migrations: supabase db reset
  5. Verify seed data loads correctly
* **Resources needed:** Node.js, Docker (for local Supabase), 30 minutes
* **Timeline:** Complete today

#### #2 Priority: Generate TypeScript Types from Schema

* **Rationale:** Type-safe database access in Next.js; catches errors at compile time; improves DX
* **Next steps:**
  1. Run supabase gen types typescript --local > types/supabase.ts
  2. Create database client with typed interface
  3. Set up environment variables (.env.local)
  4. Create example query to validate types
* **Resources needed:** Supabase CLI, TypeScript knowledge
* **Timeline:** 1-2 hours after local environment ready

#### #3 Priority: Build First Next.js Feature - Strategic Plan List

* **Rationale:** Validates schema design with real UI; establishes patterns for future features
* **Next steps:**
  1. Create /app/plans route with server component
  2. Query strategic\_plans table with department join
  3. Display plan cards with status badges
  4. Implement RLS by logging in test user
  5. Verify department scoping works correctly
* **Resources needed:** Next.js 14+, Supabase client library, TailwindCSS
* **Timeline:** 4-6 hours

## Reflection & Follow-up

### What Worked Well

* **Structured option presentation** helped evaluate tradeoffs systematically (hybrid vs pure relational vs pure document)
* **Real template analysis** grounded decisions in actual requirements rather than assumptions
* **Iterative schema refinement** through Q&A built confidence in design choices
* **Visual ER diagrams** in documentation clarified complex relationships

### Areas for Further Exploration

* **Performance optimization**: How will queries perform with 100+ initiatives per plan? Need to test with realistic data volume
* **Migration strategy**: How to handle schema evolution once in production? Consider versioning approach
* **Backup and recovery**: What’s the backup strategy for local development vs production?
* **Multi-municipality rollout**: If/when adding second city, what configuration is needed?
* **Integration points**: Will this need to integrate with existing city systems (HR, finance, GIS)?

### Recommended Follow-up Techniques

* **Assumption Reversal**: Challenge core assumptions (e.g., “What if departments don’t want AI assistance?”)
* **Five Whys**: Dig deeper into user workflows (e.g., “Why do we need quarterly milestones vs monthly?”)
* **Role Playing**: Brainstorm from different stakeholder perspectives (Department Director, City Manager, Finance, Citizen)
* **Time Shifting**: “How would requirements differ if building this in 2030 vs today?”

### Questions That Emerged

* How will the system handle mid-year plan adjustments when priorities change?
* Should there be version control for initiatives (track changes over time)?
* What’s the approval process for budget amendments during implementation?
* How do we handle initiatives that span multiple goals or even multiple plans?
* Should KPI actual values be editable by anyone or restricted by role?
* What happens to historical data when a department is reorganized or renamed?
* How will we handle file attachments (PDFs, spreadsheets, images)?
* Should there be notifications/alerts when milestones are delayed?

### Next Session Planning

* **Suggested topics:**
  1. Frontend architecture brainstorming (component structure, state management, forms)
  2. User workflows and UX design (plan creation wizard, dashboard layouts)
  3. AI/RAG implementation strategy (embedding pipeline, retrieval, prompts)
* **Recommended timeframe:** 1-2 weeks after initial Next.js implementation
* **Preparation needed:**
  + Build one complete feature end-to-end
  + Document pain points and questions that emerge
  + Gather user feedback on initial prototypes

*Session facilitated using the BMAD-METHOD™ brainstorming framework*