Avenir AI Solutions - Database Documentation

Last Updated: October 22, 2025
Total Tables: 24
Database: PostgreSQL (Supabase + Neon Failover)

📋 Table of Contents

1. Client Management (1 table)
2. Lead Processing (3 tables)
3. AI Intelligence & Learning (5 tables)
4. Prospect Discovery (4 tables)
5. Outreach Automation (5 tables)
6. Translation System (2 tables)
7. Background Processing (1 table)
8. Supporting Tables (3 tables)

---

1. Client Management

`clients`

Purpose: Stores all client accounts, API keys, and configuration settings.

What It Does:

- Manages client registration and authentication
- Stores unique API keys for form integration
- Tracks client branding preferences (email tone, booking links, industry)
- Monitors connection health (last login, last lead received)
- Supports bilingual clients (EN/FR)

Key Fields:

- `id` - Internal UUID
- `client_id` - Public client identifier
- `api_key` - Unique key for API authentication
- `business_name` - Client's business name
- `email` - Client login email
- `password_hash` - Encrypted password
- `language` - Preferred language (en/fr)
- `email_tone` - Email style (Professional/Friendly/Formal/Energetic)

- `ai_personalized_reply` - Toggle for automated email responses
- `booking_link` - Custom booking calendar link
- `last_connection` - When last lead was received
- `is_test` - Marks test/demo accounts
- `is_internal` - Marks internal Avenir accounts

Row Count: ~5-10 clients

---

2. Lead Processing

`lead_memory`

Purpose: Central storage for all incoming leads with AI enrichment data.

What It Does:

- Captures every lead from client contact forms
- Stores AI analysis (intent, tone, urgency, confidence)
- Tracks status changes (archived, deleted, tagged)
- Maintains history of AI predictions over time
- Links leads to specific clients

Key Fields:

- `id` - Unique lead identifier
- `name` - Lead's name
- `email` - Lead's email
- `message` - Original lead message
- `ai_summary` - AI-generated summary
- `intent` - AI-detected intent (e.g., "B2B partnership")
- `tone` - AI-detected tone (e.g., "Professional")
- `urgency` - AI-detected urgency (High/Medium/Low)
- `confidence_score` - AI confidence (0.00-1.00)
- `tone_history` - JSONB array of tone changes
- `confidence_history` - JSONB array of confidence changes
- `urgency_history` - JSONB array of urgency changes
- `archived` - Lead archived status
- `deleted` - Lead deleted status
- `current_tag` - Current tag (e.g., "Hot Lead", "Converted")
- `client_id` - Which client this lead belongs to
- `language` - Lead language (en/fr)
- `is_test` - Marks test leads

Row Count: Growing (100-1000+ leads per client)

`lead_actions`

Purpose: Audit trail of all actions performed on leads.

What It Does:

- Logs every lead action (tag, archive, delete, reactivate)
- Tracks conversion events
- Records reversion reasons (if converted lead reverted)
- Powers the Activity Log in client dashboard
- Enables client-specific analytics

Key Fields:

- `id` - Action UUID
- `lead_id` - Which lead this action applies to
- `client_id` - Which client performed the action
- `action_type` - Type of action ("tagged", "archived", "deleted", "reactivated", "created")
- `tag` - Tag applied (if action is tagging)
- `timestamp` - When action occurred
- `conversion_outcome` - TRUE if lead converted
- `reversion_reason` - Why converted lead was reverted
- `is_test` - Marks test actions

Row Count: Growing (multiple actions per lead)

Note: The column is `action_type` in database, mapped to `action` in frontend.

`lead_notes`

Purpose: Stores user-added notes for leads.

What It Does:

- Allows clients to add notes to leads
- Tracks who added the note
- Maintains note history
- Supports expandable notes UI in dashboard

Key Fields:

- `id` - Note UUID
- `lead_id` - Which lead this note belongs to
- `client_id` - Which client created the note
- `note` - The note content
- `performed_by` - Who created the note (usually "admin")
- `created_at` - When note was created
- `updated_at` - When note was last modified

- `is_test` - Marks test notes

Row Count: Variable (0-10+ notes per lead)

---

3. AI Intelligence & Learning

`growth_brain`

Purpose: Stores AI-generated growth insights and analytics for each client.

What It Does:

- Analyzes ALL active leads for each client
- Identifies top intents, urgency patterns, tone distributions
- Calculates engagement scores
- Generates predictive insights (bilingual)
- Updates weekly or on-demand via "Growth Copilot"

Key Fields:

- `id` - Insight UUID
- `client_id` - Which client this analysis is for
- `analysis_period_start` - Analysis time range start
- `analysis_period_end` - Analysis time range end
- `total_leads` - How many leads analyzed
- `top_intents` - JSONB array of top 5 intents
- `urgency_distribution` - JSONB: {high, medium, low} counts
- `urgency_trend_percentage` - Week-over-week urgency change
- `tone_distribution` - JSONB array of tone frequencies
- `tone_sentiment_score` - 0-100 professional signal score
- `avg_confidence` - Average AI confidence across all leads
- `confidence_trajectory` - JSONB array tracking confidence over time
- `language_ratio` - JSONB: {en, fr} percentages
- `engagement_score` - Composite metric (0-100)
- `predictive_insights` - JSONB: {en: {}, fr: {}} with recommendations
- `analyzed_at` - When analysis was performed

Row Count: 1 row per client (UPSERT on `client_id`)

Unique Constraint: `growth_brain_client_unique` prevents duplicate client records

`feedback_tracking`

Purpose: Tracks AI outcomes and user feedback for continuous learning.

What It Does:

- Logs every AI prediction and its outcome
- Compares predicted vs. actual results
- Tracks positive/negative/neutral outcomes
- Powers the adaptive learning system
- Expires after 1 year (auto-cleanup)

Key Fields:

- `id` - Feedback UUID
- `lead_id` - Related lead (optional)
- `client_id` - Related client (optional)
- `action_type` - Feedback category: `lead_conversion`, `email_response`, `user_action`, `system_performance`
- `outcome` - Result: `positive`, `negative`, `neutral`
- `confidence_score` - Confidence in this feedback (0.0-1.0)
- `impact_score` - Impact on system (-100 to +100)
- `context_data` - JSONB with action-specific details
- `notes` - Human-readable description
- `notes_en` - English notes
- `notes_fr` - French notes
- `learning_applied` - Has this been processed by learning algorithms?
- `learning_impact` - Impact on AI models (0.0-1.0)
- `processed_at` - When feedback was processed
- `expires_at` - Auto-cleanup date (NOW + 1 year)

Row Count: Growing (every AI prediction logged)

`performance_metrics`

Purpose: Tracks system performance, AI accuracy, and response times.

What It Does:

- Monitors API response times
- Tracks AI analysis accuracy
- Measures success rates
- Detects error patterns
- Enables performance optimization

Key Fields:

- `id` - Metric UUID
- `event_type` - Category: `api_response`, `ai_analysis`, `translation`, `lead_processing`, `email_response`
- `metric_name` - Specific metric: `response_time`, `accuracy`, `success_rate`, `error_rate`
- `metric_value` - The actual measurement

- `metric_unit` - Unit: `ms`, `percent`, `count`, `score`
- `response_time_ms` - Response time in milliseconds
- `success_rate` - Success percentage (0.0-1.0)
- `ai_accuracy` - AI accuracy score (0.0-1.0)
- `error_count` - Number of errors
- `source_component` - Which component logged this: `lead_api`, `translation_service`, `intelligence_engine`
- `client_id` - Client-specific metrics (optional)
- `recorded_at` - When metric was recorded
- `metadata` - JSONB with additional context
- `error_message_en` - English error message
- `error_message_fr` - French error message

Row Count: Growing (continuous logging)

`prompt_registry`

Purpose: Stores all AI prompt variants with performance scores.

What It Does:

- Tracks different versions of AI prompts
- Scores prompt performance
- Selects best-performing prompts
- Enables A/B testing
- Powers prompt evolution system

Key Fields:

- `id` - Prompt UUID
- `prompt_name` - Prompt identifier
- `variant_version` - Version number
- `variant_id` - Unique variant identifier
- `prompt_text` - The actual prompt content
- `language` - Prompt language (en/fr)
- `score` - Performance score (0.0-1.0)
- `usage_count` - How many times used
- `created_at` - When created
- `updated_at` - When last updated

Row Count: Growing (multiple variants per prompt)

`prompt_performance`

Purpose: Tracks individual AI prompt execution results.

What It Does:

- Logs every prompt execution
- Measures quality, accuracy, consistency
- Tracks token usage and costs
- Links to feedback for learning
- Enables prompt optimization

Key Fields:

- `id` - Execution UUID
- `prompt_registry_id` - Which prompt variant was used
- `execution_id` - Unique execution identifier
- `client_id` - Client-specific tracking (optional)
- `input_data` - JSONB input to the prompt
- `output_data` - JSONB output from the prompt
- `output_quality_score` - Quality rating (0.000-1.000)
- `response_time_ms` - How fast the prompt executed
- `token_count` - Tokens used
- `cost_usd` - Cost in USD
- `accuracy_score` - Accuracy rating
- `consistency_score` - Format compliance score
- `completeness_score` - Completeness rating
- `error_occurred` - Did an error happen?
- `error_message` - Error details
- `feedback_id` - Link to feedback record
- `user_rating` - User rating 1-5 (optional)
- `executed_at` - When executed
- `environment` - `production`, `staging`, `test`

Row Count: Growing (every prompt execution logged)

`prompt_ab_tests`

Purpose: Manages A/B testing of prompt variants.

What It Does:

- Configures A/B tests for prompts
- Allocates traffic between control/treatment
- Tracks test results and statistical significance
- Selects winning prompt variants
- Automates prompt optimization

Key Fields:

- `id` - Test UUID

- `test_name` - Test identifier
- `prompt_name` - Which prompt is being tested
- `control_variant_id` - Control prompt variant
- `treatment_variant_id` - Treatment prompt variant
- `control_traffic_percentage` - Control traffic allocation
- `treatment_traffic_percentage` - Treatment traffic allocation
- `min_sample_size` - Minimum samples before concluding (default: 100)
- `max_duration_days` - Max test duration (default: 7 days)
- `significance_level` - Statistical significance (default: 0.05)
- `status` - `draft`, `running`, `completed`, `cancelled`
- `control_metrics` - JSONB control group results
- `treatment_metrics` - JSONB treatment group results
- `statistical_significance` - P-value
- `winner_variant_id` - Winning prompt

Row Count: ~10-50 tests

`prompt_evolution`

Purpose: Tracks how AI prompts evolve and improve over time.

What It Does:

- Records parent-child prompt relationships
- Logs evolution strategies (mutation, crossover, optimization)
- Compares performance improvements
- Links feedback to prompt changes
- Documents evolution history

Key Fields:

- `id` - Evolution UUID
- `parent_prompt_id` - Original prompt
- `child_prompt_id` - Evolved prompt
- `evolution_type` - `mutation`, `crossover`, `optimization`, `manual_edit`
- `evolution_strategy` - Specific strategy used
- `parent_performance` - JSONB parent metrics
- `child_performance` - JSONB child metrics
- `improvement_score` - Measured improvement (0.000-1.000)
- `feedback_data` - JSONB feedback that drove evolution
- `optimization_goals` - Array of goals
- `evolved_at` - When evolution occurred
- `evolution_algorithm` - Algorithm used

Row Count: Growing (tracks all prompt improvements)

---

4. Prospect Discovery

`prospect_candidates`

Purpose: Stores discovered prospects from People Data Labs, Google, Apollo.

What It Does:

- Holds all discovered prospect companies
- Tracks automation need scores (45-95 range)
- Monitors contact status
- Links to outreach logs and form tests
- Powers daily prospect queue

Key Fields:

- `id` - Prospect UUID
- `business_name` - Company name
- `website` - Company website (unique)
- `contact_email` - Contact email
- `industry` - Industry category
- `region` - Geographic region
- `language` - Target language (en/fr)
- `form_url` - Contact form URL
- `last_tested` - Last form test date
- `response_score` - Response quality score
- `automation_need_score` - AI-calculated automation need (45-95)
- `contacted` - Has been contacted?
- `metadata` - JSONB with discovery source, LinkedIn data, etc.
- `created_at` - When discovered
- `updated_at` - Last update

Row Count: 34,823+ prospects from PDL taxonomy mapping

`prospect_outreach_log`

Purpose: Tracks all outreach emails sent to prospects.

What It Does:

- Logs every outreach email
- Tracks engagement (opened, replied)
- Monitors email status
- Stores reply content
- Enables outreach performance analysis

Key Fields:

- `id` - Log UUID
- `prospect_id` - Which prospect was contacted
- `subject` - Email subject line
- `email_body` - Email content
- `sent_at` - When email was sent
- `opened_at` - When email was opened
- `replied_at` - When prospect replied
- `status` - `sent`, `opened`, `replied`, `bounced`, `ignored`
- `reply_content` - Prospect's reply
- `metadata` - JSONB with campaign info

Row Count: Growing (one row per outreach email)

`prospect_industry_performance`

Purpose: Aggregates prospect performance by industry for learning.

What It Does:

- Tracks which industries respond best
- Calculates open rates and reply rates per industry
- Measures average response times
- Assigns priority scores to industries
- Optimizes targeting based on performance

Key Fields:

- `id` - Performance UUID
- `industry` - Industry name (unique)
- `total_contacted` - Total outreach count
- `total_opened` - Total emails opened
- `total_replied` - Total replies received
- `open_rate` - Open percentage
- `reply_rate` - Reply percentage
- `avg_response_time_hours` - Average time to reply
- `priority_score` - Industry priority (0-100)
- `last_updated` - Last calculation date

Row Count: ~10-20 industries

`prospect_form_tests`

Purpose: Records automated form testing results.

What It Does:

- Tests prospect contact forms
- Measures response quality
- Detects autoresponders
- Scores autoresponder personalization
- Validates form functionality

Key Fields:

- `id` - Test UUID
- `prospect_id` - Which prospect was tested
- `test_submitted_at` - When form test was submitted
- `response_received_at` - When response arrived
- `response_time_minutes` - Response delay
- `has_autoresponder` - Did they have an autoresponder?
- `autoresponder_tone` - `robotic`, `human`, `personalized`, `none`
- `autoresponder_content` - The autoresponder message
- `score` - Quality score (0-100)
- `test_status` - `pending`, `completed`, `failed`, `timeout`
- `metadata` - JSONB with test details

Row Count: Growing (one row per form test)

---

5. Outreach Automation

```
outreach_campaigns
```

Purpose: Manages outreach campaigns for prospect engagement.

What It Does:

- Organizes outreach into campaigns
- Links campaigns to clients
- Tracks campaign status
- Stores target criteria
- Manages follow-up schedules

Key Fields:

- `id` - Campaign UUID
- `name` - Campaign name
- `client_id` - Which client owns this campaign
- `status` - `draft`, `active`, `paused`, `completed`
- `target_criteria` - JSONB filtering rules
- `email_template_id` - Which template to use
- `follow_up_schedule` - JSONB follow-up timing
- `created_at` - Campaign creation date

- `updated_at` - Last update
- `metadata` - JSONB campaign settings

Row Count: Growing (~5-20 campaigns per client)

`outreach_emails`

Purpose: Stores individual outreach emails with tracking.

What It Does:

- Generates personalized outreach emails
- Tracks email status (pending → sent → opened → replied)
- Links to prospects and campaigns
- Monitors Gmail delivery
- Handles follow-up sequences

Key Fields:

- `id` - Email UUID
- `campaign_id` - Which campaign this email belongs to
- `prospect_id` - Target prospect
- `prospect_email` - Prospect's email address
- `prospect_name` - Prospect's name
- `company_name` - Company name
- `website` - Company website
- `template_id` - Email template used
- `subject` - Email subject
- `content` - Email body (HTML)
- `status` - `pending`, `approved`, `rejected`, `sent`, `delivered`, `opened`, `replied`, `bounced`
- `sent_at` - When sent
- `opened_at` - When opened
- `replied_at` - When replied
- `thread_id` - Gmail thread ID
- `gmail_message_id` - Gmail message ID
- `follow_up_sequence` - Follow-up number (1, 2, 3...)
- `sender_email` - From email address
- `missing_email` - Flag for prospects without email
- `metadata` - JSONB tracking details

Row Count: Growing (one row per email)

`outreach_tracking`

Purpose: Tracks granular email engagement events.

What It Does:

- Logs every email action (sent, delivered, opened, clicked, replied)
- Timestamps each event
- Links events to emails, prospects, campaigns
- Enables detailed engagement analytics

Key Fields:

- `id` - Event UUID
- `email_id` - Which email this event is for
- `prospect_id` - Related prospect
- `campaign_id` - Related campaign
- `action` - Event type: `sent`, `delivered`, `opened`, `clicked`, `replied`, `bounced`, `unsubscribed`
- `timestamp` - When event occurred
- `metadata` - JSONB event details

Row Count: Growing (multiple events per email)

`outreach_metrics`

Purpose: Aggregates outreach campaign performance metrics.

What It Does:

- Calculates campaign-level statistics
- Tracks total emails sent, opened, replied
- Computes open rates and reply rates
- Monitors revenue generated from campaigns
- Enables campaign optimization

Key Fields:

- `id` - Metrics UUID
- `campaign_id` - Which campaign these metrics are for
- `total_emails_sent` - Total sent count
- `total_emails_delivered` - Successfully delivered
- `total_emails_opened` - Emails opened
- `total_emails_clicked` - Links clicked
- `total_emails_replied` - Replies received
- `total_emails_bounced` - Bounced emails
- `total_emails_unsubscribed` - Unsubscribes
- `open_rate` - Open percentage
- `click_rate` - Click percentage
- `reply_rate` - Reply percentage

- `bounce_rate` - Bounce percentage
- `revenue_generated` - USD revenue from campaign
- `created_at` - When metrics started tracking
- `updated_at` - Last update

Row Count: 1 row per campaign (updated continuously)

`email_templates`

Purpose: Stores reusable email templates for outreach.

What It Does:

- Manages email templates
- Supports personalization variables
- Tracks template performance
- Enables A/B testing
- Powers automated outreach

Key Fields:

- `id` - Template UUID
- `name` - Template name
- `description` - Template description
- `subject` - Email subject (with variables)
- `html_content` - HTML email body
- `plain_content` - Plain text email body
- `language` - Template language (en/fr)
- `category` - Template type
- `variables` - JSONB list of personalization variables
- `is_active` - Template enabled?
- `usage_count` - Times used
- `performance_score` - Performance rating (0.0-1.0)
- `created_at` - When created
- `updated_at` - Last update

Row Count: ~10-30 templates

---

6. Translation System

`translation_cache`

Purpose: Caches AI translations to reduce API costs.

What It Does:

- Stores translated text for reuse
- Prevents duplicate translation API calls
- Speeds up bilingual responses
- Tracks translation quality
- Expires old translations (90 days)

Key Fields:

- `id` - Cache UUID
- `source_text` - Original text
- `source_language` - Source language (en/fr)
- `target_language` - Target language (fr/en)
- `translated_text` - Translated result
- `context_type` - Context: `lead_summary`, `intent`, `tone`, `ui_text`
- `quality_score` - Translation quality (0.0-1.0)
- `verified` - Human-verified?
- `created_at` - When translated
- `last_used_at` - Last usage
- `expires_at` - Expiration date (NOW + 90 days)

Row Count: Growing (cached translations)

`translation_dictionary`

Purpose: Stores verified translations for common terms.

What It Does:

- Maps specific terms to verified translations
- Ensures consistent terminology
- Provides instant lookups (no AI needed)
- Supports fuzzy matching
- Bilingual glossary

Key Fields:

- `id` - Dictionary UUID
- `source_text` - Original term
- `source_language` - Source language
- `target_language` - Target language
- `translated_text` - Verified translation
- `context` - Usage context
- `category` - Term category
- `verified` - Human-verified?
- `usage_count` - Times used
- `confidence` - Translation confidence (0.0-1.0)
- `created_at` - When added

- `updated_at` - Last update

Row Count: ~50-200 verified terms

---

7. Background Processing

`queue_jobs`

Purpose: Background job queue for long-running operations.

What It Does:

- Queues asynchronous jobs (daily prospect discovery, bulk operations)
- Processes jobs with 300-second timeout (vs. 60s API limit)
- Tracks job status (pending → processing → completed/failed)
- Stores job results and errors
- Enables reliable background processing

Key Fields:

- `id` - Job UUID
- `job_type` - Job category: `daily_prospect_queue`, `bulk_email`, `intelligence_analysis`
- `status` - `pending`, `processing`, `completed`, `failed`
- `payload` - JSONB input data for job
- `result` - JSONB output data after completion
- `error` - Error message if failed
- `created_at` - When job was queued
- `started_at` - When processing began
- `completed_at` - When job finished

Row Count: Growing (cleared periodically after completion)

---

8. Supporting Tables

`integration_logs`

Purpose: Logs all API integrations and form submissions.

What It Does:

- Tracks every API request from client forms
- Logs request/response data
- Monitors integration health
- Debugs integration issues
- Validates API key usage

Key Fields:

- `id` - Log UUID
- `client_id` - Which client's integration
- `api_key` - API key used (partial)
- `endpoint` - API endpoint called
- `method` - HTTP method
- `request_body` - JSONB request data
- `response_status` - HTTP status code
- `response_body` - JSONB response data
- `ip_address` - Request IP
- `user_agent` - Request user agent
- `created_at` - When logged

Row Count: Growing (every API call logged)

`avenir_profile_embeddings`

Purpose: Stores AI embeddings of Avenir's company profile.

What It Does:

- Holds vector embeddings of company description
- Enables semantic matching with prospects
- Powers similarity scoring
- Optimizes prospect targeting

Key Fields:

- `id` - Embedding UUID
- `chunk_text` - Text chunk from company profile
- `embedding` - Vector embedding (1536 dimensions)
- `embedding_model` - Model used (e.g., text-embedding-ada-002)
- `metadata` - JSONB chunk metadata
- `created_at` - When created

Row Count: ~10-50 chunks (company profile broken into chunks)

`intent_translations`

Purpose: Maps lead intents across English and French.

What It Does:

- Ensures consistent intent classification across languages
- Provides English ↔ French intent translations
- Powers bilingual analytics

- Maintains intent taxonomy

Key Fields:

- `id` - Translation UUID
- `intent_en` - English intent
- `intent_fr` - French intent
- `category` - Intent category
- `created_at` - When added

Row Count: ~20-50 intent mappings

📊 Database Statistics

Total Tables: 24

By Category:

- Client Management: 1 table
- Lead Processing: 3 tables
- AI Intelligence & Learning: 5 tables
- Prospect Discovery: 4 tables
- Outreach Automation: 5 tables
- Translation System: 2 tables
- Background Processing: 1 table
- Supporting Tables: 3 tables

Storage Breakdown:

- **Lead Data:** 70-80% (lead_memory, lead_actions, lead_notes)
- **Prospect Data:** 10-15% (prospect_candidates, outreach_emails)
- **Learning Data:** 5-10% (feedback_tracking, performance_metrics, prompt_performance)
- **Supporting Data:** 5% (caches, logs, translations)

Growth Rate:

- **High Growth:** lead_memory, lead_actions, feedback_tracking, performance_metrics
- **Medium Growth:** prospect_candidates, outreach_emails, prompt_performance
- **Low Growth:** prompt_registry, translation_dictionary, intent_translations
- **Static/Slow:** email_templates, growth_brain (1 per client)

🔐 Security & Access Control

Row Level Security (RLS): Enabled on ALL tables

Policy Types:

1. **Client Isolation:** Clients can only see their own data
2. **Service Role:** Full access for system operations
3. **Public Read:** Some tables allow public read for demos
4. **Authenticated Write:** Most writes require authentication

Key Security Features:

- API key authentication
- Encrypted password hashes
- Client-specific data isolation
- Audit trails (lead_actions, integration_logs)
- Automatic test data flagging (`is_test` column)

📈 Key Database Features

1. Multi-Tenant Architecture

- Every table links to `client_id`
- RLS enforces strict data isolation
- Clients see only their own leads/actions/notes

2. Adaptive Learning Infrastructure

- `feedback_tracking` + `performance_metrics` = Learning loops
- `prompt_registry` + `prompt_performance` = Prompt optimization
- `growth_brain` = Weekly intelligence updates

3. JSONB Flexibility

- `metadata` fields in most tables for extensibility
- `context_data` for learning system
- `predictive_insights` for bilingual recommendations
- History tracking arrays (tone_history, confidence_history)

4. Bilingual Support

- `notes_en` + `notes_fr` columns
- `language` columns throughout
- `intent_translations` mapping table
- Bilingual insights in `growth_brain`

5. Performance Optimizations

- Comprehensive indexing on all query columns
- Timestamp indexes for sorting
- Composite indexes for common queries
- Foreign key indexes for joins

🔄 Data Flow

Lead Capture Flow:

```
Client Form → /api/lead → lead_memory (insert)
                        → AI Enrichment
                        → lead_actions (log "created")
                        → feedback_tracking (log prediction)
                        → Email Response
```

Learning Loop Flow:

```
AI Prediction → feedback_tracking (log)
              → performance_metrics (log speed, accuracy)
              → Actual Outcome
              → Compare & Score
              → prompt_performance (log result)
              → Weekly Analysis
              → growth_brain (update insights)
              → Better Future Predictions
```

Prospect Discovery Flow:

```
ICP Definition → PDL/Google/Apollo Search
               → prospect_candidates (insert 16+ daily)
               → automation_need_score calculation
               → prospect_outreach_log (email sent)
               → outreach_tracking (engagement events)
               → prospect_industry_performance (update stats)
```

Documentation Prepared By: AI Growth Infrastructure Team
Database Type: PostgreSQL (Supabase Primary, Neon Failover)
Total Tables: 24
Last Updated: October 22, 2025