Contents

RADIANT Database Schema Reference

Complete Migration Inventory (40 Migrations)

Migration Index

#	Migration	Tables Created	Purpose
001	initial_schema	tenants, users, administrators, invitations, approval_requests	Core platform tables
002	tenant_isolation	RLS policies	Row-level security
003	ai_models	ai_models, model_capabilities	Model registry
004	usage_billing	usage_records, invoices	Usage tracking
005	admin_approval	approval_workflows	Admin approvals

#	Migration	Tables Created	Purpose
006	self_hosted_models	self_hosted_models, model_deployments	Self-hosted AI
007	external_providers	external_providers, provider_configs	Provider management
010	visual_ai_pipeline	visual_pipelines, pipeline_stages	Visual AI processing
011	brain_router	routing_rules, task_classifications	Request routing
012	metrics_analytics	metrics, analytics_snapshots	Analytics
013	neural_engine	neural_configs, neural_executions	Neural processing
014	error_logging	error_logs, error_patterns	Error tracking
015	credentials_registry	credentials, credential_rotations	Credential management
016	think_tank	thinktank_sessions, thinktank_steps, thinktank_tools	Think Tank
017	concurrent_chat	chat_sessions, chat_messages	Chat management
018	realtime_collaboration	collaborations, collaboration_members	Real-time collab
019	persistent_memory	memories, memory_associations	Memory system
020	focus_personas	personas, persona_configs	AI personas
021	team_plans	teams, team_members, team_plans	Team management
022	provider_registry	providers, provider_health	Provider registry
023	time_machine	snapshots, snapshot_diffs	Time machine
024	orchestration_engine	workflow_definitions, workflow_tasks, workflow_executions, task_executions	Orchestration
025	license_management	licenses, license_activations	Licensing
026	unified_model_registry	unified_models, model_versions	Model registry
027	feedback_learning	feedback, learning_samples	Feedback system
028	neural_orchestration	neural_workflows, neural_steps	Neural orchestration
029	workflow_proposals	proposals, proposal_evidence	Workflow improvements
030	app_isolation	app_contexts, app_permissions	App isolation
031	internationalization	locales, translations	i18n
032	dynamic_configuration	configs, config_history	Dynamic config
033	billing_credits	credit_balances, credit_transactions, credit_purchases	Credits
034	storage_billing	storage_usage, storage_costs	Storage billing
035	versioned_subscriptions	subscription_tiers, subscriptions	Subscriptions
036	dual_admin_approval	dual_approvals, approval_chains	Dual approval
037	canvas_artifacts	canvases, canvas_elements	Canvas
038	scheduled_prompts	scheduled_prompts, prompt_executions	Scheduling
039	auto_resolve	auto_resolutions, resolution_rules	Auto-resolve
040	model_selection_pricing	model_pricing, selection_history	Pricing

041	admin_billing_enhancements	billing_reports, revenue_tracking	Billing enhancements

---

Core Tables (Migration 001)

tenants

Primary multi-tenant organization table.

```
CREATE TABLE tenants (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    name VARCHAR(100) NOT NULL UNIQUE,
    display_name VARCHAR(200) NOT NULL,
    domain VARCHAR(255),
    settings JSONB NOT NULL DEFAULT '{}',
    status VARCHAR(20) NOT NULL DEFAULT 'active'
        CHECK (status IN ('active', 'suspended', 'pending')),
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

-- Indexes
CREATE INDEX idx_tenants_name ON tenants(name);
CREATE INDEX idx_tenants_domain ON tenants(domain) WHERE domain IS NOT NULL;
CREATE INDEX idx_tenants_status ON tenants(status);
```

Settings JSON Structure:

```
{
  "branding": {
    "logo_url": "string",
    "primary_color": "#hex",
    "company_name": "string"
  },
  "limits": {
    "max_users": 100,
    "max_api_keys": 10,
    "monthly_token_limit": 1000000
  },
  "features": {
    "think_tank_enabled": true,
    "orchestration_enabled": true,
    "collaboration_enabled": true
  },
  "compliance": {
    "hipaa_mode": false,
    "data_retention_days": 90
```

```
  }
}
```

---

users

End users within tenants.

```
CREATE TABLE users (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    cognito_user_id VARCHAR(128) NOT NULL,
    email VARCHAR(255) NOT NULL,
    display_name VARCHAR(200),
    role VARCHAR(50) NOT NULL DEFAULT 'user'
        CHECK (role IN ('user', 'power_user', 'admin')),
    status VARCHAR(20) NOT NULL DEFAULT 'active'
        CHECK (status IN ('active', 'suspended', 'pending')),
    settings JSONB NOT NULL DEFAULT '{}',
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    UNIQUE (tenant_id, email),
    UNIQUE (cognito_user_id)
);

-- Indexes
CREATE INDEX idx_users_tenant_id ON users(tenant_id);
CREATE INDEX idx_users_email ON users(email);
CREATE INDEX idx_users_cognito_user_id ON users(cognito_user_id);
CREATE INDEX idx_users_status ON users(status);
```

User Roles: | Role | Permissions | |——|————-| | `user` | Basic API access, own resources | | `power_user` | + Create API keys, advanced features | | `admin` | + Manage users, view analytics |

---

administrators

Platform administrators (separate from tenant users).

```
CREATE TABLE administrators (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    cognito_user_id VARCHAR(128) NOT NULL UNIQUE,
    email VARCHAR(255) NOT NULL UNIQUE,
    display_name VARCHAR(200) NOT NULL,
    role VARCHAR(50) NOT NULL DEFAULT 'admin'
        CHECK (role IN ('super_admin', 'admin', 'operator', 'auditor')),
    permissions TEXT[] NOT NULL DEFAULT '{}',
    mfa_enabled BOOLEAN NOT NULL DEFAULT false,
    last_login_at TIMESTAMPTZ,
```

```
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    invited_by UUID REFERENCES administrators(id)
);
```

Admin Roles: | Role | Description | Permissions | |——|————-|————-| | `super_admin` | Full platform access | All operations | | `admin` | Standard admin | Manage tenants, users, models | | `operator` | Operations | View logs, manage deployments | | `auditor` | Read-only audit | View all, modify none |

---

approval_requests

Two-person approval system.

```
CREATE TABLE approval_requests (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    requester_id UUID NOT NULL REFERENCES administrators(id),
    action_type VARCHAR(100) NOT NULL,
    resource_type VARCHAR(100) NOT NULL,
    resource_id VARCHAR(255),
    payload JSONB NOT NULL DEFAULT '{}',
    status VARCHAR(20) NOT NULL DEFAULT 'pending'
        CHECK (status IN ('pending', 'approved', 'rejected', 'expired')),
    required_approvals INTEGER NOT NULL DEFAULT 1,
    approvals JSONB NOT NULL DEFAULT '[]',
    expires_at TIMESTAMPTZ NOT NULL,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);
```

Action Types Requiring Approval: - `delete_tenant` - Delete a tenant - `modify_billing` - Change billing settings - `grant_super_admin` - Elevate to super admin - `bulk_data_export` - Export all data - `disable_security_feature` - Disable security

---

Think Tank Tables (Migration 016)

thinktank_sessions

Problem-solving session tracking.

```
CREATE TABLE thinktank_sessions (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    problem_summary TEXT,
    domain VARCHAR(50),
    complexity VARCHAR(20) CHECK (complexity IN ('low', 'medium', 'high', 'extreme')),
    total_steps INTEGER DEFAULT 0,
```

```
    avg_confidence DECIMAL(3, 2),
    solution_found BOOLEAN DEFAULT false,
    total_tokens INTEGER DEFAULT 0,
    total_cost DECIMAL(10, 6) DEFAULT 0,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    completed_at TIMESTAMPTZ
);

-- Indexes
CREATE INDEX idx_thinktank_sessions_tenant ON thinktank_sessions(tenant_id, created_at DESC);
CREATE INDEX idx_thinktank_sessions_user ON thinktank_sessions(user_id);

-- RLS
ALTER TABLE thinktank_sessions ENABLE ROW LEVEL SECURITY;
CREATE POLICY thinktank_sessions_isolation ON thinktank_sessions
    FOR ALL USING (tenant_id = current_setting('app.current_tenant_id', true)::uuid);
```

Domains: - `research` - Academic research - `engineering` - Technical problems - `analytical` - Data analysis - `creative` - Creative tasks - `legal` - Legal analysis - `medical` - Medical queries (HIPAA) - `business` - Business strategy - `general` - General problems

---

thinktank_steps

Individual reasoning steps within sessions.

```
CREATE TABLE thinktank_steps (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    session_id UUID NOT NULL REFERENCES thinktank_sessions(id) ON DELETE CASCADE,
    step_number INTEGER NOT NULL,
    step_type VARCHAR(50) NOT NULL
        CHECK (step_type IN ('decompose', 'reason', 'execute', 'verify', 'synthesize')),
    description TEXT,
    reasoning TEXT,
    result TEXT,
    confidence DECIMAL(3, 2) CHECK (confidence >= 0 AND confidence <= 1),
    model_used VARCHAR(100),
    tokens_used INTEGER,
    duration_ms INTEGER,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

-- Indexes
CREATE INDEX idx_thinktank_steps_session ON thinktank_steps(session_id, step_number);

-- RLS
ALTER TABLE thinktank_steps ENABLE ROW LEVEL SECURITY;
CREATE POLICY thinktank_steps_isolation ON thinktank_steps
```

```
    FOR ALL USING (
        session_id IN (SELECT id FROM thinktank_sessions
                       WHERE tenant_id = current_setting('app.current_tenant_id', true)::uuid)
    );
```

Step Types: | Type | Description | Typical Model | |——|————-|————| | `decompose` | Break problem into parts | Claude 3.5 | | `reason` | Chain-of-thought reasoning | o1, Claude | | `execute` | Execute solution step | Task-specific | | `verify` | Verify result accuracy | Different model | | `synthesize` | Combine into final answer | Claude 3.5 |

---

thinktank_tools

Available tools for Think Tank.

```
CREATE TABLE thinktank_tools (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    tool_name VARCHAR(100) NOT NULL UNIQUE,
    tool_type VARCHAR(50) NOT NULL,
    description TEXT,
    parameters_schema JSONB NOT NULL DEFAULT '{}',
    implementation TEXT,
    is_active BOOLEAN DEFAULT true,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

-- Default tools
INSERT INTO thinktank_tools (tool_name, tool_type, description, parameters_schema) VALUES
    ('web_search', 'search', 'Search the web for information', '{"query": "string"}'),
    ('calculator', 'compute', 'Perform mathematical calculations', '{"expression": "string"}')
    ('code_executor', 'compute', 'Execute code snippets', '{"language": "string", "code": "str
    ('file_reader', 'io', 'Read file contents', '{"path": "string"}'),
    ('api_caller', 'network', 'Make API requests', '{"url": "string", "method": "string", "bod
```

Orchestration Tables (Migration 024)

workflow_definitions

Workflow pattern definitions.

```
CREATE TABLE workflow_definitions (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    workflow_id VARCHAR(100) NOT NULL UNIQUE,
    name VARCHAR(200) NOT NULL,
    description TEXT,
    category VARCHAR(50) NOT NULL
        CHECK (category IN ('generation', 'analysis', 'transformation', 'pipeline', 'custom'))
    version VARCHAR(20) NOT NULL DEFAULT '1.0.0',
```

```
    dag_definition JSONB NOT NULL DEFAULT '{}',
    input_schema JSONB NOT NULL DEFAULT '{}',
    output_schema JSONB NOT NULL DEFAULT '{}',
    default_parameters JSONB NOT NULL DEFAULT '{}',

    timeout_seconds INTEGER DEFAULT 3600,
    max_retries INTEGER DEFAULT 3,
    min_tier INTEGER DEFAULT 1,

    is_active BOOLEAN DEFAULT true,
    requires_audit_trail BOOLEAN DEFAULT false,
    hipaa_compliant BOOLEAN DEFAULT false,

    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    created_by UUID
);
```

DAG Definition Structure:

```
{
  "nodes": [
    {
      "taskId": "decompose",
      "taskType": "model_inference",
      "modelId": "anthropic/claude-3-5-sonnet",
      "config": {
        "systemPrompt": "Break down the problem...",
        "temperature": 0.3
      },
      "dependsOn": []
    },
    {
      "taskId": "solve_part_1",
      "taskType": "model_inference",
      "dependsOn": ["decompose"]
    }
  ],
  "edges": [
    {"from": "decompose", "to": "solve_part_1"}
  ]
}
```

---

workflow_tasks

Individual tasks within workflows.

```sql
CREATE TABLE workflow_tasks (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    workflow_id UUID NOT NULL REFERENCES workflow_definitions(id) ON DELETE CASCADE,
    task_id VARCHAR(100) NOT NULL,
    name VARCHAR(200) NOT NULL,
    description TEXT,

    task_type VARCHAR(50) NOT NULL
        CHECK (task_type IN ('model_inference', 'transformation', 'condition',
                             'parallel', 'aggregation', 'external_api', 'human_review')),
    model_id VARCHAR(100),
    service_id VARCHAR(100),

    config JSONB NOT NULL DEFAULT '{}',
    input_mapping JSONB DEFAULT '{}',
    output_mapping JSONB DEFAULT '{}',

    sequence_order INTEGER DEFAULT 0,
    depends_on TEXT[] DEFAULT '{}',
    condition_expression TEXT,
    timeout_seconds INTEGER DEFAULT 300,

    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    UNIQUE(workflow_id, task_id)
);
```

Task Types: | Type | Description | Example | |——|————-|———| | `model_inference` | AI model call | Generate text | | `transformation` | Data transformation | Format output | | `condition` | Conditional branching | If confidence > 0.8 | | `parallel` | Parallel execution | Call 3 models | | `aggregation` | Combine results | Merge responses | | `external_api` | External API call | Web search | | `human_review` | Human-in-the-loop | Approval step |

---

workflow_executions

Workflow execution tracking.

```sql
CREATE TABLE workflow_executions (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    workflow_id UUID NOT NULL REFERENCES workflow_definitions(id),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    user_id UUID NOT NULL,

    status VARCHAR(20) NOT NULL DEFAULT 'pending'
        CHECK (status IN ('pending', 'running', 'paused', 'completed', 'failed', 'cancelled'))

    input_parameters JSONB NOT NULL DEFAULT '{}',
```

```
    resolved_parameters JSONB DEFAULT '{}',
    output_data JSONB,

    error_message TEXT,
    error_details JSONB,

    started_at TIMESTAMPTZ,
    completed_at TIMESTAMPTZ,
    duration_ms INTEGER,

    estimated_cost_usd DECIMAL(10, 4),
    actual_cost_usd DECIMAL(10, 4),

    checkpoint_data JSONB,
    priority INTEGER DEFAULT 5 CHECK (priority >= 1 AND priority <= 10),

    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

-- RLS
ALTER TABLE workflow_executions ENABLE ROW LEVEL SECURITY;
CREATE POLICY workflow_executions_isolation ON workflow_executions
    FOR ALL USING (tenant_id = current_setting('app.current_tenant_id', true)::uuid);
```

---

task_executions

Individual task execution tracking.

```
CREATE TABLE task_executions (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    workflow_execution_id UUID NOT NULL REFERENCES workflow_executions(id) ON DELETE CASCADE,
    task_id VARCHAR(100) NOT NULL,

    status VARCHAR(20) NOT NULL DEFAULT 'pending'
        CHECK (status IN ('pending', 'running', 'completed', 'failed', 'skipped', 'retrying')),
    attempt_number INTEGER DEFAULT 1,

    input_data JSONB,
    output_data JSONB,

    error_message TEXT,
    error_code VARCHAR(50),

    started_at TIMESTAMPTZ,
    completed_at TIMESTAMPTZ,
    duration_ms INTEGER,
```

```
    resource_usage JSONB DEFAULT '{}',
    cost_usd DECIMAL(10, 4),

    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);
```

Billing Tables (Migrations 033-035)

credit_balances

Tenant credit balance tracking.

```
CREATE TABLE credit_balances (
    tenant_id UUID PRIMARY KEY REFERENCES tenants(id) ON DELETE CASCADE,
    balance DECIMAL(12, 4) NOT NULL DEFAULT 0,
    lifetime_purchased DECIMAL(12, 4) NOT NULL DEFAULT 0,
    lifetime_used DECIMAL(12, 4) NOT NULL DEFAULT 0,
    lifetime_bonus DECIMAL(12, 4) NOT NULL DEFAULT 0,
    low_balance_alert_threshold DECIMAL(12, 4),
    last_low_balance_alert TIMESTAMPTZ,
    auto_purchase_enabled BOOLEAN DEFAULT false,
    auto_purchase_threshold DECIMAL(12, 4),
    auto_purchase_amount DECIMAL(12, 4),
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);
```

---

credit_transactions

Credit transaction history.

```
CREATE TABLE credit_transactions (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    transaction_type VARCHAR(30) NOT NULL
        CHECK (transaction_type IN ('purchase', 'bonus', 'refund', 'usage',
                                    'transfer_in', 'transfer_out',
                                    'subscription_allocation', 'expiration', 'adjustment')),
    amount DECIMAL(12, 4) NOT NULL,
    balance_after DECIMAL(12, 4) NOT NULL,
    description TEXT,
    reference_id VARCHAR(255),
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);
```

```
-- Indexes
CREATE INDEX idx_credit_transactions_tenant ON credit_transactions(tenant_id, created_at DESC)
```

---

subscription_tiers

Available subscription plans.

```
CREATE TABLE subscription_tiers (
    id VARCHAR(50) PRIMARY KEY,
    display_name VARCHAR(100) NOT NULL,
    description TEXT,
    price_monthly DECIMAL(10, 2),
    price_annual DECIMAL(10, 2),
    included_credits_per_user DECIMAL(10, 2) NOT NULL DEFAULT 0,
    features JSONB NOT NULL DEFAULT '{}',
    limits JSONB NOT NULL DEFAULT '{}',
    is_public BOOLEAN DEFAULT true,
    sort_order INTEGER DEFAULT 0,
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);

-- Default tiers
INSERT INTO subscription_tiers (id, display_name, price_monthly, price_annual, included_credits
('free', 'Free', 0, NULL, 100, '{"think_tank": false, "orchestration": false, "models": ["gpt-4
('pro', 'Pro', 49, 490, 5000, '{"think_tank": true, "orchestration": true, "models": "all"}'),
('team', 'Team', 199, 1990, 25000, '{"think_tank": true, "orchestration": true, "collaboration"
('enterprise', 'Enterprise', NULL, NULL, 0, '{"think_tank": true, "orchestration": true, "colla
```

---

subscriptions

Active tenant subscriptions.

```
CREATE TABLE subscriptions (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    tenant_id UUID NOT NULL REFERENCES tenants(id) ON DELETE CASCADE,
    tier_id VARCHAR(50) NOT NULL REFERENCES subscription_tiers(id),
    status VARCHAR(20) NOT NULL DEFAULT 'active'
        CHECK (status IN ('active', 'cancelled', 'past_due', 'trialing', 'paused')),
    billing_cycle VARCHAR(10) NOT NULL CHECK (billing_cycle IN ('monthly', 'annual')),
    seats_purchased INTEGER NOT NULL DEFAULT 1,
    seats_used INTEGER NOT NULL DEFAULT 0,
    current_period_start TIMESTAMPTZ NOT NULL,
    current_period_end TIMESTAMPTZ NOT NULL,
    cancel_at_period_end BOOLEAN DEFAULT false,
    cancelled_at TIMESTAMPTZ,
    stripe_customer_id VARCHAR(255),
```

```
    stripe_subscription_id VARCHAR(255),
    created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW()
);
```

Row-Level Security (Migration 002)

RLS Pattern

All tenant-scoped tables use this pattern:

```
-- Enable RLS on table
ALTER TABLE {table_name} ENABLE ROW LEVEL SECURITY;

-- Create isolation policy
CREATE POLICY {table_name}_isolation ON {table_name}
    FOR ALL USING (tenant_id = current_setting('app.current_tenant_id', true)::uuid);
```

Setting Tenant Context

Every request sets the tenant context before queries:

```
SET app.current_tenant_id = '{tenant_uuid}';
```

Tables with RLS Enabled:

- users
- thinktank_sessions
- thinktank_steps
- workflow_executions
- task_executions
- credit_transactions
- subscriptions
- chat_sessions
- chat_messages
- collaborations
- memories
- feedback
- api_keys
- (all tenant-scoped tables)

Common Patterns

Updated At Trigger

```
CREATE OR REPLACE FUNCTION update_updated_at_column()
RETURNS TRIGGER AS $$
BEGIN
```

```
    NEW.updated_at = NOW();
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Apply to tables
CREATE TRIGGER update_{table}_updated_at
    BEFORE UPDATE ON {table}
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();
```

Soft Delete Pattern

```
-- Add columns
deleted_at TIMESTAMPTZ,
deleted_by UUID,

-- Query with filter
WHERE deleted_at IS NULL
```

Audit Columns

```
created_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
updated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
created_by UUID,
updated_by UUID
```