SRC–RWCM Workflow Logic Database v1.0 (Production Blueprint)

Architecture: Symbiotic Recursive Cognition (SRC) • Recursive Workflow Cognition Model (RWCM) • Continual Learning Multi-Agent Ecosystem

Objective: Provide a production-ready, unique, defensible automation schema that: - Represents roles, workflows, steps, and skill nodes as **executable + reflective** objects. - Enables **recursive lookback loops** that (a) optimize execution, (b) synthesize **ad-hoc workflows** and **new skill nodes**, and (c) publish them with governance. - Is ingestible as **SQL (relational)**, **JSON (API/graph)**, and **Sheets/CSV** without transformation.

---

0. Design Tenets

1. **Neural-Symbolic Objects:** Every workflow artifact is both structured (SQL/JSON) and learnable (vectorized memory, metrics, feedback).
2. **Two-Phase Steps:** Each step has **Execution** + **Reflection**; reflection emits learning signals, telemetry, and proposals.
3. **Recursive Learning Mesh (RLM):** Global, append-only memory that powers cross-workflow generalization and synthesis.
4. **Governed Autogenesis:** Agents may propose new steps, workflows, and skill nodes; publication requires policy checks, simulation gates, and provenance.
5. **Deterministic Surfaces:** Deterministic contracts at the boundaries (APIs, events, schemas) allow safe stochastic exploration inside.

---

1. Entity Dictionary (Conceptual)

- **Role**: Human/AI/hybrid responsibility node with authority & capability vectors.
- **SkillNode**: Atomic capability with versioned signature and deterministic contract.
- **Workflow**: Goal-directed program encoded as **Workflow Genome** (objectives + constraints), decomposable into Step-Actions.
- **StepAction**: Executable unit (tool calls, checks, comms) with **reflection hooks**.
- **Trigger**: Event or condition initiating a workflow/step.
- **Policy**: Guardrails for risk, compliance, data, and publication.
- **Agent**: Execution principal (human or autonomous) with role & skill bindings.
- **RLM Memory**: Vector + graph memory of episodes, errors, proposals, and proofs.

---

2. Relational Schema (SQL DDL)

ANSI-SQL; names chosen to avoid reserved keywords; all tables include `created_at`, `updated_at`, `created_by`, `updated_by`.

```
-- ============================
-- A. Core: Roles & Capabilities
-- ============================
CREATE TABLE role (
  role_id           VARCHAR(32) PRIMARY KEY,
  role_title        VARCHAR(128) NOT NULL,
  department        VARCHAR(64)  NOT NULL,
  hierarchy_level   VARCHAR(8)   NOT NULL, -- e.g., I, II, III, IV, V
  supervises_role_id VARCHAR(32),
  role_type         VARCHAR(16)  NOT NULL DEFAULT 'human', -- human|agent|hybrid
  capability_vector VARBINARY(4096), -- optional embedding
  CONSTRAINT fk_role_supervises FOREIGN KEY (supervises_role_id) REFERENCES
role(role_id)
);
CREATE INDEX ix_role_dept ON role(department);

CREATE TABLE skill_node (
  skill_id          VARCHAR(32) PRIMARY KEY,
  skill_name        VARCHAR(128) NOT NULL,
  category          VARCHAR(64)  NOT NULL, -- ingestion|tool|logic|comm|hcm|api|
iam|ml|nlp|viz|etl
  signature         JSON         NOT NULL, -- function contract: name, args,
returns, error_codes
  description       TEXT         NOT NULL,
  runtime_binding   JSON         NOT NULL, -- adapter spec: system, endpoint,
auth, timeouts
  version           VARCHAR(24)  NOT NULL,
  stability_tier    VARCHAR(16)  NOT NULL DEFAULT 'ga', -- exp|beta|ga|
restricted
  owner_role_id     VARCHAR(32)  NOT NULL,
  is_generator      BOOLEAN      NOT NULL DEFAULT FALSE, -- can synthesize
skills/workflows
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  created_by VARCHAR(64) NOT NULL,
  updated_by VARCHAR(64) NOT NULL,
  CONSTRAINT fk_skill_owner_role FOREIGN KEY (owner_role_id) REFERENCES
role(role_id)
);
CREATE UNIQUE INDEX ux_skill_name_version ON skill_node(skill_name, version);
```

```
CREATE TABLE role_skill_map (
  role_id  VARCHAR(32) NOT NULL,
  skill_id VARCHAR(32) NOT NULL,
  permission VARCHAR(16) NOT NULL DEFAULT 'use', -- use|maintain|publish
  PRIMARY KEY(role_id, skill_id),
  CONSTRAINT fk_rsm_role FOREIGN KEY (role_id) REFERENCES role(role_id),
  CONSTRAINT fk_rsm_skill FOREIGN KEY (skill_id) REFERENCES skill_node(skill_id)
);

-- ============================
-- B. Workflows & Steps
-- ============================
CREATE TABLE workflow (
  workflow_id       VARCHAR(32) PRIMARY KEY,
  workflow_name     VARCHAR(128) NOT NULL,
  objective         TEXT         NOT NULL,
  constraints       JSON         NOT NULL, -- SLOs: latency, accuracy, cost;
policy tags
  responsible_role_id VARCHAR(32) NOT NULL,
  genome            JSON         NOT NULL, -- decomposable goals, predicates,
resources
  status            VARCHAR(16)  NOT NULL DEFAULT 'active', -- draft|active|
retired
  version           VARCHAR(24)  NOT NULL,
  lineage           JSON         NOT NULL, -- parent_ids, derived_from,
proposal_reason
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  created_by VARCHAR(64) NOT NULL,
  updated_by VARCHAR(64) NOT NULL,
  CONSTRAINT fk_wf_role FOREIGN KEY (responsible_role_id) REFERENCES
role(role_id)
);
CREATE UNIQUE INDEX ux_workflow_name_version ON workflow(workflow_name,
version);

CREATE TABLE trigger_def (
  trigger_id   VARCHAR(32) PRIMARY KEY,
  workflow_id  VARCHAR(32) NOT NULL,
  trigger_type VARCHAR(24) NOT NULL, -- event|schedule|condition|webhook
  selector     JSON        NOT NULL, -- topic, cron, predicate
  CONSTRAINT fk_trig_wf FOREIGN KEY (workflow_id) REFERENCES
workflow(workflow_id)
);

CREATE TABLE step_action (
  step_id         VARCHAR(32) PRIMARY KEY,
  workflow_id     VARCHAR(32) NOT NULL,
```

```
  sequence        INT NOT NULL,
  action_type     VARCHAR(24) NOT NULL, -- data_extraction|logic_check|
tool_call|branch|handoff|notify|generate
  skill_id        VARCHAR(32) NOT NULL,
  parameters      JSON        NOT NULL,
  next_step_logic JSON        NOT NULL, -- DSL of conditions → step_id(s)
  timeout_ms      INT         NOT NULL DEFAULT 300000,
  retries         INT         NOT NULL DEFAULT 2,
  idempotency_key VARCHAR(64),
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  created_by VARCHAR(64) NOT NULL,
  updated_by VARCHAR(64) NOT NULL,
  CONSTRAINT fk_step_wf FOREIGN KEY (workflow_id) REFERENCES
workflow(workflow_id),
  CONSTRAINT fk_step_skill FOREIGN KEY (skill_id) REFERENCES
skill_node(skill_id)
);
CREATE INDEX ix_step_wf_sequence ON step_action(workflow_id, sequence);

-- ============================
-- C. Execution, Reflection, Learning
-- ============================
CREATE TABLE agent (
  agent_id       VARCHAR(32) PRIMARY KEY,
  agent_name     VARCHAR(128) NOT NULL,
  agent_type     VARCHAR(24)  NOT NULL, -- l2_orchestrator|skill_agent|
human_proxy|evaluator
  role_id        VARCHAR(32)  NOT NULL,
  policy_profile JSON         NOT NULL,
  embedding      VARBINARY(4096),
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  created_by VARCHAR(64) NOT NULL,
  updated_by VARCHAR(64) NOT NULL,
  CONSTRAINT fk_agent_role FOREIGN KEY (role_id) REFERENCES role(role_id)
);

CREATE TABLE run_episode (
  episode_id   VARCHAR(36) PRIMARY KEY,
  workflow_id  VARCHAR(32) NOT NULL,
  trigger_id   VARCHAR(32),
  initiator_id VARCHAR(32), -- agent or user
  started_at   TIMESTAMP NOT NULL,
  ended_at     TIMESTAMP,
  outcome      VARCHAR(24) NOT NULL DEFAULT 'running', -- success|failed|
partial|running
```

```
  metrics      JSON        NOT NULL, -- cost_ms, latency_ms, tokens, error_rates
  context_hash VARCHAR(64) NOT NULL,
  CONSTRAINT fk_episode_wf FOREIGN KEY (workflow_id) REFERENCES
workflow(workflow_id)
);
CREATE INDEX ix_episode_wf_time ON run_episode(workflow_id, started_at DESC);

CREATE TABLE step_run (
  step_run_id  VARCHAR(36) PRIMARY KEY,
  episode_id   VARCHAR(36) NOT NULL,
  step_id      VARCHAR(32) NOT NULL,
  agent_id     VARCHAR(32) NOT NULL,
  started_at   TIMESTAMP NOT NULL,
  ended_at     TIMESTAMP,
  status       VARCHAR(24) NOT NULL, -- success|failed|skipped|fallback
  input        JSON        NOT NULL,
  output       JSON,
  error        JSON,
  telemetry    JSON        NOT NULL, -- tokens, latency, cost, model, system
  CONSTRAINT fk_sr_episode FOREIGN KEY (episode_id) REFERENCES
run_episode(episode_id),
  CONSTRAINT fk_sr_step FOREIGN KEY (step_id) REFERENCES step_action(step_id),
  CONSTRAINT fk_sr_agent FOREIGN KEY (agent_id) REFERENCES agent(agent_id)
);

CREATE TABLE reflection_log (
  reflection_id  VARCHAR(36) PRIMARY KEY,
  source_type    VARCHAR(24) NOT NULL, -- step_run|episode|agent
  source_id      VARCHAR(36) NOT NULL,
  insight_type   VARCHAR(32) NOT NULL, -- variance|pattern|hallucination|drift|
opportunity
  insight        JSON        NOT NULL, -- normalized reasoning, embeddings,
spans
  learning_signal JSON       NOT NULL, -- reward, penalty, confidence deltas
  proposed_actions JSON      NOT NULL, -- retune, update_param, new_skill,
new_workflow
  reviewer_role_id VARCHAR(32),        -- governance reviewer
  created_at     TIMESTAMP   NOT NULL DEFAULT CURRENT_TIMESTAMP,
  approved_at    TIMESTAMP,
  status         VARCHAR(16) NOT NULL DEFAULT 'recorded' -- recorded|approved|
rejected|published
);
CREATE INDEX ix_reflection_status ON reflection_log(status, created_at DESC);

CREATE TABLE publication_queue (
  publication_id VARCHAR(36) PRIMARY KEY,
  proposal_type  VARCHAR(24) NOT NULL, -- workflow|step|skill
  proposal_body  JSON        NOT NULL,
```

```
  provenance     JSON        NOT NULL, -- episodes, step_runs, proofs, tests
  risk_score     DECIMAL(5,2) NOT NULL,
  policy_checks  JSON        NOT NULL,
  decision       VARCHAR(16) NOT NULL DEFAULT 'pending', -- pending|approved|
rejected
  decided_by     VARCHAR(64),
  decided_at     TIMESTAMP,
  published_object_id VARCHAR(32)
);

-- ============================
-- D. Policy, Ontology, Events
-- ============================
CREATE TABLE policy (
  policy_id    VARCHAR(32) PRIMARY KEY,
  policy_name  VARCHAR(128) NOT NULL,
  scope        VARCHAR(24)  NOT NULL, -- data|security|compliance|mlops|
publishing
  spec         JSON         NOT NULL,
  owner_role_id VARCHAR(32) NOT NULL,
  CONSTRAINT fk_policy_owner_role FOREIGN KEY (owner_role_id) REFERENCES
role(role_id)
);

CREATE TABLE ontology_node (
  node_id      VARCHAR(32) PRIMARY KEY,
  node_type    VARCHAR(24)  NOT NULL, -- entity|event|metric|policy_tag
  label        VARCHAR(128) NOT NULL,
  attributes   JSON         NOT NULL,
  parent_id    VARCHAR(32)
);

CREATE TABLE event_bus (
  event_id     VARCHAR(36) PRIMARY KEY,
  topic        VARCHAR(128) NOT NULL, -- e.g., ap.invoice.received
  payload      JSON         NOT NULL,
  produced_at  TIMESTAMP    NOT NULL,
  producer_id  VARCHAR(64)  NOT NULL
);
CREATE INDEX ix_event_topic_time ON event_bus(topic, produced_at DESC);
```

---

3. JSON API Schemas (Canonical)

3.1 SkillNode (POST /skills)

```
{
  "skill_id": "SK-OCR-001",
  "skill_name": "Document_Entity_Extractor",
  "category": "ingestion",
  "signature": {
    "name": "extract",
    "args": [
      {"name": "file_uri", "type": "string", "required": true},
      {"name": "entities", "type": "array<string>", "required": true},
      {"name": "min_confidence", "type": "number", "default": 0.85}
    ],
    "returns": {"type": "object", "properties": {"entities": "map<string,any>",
"confidence": "number"}},
    "errors": ["CONNECTIVITY", "SCHEMA_MISMATCH", "LOW_CONFIDENCE"]
  },
  "runtime_binding": {
    "adapter": "aws_textract_v2",
    "endpoint": "arn:aws:textract:...",
    "auth": "role/iam/textract-exec",
    "timeouts_ms": 30000
  },
  "version": "2.1.0",
  "stability_tier": "ga",
  "owner_role_id": "IT-M-001",
  "is_generator": false
}
```

3.2 Workflow (POST /workflows)

```
{
  "workflow_id": "WF-FIN-001",
  "workflow_name": "Automated Invoice Processing",
  "objective": "Record invoices to GL with compliant PO matching and
notifications",
  "constraints": {"latency_ms": 600000, "max_error_rate": 0.005, "policy_tags":
["sox", "pii_redaction"]},
  "responsible_role_id": "FA-IC-001",
  "genome": {
    "goals": ["capture_invoice", "validate_po", "book_entry", "notify_vendor"],
    "resources": ["sap", "ap_inbox", "vendor_master"],
    "predicates": ["amount_within_tolerance", "vendor_active"]
```

```
  },
  "version": "1.3.0",
  "lineage": {"derived_from": [], "proposal_reason": "initial_enterprise_pack"}
}
```

3.3 StepAction (POST /workflows/{id}/steps)

```
{
  "step_id": "F-02",
  "workflow_id": "WF-FIN-001",
  "sequence": 2,
  "action_type": "logic_check",
  "skill_id": "SK-LOGIC-003",
  "parameters": {"check": "po_match", "tolerance": 0.05},
  "next_step_logic": {
    "if": [{"expr": "pass==true", "goto": "F-04"}, {"expr": "pass==false",
"goto": "F-03"}],
    "on_error": "F-03"
  },
  "timeout_ms": 20000,
  "retries": 1,
  "idempotency_key": "wf_fin_001_f02_v1"
}
```

3.4 Reflection (POST /reflection)

```
{
  "source_type": "step_run",
  "source_id": "8b8f...",
  "insight_type": "variance",
  "insight": {"pattern": "frequent_po_over_by_3pct", "vendors":
["VEND-445","VEND-992"]},
  "learning_signal": {"reward": -0.2, "confidence": 0.91},
  "proposed_actions": [{"type": "update_param", "target":
"F-02.parameters.tolerance", "value": 0.04}, {"type": "new_workflow",
"template": "WF-FIN-009-VendorVarianceMitigation"}]
}
```

---

4. Skills Node Matrix (Expanded)

Skill_ID	Name	Category	Signature (Summary)	Description	is_genera
SK-OCR-001	Document_Entity_Extractor	ingestion	`extract(file_uri, entities[], min_conf)`	OCR/NLP for unstructured docs	false
SK-ERP-002	System_Record_Writeback	tool	`write(system, action, data_payload)`	Transaction write to ERP/GL/CRM	false
SK-LOGIC-003	Compliance_Variance_Check	logic	`check(lhs, rhs, policy, tolerance)`	Policy/rule evaluation	false
SK-COMM-004	Human_Handoff_Protocol	comm	`handoff(reason_code, owner_id, context)`	Escalate with state carryover	false
SK-COMM-005	Standard_Notification	comm	`send(channel, recipient, template_id, data)`	Email/Slack/SMS	false
SK-HCM-006	HCM_System_Interface	tool	`query(system, object_id)`	Workday/SuccessFactors adapter	false
SK-API-007	Inter_Agent_API_Caller	api	`call(target_workflow_id, payload)`	Cross-workflow trigger	false
SK-IT-008	Identity_Access_Manager	tool	`provision(system, user_id, access_group)`	AD/Okta/Exchange ops	false
SK-GEN-009	Workflow_Synthesizer	generator	`synthesize(genome, episodes[], constraints)`	Proposes new workflows	true
SK-GEN-010	Skill_Induction	generator	`induce(capability_gap, traces[], io_pairs[])`	Proposes new skill nodes	true
SK-ETL-011	Tabular_ETL	etl	`transform(sql_or_dag, inputs[], outputs[])`	Data pipelines	false
SK-ML-012	Model_Retrainer	ml	`retrain(dataset_id, objective, hyperparams)`	Continual model updates	false
SK-NLP-013	Redaction_Filter	logic	`redact(text, policies[])`	PII/SOX redaction	false
SK-VIZ-014	Metrics_Dashboard	viz	`render(view_id, params)`	Observability/BI	false

Note: `SK-GEN-009` and `SK-GEN-010` are the **autogenesis primitives** that enable ad-hoc workflow/skill creation.

5. Governance & Publication (Autogenesis)

5.1 Publication States

- **recorded → approved → published** (or rejected). Managed via `reflection_log` + `publication_queue`.

5.2 Policy Profiles (examples)

```
{
  "policy_id": "POL-PUBLISH-001",
  "scope": "publishing",
  "spec": {
    "require_tests": true,
    "min_confidence": 0.85,
    "risk_threshold": 0.40,
    "mandatory_reviewers": ["QA-M-001","SEC-M-001"],
    "blocked_categories": ["payments.write"],
    "audit_trail": true
  }
}
```

5.3 Deterministic Gates

- **Contract tests:** signature conformance, schema diffs, backward compatibility.
- **Replay tests:** determinism on historical traces.
- **Sandboxes:** non-prod execution with synthetic PII.
- **Risk scoring:** via `risk_score` on `publication_queue` computed from blast radius, data sensitivity, privilege level, and historical error.

---

6. Event Model (Triggers & Topics)

Topic	Payload (canonical)	Producer	Consumer
`ap.invoice.received`	`{invoice_uri, vendor_id, po_id, received_at}`	Mail Ingestor	WF-FIN-001 trigger
`hr.offer.signed`	`{candidate_id, role_id, start_date}`	HCM	WF-HR-002 trigger
`it.provision.complete`	`{user_id, accounts[], dt}`	IAM	HR Onboarding step H-05
`reflection.proposed`	`{source_id, proposal_type, body}`	Orchestrator	Publication queue

Topic	Payload (canonical)	Producer	Consumer
`policy.updated`	`{policy_id, spec}`	Sec/ Compliance	Orchestrator cache

---

7. Ad-Hoc Synthesis (Algorithms)

7.1 Workflow Synthesis (SK-GEN-009)

Inputs: `genome`, `episodes[]`, `constraints`, `ontology`

Procedure: 1. mine episodes for frequent subgraphs (successful step sequences under SLOs) 2. detect bottlenecks/exception motifs 3. align with ontology goals & policies 4. propose `workflow` with steps, parameters, and triggers 5. emit `publication_queue` item with proofs (coverage %, deltas vs baseline)

Output: deterministic `workflow` JSON + unit tests + migration script.

7.2 Skill Induction (SK-GEN-010)

Inputs: `capability_gap`, `traces[]`, `io_pairs[]`

Procedure: 1. cluster failure modes and unknown tool invocations 2. infer minimal signature to close gaps 3. generate adapter template (runtime_binding) and contract tests 4. propose `skill_node` with version `0.1.0` and `stability_tier='exp'`

Output: new `skill_node` plus `role_skill_map` suggestions and sandbox runs.

---

8. Coverage: Fortune-500 Workflows (Curated Set)

8.1 Finance

- **WF-FIN-001** Automated Invoice Processing (PO match, book to GL, vendor notify)
- **WF-FIN-002** Cash Application (remittance parse, apply to AR, exception queue)
- **WF-FIN-003** Expense Audit (policy check, receipts OCR, GL post)
- **WF-FIN-004** Close & Consolidation (trial balance checks, variance, report pack)

8.2 HR

- **WF-HR-002** New Hire Onboarding (welcome, provisioning, compliance)
- **WF-HR-003** Performance Review Cycle (notify, collect, calibrate, finalize)
- **WF-HR-004** Offboarding (access revoke, equipment return, exit data)

8.3 IT

- **WF-IT-003** Access Provisioning (AD, email, SSO)
- **WF-IT-004** Incident Response Triage (classify, route, remediate, RCA)
- **WF-IT-005** Patch Management (KB ingest, maintenance window, rollout)

8.4 Procurement / Supply Chain

- **WF-PROC-004** Requisition → PO (budget check, approvals, PO issue)
- **WF-PROC-005** Vendor Onboarding (KYV, tax forms, banking validation)
- **WF-SC-006** Demand Planning (forecast, plan, commit, monitor)

8.5 Sales & Marketing

- **WF-SAL-005** Sales Order Processing (intake, ATP, record, fulfill)
- **WF-SAL-006** Quote-to-Cash (CPQ config, approval, contract, invoice)
- **WF-MKT-006** Campaign Launch (brief, assets, run, report)

8.6 Customer Service

- **WF-CS-007** Ticket Resolution (triage, KB, escalate, close)
- **WF-CS-008** CSAT Loop (survey, analyze, remediate trend)

8.7 Legal & Compliance

- **WF-LGL-001** Contract Review (extraction, clause check, redlines, sign)
- **WF-COM-001** Policy Update Rollout (draft, impact assess, notify, attest)

8.8 Data & Analytics

- **WF-DA-001** Data Ingest & QA (schema check, PII redact, lineage write)
- **WF-DA-002** Model Retraining (drift detect, sample, retrain, validate, deploy)

Each workflow is stored with genome + steps; see §2 and §3 for exact schemas.

---

9. Detailed Step-Action Example (Finance: WF-FIN-001)

Step	Seq	Action	Skill	Parameters	Next Logic
F-01	1	data_extraction	SK-OCR-001	`{file_uri, entities: [vendor_id, amount, po_id]}`	`→ F-02`
F-02	2	logic_check	SK-LOGIC-003	`{check: po_match, tolerance: 0.05}`	`pass→F-04; fail→F-03`
F-03	3	handoff	SK-COMM-004	`{reason: PO_MISMATCH, owner: FA-M-001}`	`await decision`

Step	Seq	Action	Skill	Parameters	Next Logic
F-04	4	tool_call	SK-ERP-002	`{system: SAP, action: Book_Invoice}`	`→ F-05`
F-05	5	notify	SK-COMM-005	`{channel: email, recipient: vendor}`	`END`

Reflection hooks (auto-attached): - delta on tolerance effectiveness; vendor-specific outlier model; cost/latency. - propose `WF-FIN-009 Vendor Variance Mitigation` when variance pattern sustained.

---

10. Continual Learning (RLM) Flow

1. **Emit:** Every `step_run` writes telemetry + embeddings.
2. **Reflect:** Orchestrator summarizes runs → `reflection_log` with insight types.
3. **Propose:** `SK-GEN-009/010` formulate structured proposals.
4. **Gate:** `publication_queue` runs policy checks + tests.
5. **Publish:** On approval, new/updated objects inserted with bumped versions; lineage written; dashboards updated.

---

11. Access, Audit, and Observability

- **RBAC:** via `role`, `role_skill_map`, and `policy_profile` on `agent`.
- **Provenance:** `lineage` on workflows; `provenance` in `publication_queue`.
- **Audit:** append-only `event_bus` + hashed `context_hash` in `run_episode`.
- **Metrics:** latency, success rate, error taxonomy, dollarized savings; rendered by `SK-VIZ-014`.

---

12. Deterministic Interfaces (Agent Runtime)

12.1 Orchestrator Contract

```
{
  "execute_workflow": {"workflow_id": "string", "trigger_payload": "object"},
  "execute_step": {"step_id": "string", "input": "object"},
  "record_reflection": {"source_id": "string", "insight": "object"},
  "propose_publication": {"proposal_type": "enum", "body": "object"}
}
```

12.2 Step DSL (next_step_logic)

```
{
  "if": [
    {"expr": "output.pass == true", "goto": "F-04"},
    {"expr": "output.pass == false", "goto": "F-03"}
  ],
  "on_error": "F-03"
}
```

---

13. Seed Data (Roles)

Role_ID	Title	Dept	Level	Supervises
FA-IC-001	Accounts Payable Specialist	Finance	V	FA-M-001
FA-M-001	Finance Manager	Finance	III	FA-D-001
HR-IC-001	HR Coordinator	HR	V	HR-M-001
HR-M-001	Talent Acquisition Manager	HR	III	HR-D-001
IT-IC-001	IT Support Technician	IT	V	IT-M-001
IT-M-001	IT Operations Manager	IT	III	IT-D-001
SAL-IC-001	Sales Representative	Sales	V	SAL-M-001
SAL-M-001	Sales Manager	Sales	III	SAL-D-001
MKT-M-001	Marketing Manager	Marketing	III	MKT-D-001
LGL-M-001	Corporate Counsel	Legal	III	LGL-D-001
SEC-M-001	Security & Compliance Manager	Compliance	III	SEC-D-001

Extend as needed; schema supports unlimited roles and cross-department graphs.

---

14. Deployment Notes

- **Migrations:** Apply SQL DDL; register API schemas; seed mandatory policies.
- **Connectors:** Bind `runtime_binding` for ERP/HCM/IAM; store secrets in vault.
- **Backfills:** Import historical episodes to prime RLM for synthesis quality.
- **Dashboards:** Provision views over `run_episode`, `step_run`, `reflection_log`.

---

15. Acceptance Tests (Extract)

- **Schema Round-Trip:** JSON → SQL persist → JSON export is lossless.
- **Determinism:** Replays of the same `context_hash` produce identical `next_step_logic` paths.
- **Autogenesis Safety:** No proposal with `risk_score > threshold` can be published; blocked categories enforced.
- **PII Guarding:** `SK-NLP-013` redaction invoked on any payload with `policy_tags` containing `pii_*`.

---

16. What Makes This Unique

- **Genome-based workflows** + **reflection-first steps** with **learned synthesis**.
- **Autonomous creation** of workflows and skills under **provable gates**.
- **Neural-symbolic continuity:** every artifact is both a contract and a learning unit.

This blueprint is implementation-ready across SQL, JSON, and Sheets; it includes all contracts, safety rails, and learning primitives to support recursive lookback loops that spawn ad-hoc workflows and skill nodes while maintaining enterprise-grade governance.

---

17. Governance Enhancements (Reviewer Workflow, Precedence, Conflict Resolution)

17.1 Audit & Timestamps — DDL Patches

Apply these once to bring every table to governance parity.

```
-- Role
ALTER TABLE role
  ADD COLUMN created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN created_by VARCHAR(64) NOT NULL DEFAULT 'system',
  ADD COLUMN updated_by VARCHAR(64) NOT NULL DEFAULT 'system';

-- Role-Skill Map
ALTER TABLE role_skill_map
  ADD COLUMN created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN created_by VARCHAR(64) NOT NULL DEFAULT 'system',
  ADD COLUMN updated_by VARCHAR(64) NOT NULL DEFAULT 'system';

-- Trigger
ALTER TABLE trigger_def
  ADD COLUMN created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
```

```
  ADD COLUMN updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN created_by VARCHAR(64) NOT NULL DEFAULT 'system',
  ADD COLUMN updated_by VARCHAR(64) NOT NULL DEFAULT 'system';

-- Policy
ALTER TABLE policy
  ADD COLUMN created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN created_by VARCHAR(64) NOT NULL DEFAULT 'system',
  ADD COLUMN updated_by VARCHAR(64) NOT NULL DEFAULT 'system';

-- Ontology
ALTER TABLE ontology_node
  ADD COLUMN created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN created_by VARCHAR(64) NOT NULL DEFAULT 'system',
  ADD COLUMN updated_by VARCHAR(64) NOT NULL DEFAULT 'system';

-- Event Bus
ALTER TABLE event_bus
  ADD COLUMN created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  ADD COLUMN created_by VARCHAR(64) NOT NULL DEFAULT 'system';
```

17.2 Reviewer Assignment Workflow (WF-GOV-001)

Trigger: `reflection.proposed` → **Goal:** Assign qualified reviewers, time-box decision, enforce SLAs.

Step	Seq	Action	Skill	Parameters	Next
G-01	1	logic_check	SK-LOGIC-003	`{policy: reviewer_rules, inputs: proposal_type, risk_score}`	→ G-02
G-02	2	generate	SK-API-007	`{target_workflow_id: "WF-GOV-002-SelectReviewers", payload:{domain, risk, skills}}`	→ G-03
G-03	3	notify	SK-COMM-005	`{recipients: [reviewer_ids], template: assign}`	→ G-04
G-04	4	branch	SK-LOGIC-003	`{check: sla_ack, window_h: 8}`	ack→G-05; timeout→G-06
G-05	5	record	SK-ERP-002	`{system: governance_db, action: record_assignment}`	END

Step	Seq	Action	Skill	Parameters	Next
G-06	6	handoff	SK-COMM-004	`{reason: reviewer_timeout, owner: QA-M-001}`	END

Reviewer selection rule: map `proposal_type` and `ontology tags` to **mandatory** domains (e.g., Security, Data, Business) with at least one independent reviewer per domain; add proposer's org-external reviewer for high-risk proposals.

17.3 Policy Precedence & Conflict Resolution

- **Precedence Order:** `statutory > contractual > corporate > domain > workflow-local`.
- **Conflict Resolution Algorithm:** 1) Normalize policies into canonical predicates; 2) Detect overlaps using ontology tags; 3) If contradictions exist, choose the **highest precedence** predicate; 4) If same level, select **most restrictive**; 5) Emit decision proof into `publication_queue.policy_checks` with conflicting policy IDs and rationale; 6) Notify owners.
- **Change Windows:** `policy.updated` events force cache refresh; workflows with incompatible policy diffs are auto-paused pending review.

---

18. Performance Plan (Scale Patterns)

18.1 Partitioning & Storage

- **Time–Range Partitioning:**
- `run_episode`, `step_run`, `reflection_log`, `event_bus` partitioned **monthly** by `started_at/created_at`.
- **Sub-partition by Workflow:** optional hash on `workflow_id` for `step_run`.
- **Hot/Cold Split:** retain **hot 90 days** in OLTP; archive older partitions to object storage (data lake) with external tables.

18.2 Indexing Guidelines

- Covering indexes:
- `ix_episode_wf_time(workflow_id, started_at DESC)` (exists)
- Add `ix_step_run_episode(step_id, status, ended_at DESC)`
- `ix_reflection_status(status, created_at DESC)` (exists)
- `ix_event_topic_time(topic, produced_at DESC)` (exists)
- JSON access: computed columns for frequent paths (e.g., `(parameters->>'tolerance')::numeric`).

18.3 Retention & Purge

- **Policy-driven:** by `policy.scope=data` → `{pii: 180 days, telemetry: 365 days, events: 400 days}`.
- **Legal hold:** tag partitions; disable purge via policy override.

18.4 Telemetry Schema

- **Structured Columns:** `latency_ms INT`, `cost_cents INT`, `token_in INT`, `token_out INT`, `model VARCHAR(64)`, `error_code VARCHAR(64)` in `step_run.telemetry` **plus** materialized view `mv_step_telemetry` with extracted columns.
- **External Lake:** write full verbose telemetry JSON to `s3://…/telemetry/yyyy=MM/dd=DD/` with schema registered (see §22). Use table federation for ad-hoc analytics.

---

19. Secrets & Connectors (runtime_binding)

19.1 Vault Integration Pattern

- `skill_node.runtime_binding` must reference **indirect secrets**:

```
{
  "adapter":"sap_rest",
  "endpoint":"https://sap.company.tld/api",
  "auth": {"secret_ref": "vault:kv/prod/integrations/sap#token"},
  "network": {"egress": "privatelink:sap-prod"},
  "timeouts_ms": 30000,
  "retries": 3
}
```

- Secrets never stored in DB; rotated via vault; agents fetch ephemeral tokens via workload identity (OIDC/STS).

19.2 Connectivity Patterns

- **Outbound-only** from orchestrator → vendors via NAT/egress proxy.
- **PrivateLink/VPC-Peering** for internal SaaS; deny public IPs.
- **mTLS** with SPIFFE/SPIRE for service identity.

---

20. Autogenesis Ops (Concurrency & Cadence)

20.1 Deduplication & Merge

- Compute `proposal_hash = SHA256(normalize(proposal_body))`; reject duplicates.
- **Similarity Merge:** dense embedding of proposal body; if cosine ≥ 0.92, merge into a **meta-proposal** with unioned proofs; maintain `merged_from[]` in `publication_queue.provenance`.

20.2 Scheduled Reviews

- **Queues by Risk:** `low: weekly`, `med: twice weekly`, `high: daily` review runs.

- **Auto-expire:** proposals with no action in 21 days → auto-close with summary reflection.
- **Reviewer Load Shedding:** if reviewer SLA risk > threshold, auto-reassign using WF-GOV-001.

---

21. Testing Harness (Simulation → CI/CD → Canary)

21.1 Simulation Environment

- **Ephemeral Namespaces:** spin up isolated env with stubbed connectors; seed with sanitized fixtures.
- **Event Replay:** deterministic replays from `event_bus` for regression.

21.2 Contract Test Library

- **Skill Contracts:** signature, error taxonomy, idempotency.
- **Workflow Contracts:** step order, branch reachability, timeouts, compensation.
- **Policy Tests:** publishing gate checks, data residency, PII redaction.

21.3 CI/CD Steps

1. Lint schemas/DSL → 2. Compile contracts → 3. Unit (skills) → 4. Workflow sims → 5. Policy audit → 6. Risk score → 7. Human review (WF-GOV-001) → 8. Canary publish → 9. Promote to GA.

---

22. RLM Implementation (Vector + Graph)

22.1 Storage Choices

- **Vector DB:** pgvector or Pinecone for `episode`, `step`, `insight` embeddings.
- **Graph DB:** Neo4j or Neptune for role/workflow/skill/ontology edges.

22.2 Embedding Schema (SQL excerpt)

```
CREATE TABLE rlm_embedding (
  emb_id VARCHAR(36) PRIMARY KEY,
  source_type VARCHAR(24) NOT NULL, -- step|episode|insight|policy|ontology
  source_id VARCHAR(36) NOT NULL,
  vector VECTOR(1024) NOT NULL, -- pgvector
  metadata JSON NOT NULL,
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
);
CREATE INDEX ix_rlm_stype ON rlm_embedding(source_type);
```

22.3 Retrieval Algorithms

- **Hybrid:** BM25 over normalized text + ANN over vectors.

- **Cross-workflow Mining:** mine frequent subgraphs via gSpan-like algorithm on graph DB; feed motifs to `SK-GEN-009`.
- **Rerank:** MMR or learning-to-rank with business SLO features.

---

23. Event Schema Governance (Registry & Typed Payloads)

23.1 Schema Registry

```
CREATE TABLE schema_registry (
  subject VARCHAR(128) PRIMARY KEY, -- e.g., ap.invoice.received
  version INT NOT NULL,
  format VARCHAR(16) NOT NULL CHECK (format IN ('json','avro')),
  schema JSON NOT NULL,
  compatibility VARCHAR(16) NOT NULL DEFAULT 'backward', -- none|backward|
forward|full
  created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
);
```

23.2 Typed Payloads & Enforcement

- `event_bus.payload` validated at ingest against `schema_registry(subject)` latest **compatible** version.
- **Versioning:** bump minor for backward-compatible changes; major for breaking changes; orchestrator enforces topic-version allowlist per consumer.

23.3 Consumer Sync

- Auto-generate client types from registry (OpenAPI/Avro); publish in package repo; CI blocks deployments with incompatible consumers.

---

24. Operational Runbooks (Highlights)

- **Backfill:** load historical `event_bus` → compute embeddings → prime RLM.
- **Rotation:** secrets rotated quarterly or on breach; revoke tokens on role changes.
- **Disaster Recovery:** PITR for OLTP; lake is immutable with lifecycle rules.
- **KPIs:** proposal lead-time, publish rate, rollback rate, SLO adherence, $-savings.

This enhancement adds concrete governance, scale, security, autogenesis operations, testing, learning infrastructure, and schema-registry guarantees—fully production-ready and aligned with your SRC–RWCM architecture.