Contents

Cortex Memory System - Administrator Guide

Version: 4.20.0
Last Updated: January 2026
Component: RADIANT Platform Core

Table of Contents

---

1. Executive Summary

1.1 Why Tiered Memory?

Direct database storage fails at enterprise scale due to:

Problem	Impact
Volume Limits	PostgreSQL degrades past 100M rows per table
Latency Degradation	Cold queries block hot context retrieval
Cost Inefficiency	Paying hot-storage prices for cold data
Compliance Conflicts	GDPR/HIPAA require different retention per data type
Data Gravity	Customers can't bring their own data lakes

1.2 The Solution: Hot/Warm/Cold Architecture

```
                    CORTEX MEMORY SYSTEM

   HOT TIER          WARM TIER              COLD TIER
 Redis + DynamoDB  Neptune + pgvector      S3 + Iceberg

 Real-time context  Knowledge Graph   Historical Archive
 Session state     90-day window     Zero-Copy Mounts
 Ghost Vectors     Graph-RAG         Glacier tiering

   < 10ms            < 100ms               < 2 seconds
  (p99 target)      (p99 target)          (p99 target)
```

1.3 Key Benefits

- **Performance**: Sub-10ms hot reads, sub-100ms graph queries
- **Cost Efficiency**: 90% reduction in storage costs for archival data
- **Compliance**: Automated retention policies per data classification
- **Data Sovereignty**: Zero-Copy mounts to tenant-owned data lakes
- **Scalability**: Each tier scales independently

---

2. Architecture Overview

2.1 The "Retrieval Dance" - Runtime Query Flow

The Cortex uses a sophisticated multi-tier retrieval pattern:

```
                    THE "RETRIEVAL DANCE"


  Step 1: INTENT PARSING (Hot Tier)

  • Analyze User Query + Ghost Vectors
  • Identify entities mentioned ("Pump 302", "Q4 Report")
```

- Determine query intent and required depth

Step 2: GRAPH TRAVERSAL (Warm Tier)

- Traverse Knowledge Graph 2-3 hops from identified entities
- CRITICAL: Check for "Golden Rule" Overrides
  (e.g., "Always ignore Manual v1 for Pump 302")
- If override exists → Apply strict priority

Step 3: DEEP FETCH (Cold Tier)

- If Graph points to archived content (e.g., page 47 of 500MB PDF)
- Generate signed URL to fetch ONLY that specific content
- Retrieve via Stub Nodes (metadata pointers to external storage)

Step 4: SYNTHESIS (Foundation Model)

- Package: Query + Graph Logic + Fetched Content
- Route to appropriate model (Claude, Gemini, etc.)
- Generate response with Chain of Custody audit trail

Latency Breakdown: | Step | Target | Actual p99 | |——|——|————| | Intent Parsing | < 10ms | 3ms | | Graph Traversal | < 100ms | 75ms | | Deep Fetch (if needed) | < 2s | 1.2s | | Synthesis | < 500ms | 350ms |

2.2 Tier Responsibilities

Tier	Data Types	Retention	Technology
Hot	Session context, Ghost Vectors, telemetry feeds	4 hours default	Redis Cluster + DynamoDB overflow
Warm	Knowledge graph, entity relationships, document embeddings	90 days default	Amazon Neptune + Aurora pgvector
Cold	Historical archives, audit logs, compliance records	7+ years	S3 Iceberg + Glacier lifecycle

2.3 Tier Coordinator

The TierCoordinator service orchestrates:

- **Automatic Promotion**: Hot → Warm when TTL expires
- **Automatic Archival**: Warm → Cold after retention period
- **On-Demand Retrieval**: Cold → Warm when historical data needed
- **Eviction Policies**: LRU for Hot tier, confidence-based for Warm

---

3. Hot Tier Administration

3.1 Redis Cluster Configuration

Default production settings:

Setting	Default	Description
`hot_redis_cluster_mode`	`true`	Enable sharding
`hot_shard_count`	`3`	Number of shards
`hot_replicas_per_shard`	`2`	Replicas for HA
`hot_instance_type`	`r7g.xlarge`	AWS instance type
`hot_max_memory_percent`	`80`	Eviction threshold
`hot_default_ttl_seconds`	`14400`	4 hours default TTL
`hot_overflow_to_dynamodb`	`true`	Overflow large values

3.2 Key Schema (Tenant Isolation)

All Redis keys follow this pattern:

`{tenant_id}:{data_type}:{identifier}`

Examples:

```
abc123:session:user_456:context
abc123:ghost:user_789
abc123:telemetry:stream_001
abc123:prefetch:doc_123
```

3.3 Data Types in Hot Tier

Data Type	Key Pattern	TTL	Description
Session Context	`{tenant}:session:{user}:context`	4h	Current conversation + tool calls
Ghost Vectors	`{tenant}:ghost:{user}`	24h	User personality embeddings (Role, Bias, Preferences)
Live Telemetry	`{tenant}:telemetry:{stream}`	1h	Real-time sensor feeds (MQTT/OPC UA)
Prefetch Cache	`{tenant}:prefetch:{doc}`	30m	Anticipated document needs

3.4 Live Telemetry Integration (Industrial IoT)

The Hot tier can ingest real-time sensor data directly into the context window:

Protocol	Use Case	Injection Method
MQTT	IoT sensors, edge devices	Subscribe to topics
OPC UA	Industrial equipment (SCADA/PLC)	Poll or subscribe

Protocol	Use Case	Injection Method
Kafka	Event streams	Consumer group
WebSocket	Real-time dashboards	Persistent connection

Configuration:

```
POST /api/admin/cortex/telemetry-feeds
{
  "name": "pump_302_sensors",
  "protocol": "opc_ua",
  "endpoint": "opc.tcp://plc.factory.local:4840",
  "nodeIds": ["ns=2;s=Pump302.Pressure", "ns=2;s=Pump302.Temperature"],
  "pollInterval": 1000,
  "contextInjection": true
}
```

Business Value: When a user asks "Why is Pump 302 showing high pressure?", the AI sees: - Current sensor values (Hot tier - real-time) - Equipment hierarchy and dependencies (Warm tier - graph) - Historical maintenance records (Cold tier - archives)

3.4 Monitoring Hot Tier

Key metrics to watch:

Metric	Warning	Critical	Action
Memory Usage %	$> 70\%$	$> 85\%$	Scale shards or reduce TTL
Cache Hit Rate	$< 90\%$	$< 80\%$	Review prefetch strategy
p99 Latency	> 5ms	> 10ms	Check network, cluster health
Connection Count	$> 80\%$ max	$> 95\%$ max	Scale or connection pooling

---

4. Warm Tier Administration

4.1 Neptune Configuration

Setting	Default	Description
`warm_neptune_mode`	`serverless`	Serverless or provisioned
`warm_neptune_min_capacity`	`1.0`	Minimum NCUs
`warm_neptune_max_capacity`	`16.0`	Maximum NCUs
`warm_retention_days`	`90`	Before archival
`warm_graph_weight_percent`	`60`	Graph vs vector weight

Setting	Default	Description
`warm_vector_weight_percent`	`40`	In hybrid search

4.2 Graph-RAG Knowledge Graph

The Warm tier implements **Graph-RAG** for superior reasoning:

Why Graph Beats Vector-Only

Scenario	Vector Search	Graph-RAG
"What causes X?"	Returns similar docs	Traverses CAUSES edges
"What depends on Y?"	Returns related docs	Follows DEPENDS_ON paths
"What supersedes Z?"	May return old versions	Explicit SUPERSEDES edges

Node Types

Type	Description	Evergreen
`document`	Source documents	No
`entity`	Named entities (Equipment, People, Orgs)	No
`concept`	Abstract concepts	No
`procedure`	Business procedures ("If X happens, do Y")	**Yes**
`fact`	Verified facts	**Yes**
`golden_qa`	Verified Q&A pairs (Golden Answers)	**Yes**

Golden Rules (Override System) **Critical Feature:** Administrators can create high-priority rules that supersede all other data.

Rule Type	Description	Priority
`force_override`	Always use this answer for this entity	**Highest**
`ignore_source`	Never use data from this source	High
`prefer_source`	Prefer this source over others	Medium
`deprecate`	Mark as obsolete (e.g., "Ignore Manual v1")	High

Creating a Golden Rule:

```
POST /api/admin/cortex/golden-rules
{
  "entityId": "pump_302",
  "ruleType": "force_override",
```

```
  "condition": "max_pressure_query",
  "override": "100 PSI (verified by Chief Engineer, Jan 2026)",
  "reason": "Manual v1 was incorrect",
  "verifiedBy": "bob@company.com",
  "signature": "sha256:abc123..."
}
```

Chain of Custody: Every Golden Rule includes: - Who verified it - When it was verified - Digital signature for audit trail - Reason for override

Edge Types

Edge	Meaning
`mentions`	Document mentions entity
`causes`	Causal relationship
`depends_on`	Dependency relationship
`supersedes`	Version replacement
`verified_by`	Source verification
`authored_by`	Authorship attribution
`relates_to`	General relationship
`contains`	Containment
`requires`	Prerequisite

4.3 pgvector Integration

Hybrid search combines: 1. **Graph traversal** (60% weight): Neptune path queries 2. **Vector similarity** (40% weight): pgvector cosine distance

```
-- Example hybrid query
SELECT n.*,
       (0.6 * graph_score + 0.4 * (1 - embedding <=> query_vector)) as hybrid_score
FROM cortex_graph_nodes n
WHERE tenant_id = $1
ORDER BY hybrid_score DESC
LIMIT 10;
```

4.4 Conflict Detection

The system automatically detects contradictory facts:

Conflict Type	Description	Resolution
`contradiction`	Mutually exclusive facts	Manual review required
`superseded`	Newer fact replaces older	Auto-archive older
`ambiguous`	Unclear relationship	Flag for clarification

---

5. Cold Tier Administration

5.1 S3 Iceberg Configuration

Setting	Default	Description
`cold_s3_bucket`	Auto-generated	Archive bucket
`cold_iceberg_enabled`	`true`	Use Iceberg tables
`cold_compression_format`	`snappy`	snappy, zstd, or gzip
`cold_zero_copy_enabled`	`false`	Enable external mounts

5.2 Storage Class Lifecycle

Data automatically transitions through storage classes:

```
Day 0-30:     S3 Standard
Day 30-90:   S3 Intelligent-Tiering
Day 90-365:  Glacier Instant Retrieval
Day 365+:    Glacier Deep Archive
```

5.3 Zero-Copy Mounts & Stub Nodes

The Innovation: We do not force tenants to move 50TB of data to our cloud. We **Mount** their existing Data Lakes.

The Mechanism: RADIANT scans external storage metadata and generates **"Stub Nodes"** in the Warm Graph: - Stub Node example: `"Log File 2024.csv exists at S3://bucket/logs/"` - Actual content is fetched **only** if Graph Traversal determines it is critical for the answer - Enables sub-second metadata queries over petabytes of external data

Source Type	Description	Connection Method
`snowflake`	Snowflake Data Share	OAuth + Data Share
`databricks`	Delta Lake / Unity Catalog	Service Principal
`s3`	Customer S3 bucket	Cross-account IAM role
`azure_datalake`	Azure Data Lake Gen2	Managed Identity
`gcs`	Google Cloud Storage	Service Account

Creating a Zero-Copy Mount

```
POST /api/admin/cortex/mounts
{
  "name": "customer-data-lake",
  "sourceType": "snowflake",
  "connectionConfig": {
    "account": "xy12345.us-east-1",
    "warehouse": "COMPUTE_WH",
    "database": "CUSTOMER_DB",
    "schema": "PUBLIC"
  }
}
```

Rescanning a Mount

`POST /api/admin/cortex/mounts/{mountId}/rescan`

This triggers: 1. Catalog synchronization 2. Schema discovery 3. Node creation for new objects 4. Index updates

5.4 Archive Retrieval

Cold data can be retrieved on-demand:

Storage Class	Retrieval Time	Cost
Standard	Immediate	Base
Intelligent-Tiering	Immediate	Base
Glacier Instant	~100ms	Higher
Glacier Flexible	1-12 hours	Lower
Deep Archive	12-48 hours	Lowest

---

6. Tenant Isolation & Security

6.1 Defense-in-Depth Strategy

Tier	Isolation Mechanism
Hot	Redis key prefixing + ACLs
Warm	Neptune IAM policies + PostgreSQL RLS
Cold	S3 bucket policies + KMS per-tenant keys

6.2 Hot Tier Security

```
# Redis key prefix enforcement
Key pattern: {tenant_id}:*
ACL: user tenant_{id} on +@all ~{tenant_id}:*
```

6.3 Warm Tier Security

PostgreSQL Row-Level Security:

```
CREATE POLICY cortex_graph_nodes_isolation ON cortex_graph_nodes
    USING (tenant_id = current_setting('app.current_tenant_id')::UUID);
```

Neptune IAM policy scoping:

```
{
  "Effect": "Allow",
  "Action": ["neptune-db:*"],
  "Resource": "arn:aws:neptune-db:*:*:cluster/*/database/*",
  "Condition": {
    "StringEquals": {
```

```
      "neptune-db:QueryLanguage": "Gremlin",
      "aws:ResourceTag/TenantId": "${aws:PrincipalTag/TenantId}"
    }
  }
}
```

6.4 Cold Tier Security

S3 bucket policy:

```
{
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::*:role/tenant-*"},
  "Action": ["s3:GetObject"],
  "Resource": "arn:aws:s3:::cortex-cold/*",
  "Condition": {
    "StringLike": {
      "s3:prefix": ["${aws:PrincipalTag/TenantId}/*"]
    }
  }
}
```

6.5 Compliance Requirements Matrix

Requirement	Hot Tier	Warm Tier	Cold Tier
Encryption at rest	AES-256	AES-256	KMS CMK
Encryption in transit	TLS 1.3	TLS 1.3	TLS 1.3
Audit logging	CloudWatch	CloudTrail	S3 Access Logs
Retention control	TTL-based	Policy-based	Lifecycle rules
GDPR erasure	Immediate	24h SLA	72h SLA
Data residency	Region-locked	Region-locked	Region-locked

---

7. Dashboard Operations

7.1 Accessing Cortex Dashboard

Navigate to: **Admin Dashboard → Memory → Cortex**

7.2 Dashboard Pages

Overview Page (`/cortex`) Displays: - **Tier Health Cards**: Status for Hot/Warm/Cold - **Data Flow Metrics**: Promotions, archivals, retrievals - **Active Alerts**: Threshold violations - **Zero-Copy Mounts**: Connected data sources - **Model Migration**: One-Click Swap

Model Migration (`/cortex/model-migration`) Swap AI models without losing your Cortex knowledge:

Supported Models: | Provider | Model | Max Tokens | Images | Tools | |———-|——-|————|———|——-| | Anthropic | Claude 3 Opus | 200K | | | | Anthropic | Claude 3 Sonnet | 200K | | | | OpenAI | GPT-4 Turbo | 128K | | | | OpenAI | GPT-4o | 128K | | | | Meta | Llama 3 70B | 8K | | | | Google | Gemini Pro | 32K | | |

Migration Workflow: 1. **Initiate** - Select target model 2. **Validate** - Check feature compatibility, estimate cost change 3. **Test** - Run accuracy, latency, cost, safety tests 4. **Execute** - Switch to new model 5. **Rollback** - Revert if needed (available for 7 days)

API:

```
POST /api/admin/cortex/v2/model-migrations
{
  "targetModel": { "provider": "meta", "modelId": "llama-3-70b-instruct" }
}
```

Graph Explorer (`/cortex/graph`) Features: - Visual knowledge graph exploration - Node/edge filtering by type - Search across labels - Confidence score display - Source document links

Conflicts Page (`/cortex/conflicts`) Shows: - Contradictory fact pairs - Conflict type classification - Resolution actions - Audit trail

GDPR Erasure (`/cortex/gdpr`) Manages: - Erasure request queue - Per-tier completion status - Audit log retention flag - Compliance documentation

---

8. Housekeeping & Maintenance

8.1 Twilight Dreaming Integration

Cortex integrates with the Twilight Dreaming background process:

Task	Frequency	Description
`ttl_enforcement`	Hourly	Expire Hot tier keys
`archive_promotion`	Nightly	Move Warm → Cold
`deduplication`	Nightly	Merge duplicate nodes
`conflict_resolution`	Nightly	Flag contradictions
`iceberg_compaction`	Nightly	Optimize Cold storage
`index_optimization`	Weekly	Reindex vectors
`integrity_audit`	Weekly	Cross-tier consistency
`storage_report`	Weekly	Cost analysis

8.2 Manual Task Trigger

```
POST /api/admin/cortex/housekeeping/trigger
{
  "taskType": "deduplication"
}
```

8.3 Task Status Monitoring

`GET /api/admin/cortex/housekeeping/status`

Returns:

```
[
  {
    "taskType": "archive_promotion",
    "frequency": "nightly",
    "status": "completed",
    "lastRunAt": "2026-01-23T04:00:00Z",
    "nextRunAt": "2026-01-24T04:00:00Z",
    "lastResult": {
      "success": true,
      "recordsProcessed": 1542,
      "errorsEncountered": 0,
      "durationMs": 3421
    }
  }
]
```

---

9. GDPR Compliance

9.1 Article 17 Erasure Process

GDPR "Right to be Forgotten" requires cascade deletion:

GDPR ERASURE CASCADE

Request Received	Hot Tier Delete	Warm Tier Anonymize	Cold Tier Tombstone
T+0	Immediate	24h SLA	72h SLA

9.2 Creating an Erasure Request

```
POST /api/admin/cortex/gdpr/erasure
{
  "targetUserId": "user_123",  // null for tenant-wide
  "scopeType": "user",         // or "tenant"
  "reason": "User request via support ticket #456"
}
```

9.3 Erasure Status Tracking

`GET /api/admin/cortex/gdpr/erasure`

Returns status per tier:

```
{
  "id": "erasure_789",
  "status": "processing",
  "hot_tier_status": "completed",
  "warm_tier_status": "completed",
  "cold_tier_status": "pending",
  "audit_log_retained": true,
  "requestedAt": "2026-01-23T10:00:00Z"
}
```

9.4 Audit Trail Retention

Even after erasure: - **Retained**: Anonymized audit entries (for compliance proof) - **Deleted**: All PII and content data

---

10. Monitoring & Alerts

10.1 Key Metrics

Hot Tier Metrics

Metric	Warning	Critical
`redis_memory_usage_percent`	> 70%	> 85%
`redis_cache_hit_rate`	< 90%	< 80%
`redis_p99_latency_ms`	> 5ms	> 10ms
`redis_connection_count`	> 80% limit	> 95% limit

Warm Tier Metrics

Metric	Warning	Critical
`neptune_cpu_percent`	> 70%	> 90%
`neptune_query_latency_p99_ms`	> 80ms	> 150ms
`graph_node_count`	> 50M	> 100M
`pgvector_index_size`	> 100GB	> 500GB

Cold Tier Metrics

Metric	Warning	Critical
`s3_storage_cost_usd`	> budget * 0.8	> budget
`iceberg_compaction_lag_hours`	> 24h	> 72h
`zero_copy_mount_error_count`	> 5/day	> 20/day

10.2 Data Flow Metrics

Metric	Description
`hot_to_warm_promotions`	Records moved Hot → Warm
`warm_to_cold_archivals`	Records moved Warm → Cold
`cold_to_warm_retrievals`	Records restored Cold → Warm
`tier_miss_rate`	Cache misses requiring tier traversal

10.3 Alert Configuration

Alerts are created automatically when thresholds are exceeded:

```
{
  "id": "alert_123",
  "tier": "hot",
  "severity": "warning",
  "metric": "redis_memory_usage",
  "threshold": 70,
  "currentValue": 75.5,
  "message": "Redis memory usage exceeds 70%",
  "triggeredAt": "2026-01-23T14:30:00Z"
}
```

10.4 Acknowledging Alerts

`POST /api/admin/cortex/alerts/{alertId}/acknowledge`

---

11. Troubleshooting

11.1 Hot Tier Issues

Symptom	Cause	Resolution
High memory usage	TTL too long, insufficient shards	Reduce TTL, add shards
Low cache hit rate	Poor prefetch strategy	Review access patterns
High latency	Network issues, cluster split	Check VPC, node health
Connection errors	Pool exhaustion	Increase pool size, add replicas

11.2 Warm Tier Issues

Symptom	Cause	Resolution
Slow graph queries	Unoptimized traversals	Add indexes, review query patterns
High CPU usage	Complex queries, under-provisioned	Scale NCUs, optimize queries

Symptom	Cause	Resolution
Vector index slow	Too many dimensions	Consider dimensionality reduction
Conflicts piling up	No resolution process	Assign reviewers, automate

11.3 Cold Tier Issues

Symptom	Cause	Resolution
High storage costs	Incorrect lifecycle rules	Review and adjust transitions
Slow retrievals	Data in Deep Archive	Use Glacier Instant for frequent access
Mount scan failures	Credential expiration	Rotate credentials
Iceberg compaction lag	Large partition count	Tune compaction schedule

11.4 Cross-Tier Issues

Symptom	Cause	Resolution
High tier miss rate	Data not warming up	Enable auto-promotion
Promotion failures	Schema mismatch	Check migration scripts
Inconsistent data	Replication lag	Review Tier Coordinator logs

---

12. API Reference

Base URL

`/api/admin/cortex`

Endpoints

Dashboard & Config

Method	Endpoint	Description
GET	`/overview`	Full dashboard data
GET	`/config`	Current tier configuration
PUT	`/config`	Update configuration

Health & Alerts

Method	Endpoint	Description
GET	`/health`	Tier health status
POST	`/health/check`	Trigger health check
GET	`/alerts`	Active alerts
POST	`/alerts/{id}/acknowledge`	Acknowledge alert

Metrics

Method	Endpoint	Description
GET	`/metrics?period=day`	Data flow metrics

Graph Explorer

Method	Endpoint	Description
GET	`/graph/stats`	Node/edge type counts
GET	`/graph/explore?search=...`	Search and explore nodes
GET	`/graph/conflicts`	Unresolved conflicts

Housekeeping

Method	Endpoint	Description
GET	`/housekeeping/status`	All task statuses
POST	`/housekeeping/trigger`	Run task manually

Zero-Copy Mounts

Method	Endpoint	Description
GET	`/mounts`	List mounts
POST	`/mounts`	Create mount
POST	`/mounts/{id}/rescan`	Rescan mount
DELETE	`/mounts/{id}`	Delete mount

GDPR

Method	Endpoint	Description
GET	`/gdpr/erasure`	List erasure requests
POST	`/gdpr/erasure`	Create erasure request

Appendix A: Implementation Checklist

Infrastructure

- ☐ Redis cluster deployed with tenant key isolation
- ☐ Neptune cluster or serverless configured
- ☐ S3 bucket with Iceberg tables created
- ☐ IAM policies scoped per tenant
- ☐ KMS keys created for encryption
- ☐ VPC endpoints configured

Database

- ☐ `cortex_config` table exists
- ☐ `cortex_graph_nodes` with RLS enabled
- ☐ `cortex_graph_edges` with RLS enabled
- ☐ `cortex_cold_archives` tracking table
- ☐ `cortex_zero_copy_mounts` configured
- ☐ Vector indexes created

Monitoring

- ☐ CloudWatch dashboards created
- ☐ Alert thresholds configured
- ☐ PagerDuty/OpsGenie integration
- ☐ Cost anomaly detection enabled

Operations

- ☐ Housekeeping tasks initialized
- ☐ Backup schedules confirmed
- ☐ DR procedures documented
- ☐ Runbooks created

Document Version: 4.20.0
For engineering implementation details, see CORTEX-ENGINEERING-GUIDE.md