# Pattern Matching, Logger, and Analytics: Quick Reference

## System Overview

Three-layer architecture:

1. **Pattern Matching** - Fast command recognition (bypasses AI)
2. **Logger** - Event tracking (non-blocking)
3. **Analytics** - Insights from logs (SQL views)

**Data Flow**: SMS → Pattern Match → Action → Log → Database → Analytics Views

---

## Pattern Matching

### How It Works

- Runs **before AI** to catch common commands
- Uses regex patterns and state-aware matching
- Extracts structured data (names, numbers, dates)
- Returns action + extracted data, or null (triggers AI)

### Pattern Types

1. **Global Commands**: RESET, EXIT (highest priority)

2. **State-Aware**: Menu selections, field editing (based on `waiting_for`)
3. **Regex Patterns**: "create crew", "check rsvps", "add member"
4. **Natural Language**: "can I create a crew?", "my group is X"

### Priority Order

1. Message validation
2. Global commands (RESET, EXIT)
3. State-specific patterns
4. Context-aware patterns
5. General patterns
6. Fallback to AI

### Benefits

- **Latency**: < 10ms (vs AI ~500ms)
- **Cost**: Bypasses AI for 90% of commands
- **Reliability**: Deterministic for recognized patterns

---

## Logger

---

### How It Works

- **Fire-and-forget**: Uses `EdgeRuntime.waitUntil()` (non-blocking)
- **Structured**: All logs follow `LogParams` interface
- **PII Compliant**: No message bodies, only phone numbers already in system

## Event Types

- **Organizer Events:** `flow_started`, `crew_created`, `event_created`, `reminder_sent`, etc.
- **Invitee Events:** `invite_sent`, `invitee_reply_yes/no/unknown`, `invitee_vote`
- **System Events:** `sms_sent`, `sms_received`, `error`

## Data Structure

- **Columns:** `organizer_id`, `event_type`, `crew_id`, `event_id`, `sync_up_id`, `invitee_contact_id`, `workflow_name`, `workflow_step`, `timestamp`
- **JSONB Metadata:** `input_data`, `output_data`, `error_details`, enriched relational IDs

## Setup

- Already integrated in `supabase/functions/funlet-sms-handler-v2/logger.ts`
- Requires `behavioral_logs` table in database
- No additional configuration needed

## Key Features

- Never blocks main request
- Never throws errors (graceful degradation)
- Automatically enriches metadata with relational IDs
- Version 1, platform 'sms'

---

# Analytics

## How It Works

- **SQL Views**: Pre-computed queries on `behavioral_logs` table
- **Real-Time**: Views computed on-demand (always up-to-date)
- **No Additional Storage**: Everything derived from logs

## Setup (One-Time)

> **Already Set Up**: The analytics views are already configured in Supabase SQL Editor:
>
> 1. **"Setup Behavioral Logs Analytics Views & Indexes"** - Contains the setup SQL
>    - Run this file once to create all views and indexes
>    - Creates 15+ analytics views and 7 performance indexes
>
>    https://supabase.com/dashboard/project/jjkduivjlzazcvdeeqde/sql/b2cff120-66c1-49ad-aebb-b9452ac98fc7
>
> 2. **"Funlet Analytics Dashboard"** - Contains all query examples
>    - Ready-to-use queries for all analytics views
>    - Can run individual queries or sections as needed
>
>    https://supabase.com/dashboard/project/jjkduivjlzazcvdeeqde/sql/b4454f09-314f-44c1-af46-8447c63e051b

**To Use:**

1. Open Supabase Dashboard → SQL Editor
2. Find "Setup Behavioral Logs Analytics Views & Indexes" (if not run yet)

3. Run it to create views (one-time setup)
4. Use "Funlet Analytics Dashboard" to query any view
5. Verify with `SELECT * FROM analytics_summary;`

## Views Created

### Summary Views

- `analytics_summary` - All key metrics in one row
- `analytics_flow_activity_summary` - Aggregate flow activity
- `analytics_invitee_behavior_summary` - Invitee metrics totals

### Flow Metrics

- `analytics_flow_activity` - Per-organizer activity
- `analytics_flow_completion` - Completion rates by workflow
- `analytics_flow_dropoffs` - Drop-off counts
- `analytics_flow_performance` - Combined (starts, completions, drop-offs)

### Invitee Metrics

- `analytics_rsvp_response_rates` - Yes/no/unknown percentages
- `analytics_rsvp_by_event` - Responses per event
- `analytics_syncup_votes` - Vote distribution
- `analytics_time_to_reply` - Average/median hours to reply
- `analytics_time_to_vote` - Average/median hours to vote

### Completion Metrics

- `analytics_reminder_effectiveness` - Reminder response rates
- `analytics_non_responders` - Invitees who never replied

### System Health

- `analytics_system_health` - SMS, errors, push notifications
- `analytics_errors_by_type` - Error frequency by type
- `analytics_unrecognized_replies_timeline` - Unrecognized replies over time

## Indexes Created

- `event_type` (most common filter)
- `timestamp` (time queries)
- `organizer_id`, `event_id`, `sync_up_id`, `invitee_contact_id`
- Composite: `event_type, timestamp`

## Querying

### Quick Start

```
SELECT * FROM analytics_summary;
SELECT * FROM analytics_flow_performance ORDER BY flows_started DESC;
SELECT * FROM analytics_invitee_behavior_summary;
```

### Query All Views

- Use the **"Funlet Analytics Dashboard"** SQL file in Supabase SQL Editor
- Contains queries for all analytics views
- Run individual queries or sections as needed

## Exporting

1. Run query in Supabase SQL Editor
2. Click "Export" → CSV or JSON
3. Use in Excel, Sheets, or data tools

---

# How They Work Together

## Example: Creating a Crew

1. **SMS**: "create crew My Team"
2. **Pattern Match**: `checkCreateCrewPattern()` matches, extracts "My Team"
3. **Action**: CREATE_CREW executed, crew created
4. **Logging**:
    - `logWorkflowStart()` → `flow_started`
    - `logCrewCreated()` → `crew_created`
    - `logWorkflowComplete()` → `flow_completed`
5. **Database**: 3 log entries in `behavioral_logs`
6. **Analytics**: Views automatically reflect new data

## State Management

- Pattern matching uses `conversation_state.waiting_for`
- Logger includes `workflow_step` (often matches `waiting_for`)
- Analytics can query by `workflow_step` for drop-off analysis

---

# Creating Reports for Partners

## What Partners See

- Aggregate metrics only (no individual user data)
- Workflow performance (completion rates, drop-offs)

- Invitee engagement (response rates, time to reply)
- System health (errors, SMS delivery)

### Privacy

- No PII (names, emails, message content)
- No individual data (only aggregates)
- Phone numbers: counts only, not actual numbers

---

## Key Takeaways

- **Pattern Matching**: Fast, rule-based, bypasses AI for 90% of commands
- **Logger**: Non-blocking, structured, PII-compliant event tracking
- **Analytics**: SQL views, real-time, no additional infrastructure
- **Integration**: All three work together seamlessly
- **Setup**: One-time SQL file execution, then automatic

---

## Quick Reference

- **Setup Analytics**: Run **"Setup Behavioral Logs Analytics Views & Indexes"** in Supabase SQL Editor (one-time)
- **Query Views**: `SELECT * FROM analytics_summary;`
- **Query All**: Use **"Funlet Analytics Dashboard"** SQL file in Supabase SQL Editor
- **Export**: Click "Export" in Supabase SQL Editor → CSV/JSON

**Note:** Both SQL files are already saved in Supabase SQL Editor for easy access