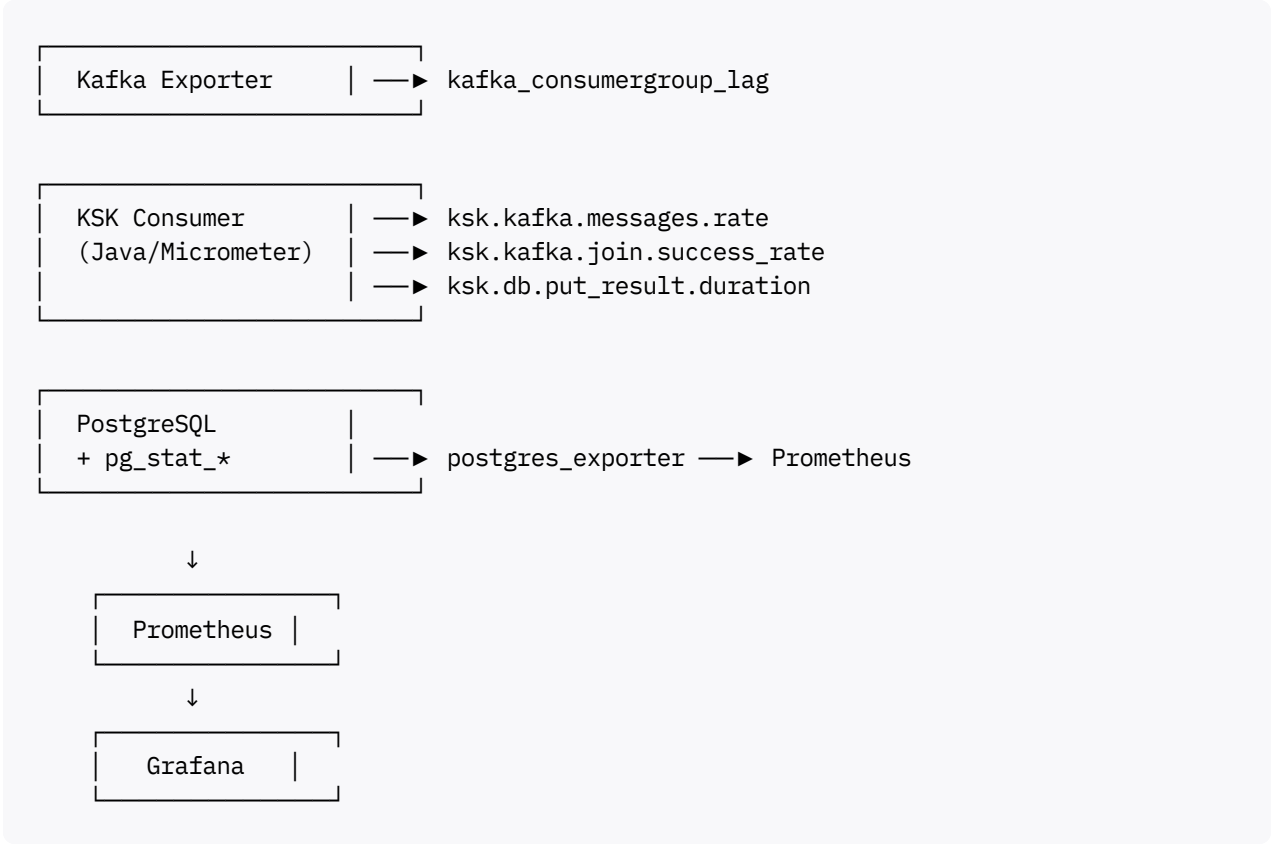
PostgreSQL Monitoring Specification для КСК

Дата: 28.10.2025
Цель: Мониторинг PostgreSQL БД КСК через стандартные метрики (без расширений и custom queries)

Архитектура



1. Подключения (Connections)

Назначение: Контроль количества активных подключений к БД

Метрика: `pg_stat_database_numbackends`

Источник данных:

```
SELECT count(*)
FROM pg_stat_activity
```

```
WHERE state = 'active' AND datname = 'ksk_database';
```

Пороги:

- До 300: норма
- 300-400: предупреждение
- 400+: критично

PromQL:

```
pg_stat_database_numbackends{datname="ksk_database"}
```

2. Скорость транзакций (TPS)

Назначение: Мониторинг интенсивности записи данных

Метрика: `pg_stat_database_xact_commit`, `pg_stat_database_xact_rollback`

Источник данных:

```
SELECT
    datname,
    xact_commit,
    xact_rollback
FROM pg_stat_database
WHERE datname = 'ksk_database';
```

Пороги:

- 30-50 TPS: норма (~3M записей/день)
- <10 TPS: критично (остановка потока данных)
- 1000 TPS: критично (аномальная нагрузка)

PromQL:

```
rate(pg_stat_database_xact_commit{datname="ksk_database"}[5m])
```

3. Размер БД и таблиц

Назначение: Контроль заполнения дискового пространства

Метрики: `pg_database_size_bytes`, `pg_table_size_bytes`

Источник данных:

```
-- Размер БД
SELECT pg_database_size('ksk_database');

-- Размер таблицы
SELECT pg_total_relation_size('upoa_ksk_reports.ksk_result');

-- Топ таблиц по размеру
SELECT
    schemaname,
    tablename,
    pg_total_relation_size(schemaname||'.'||tablename) AS bytes
FROM pg_tables
WHERE schemaname = 'upoa_ksk_reports'
ORDER BY bytes DESC
LIMIT 10;
```

Пороги:

- Рост: ~100 GB/месяц
- Предупреждение: >80% диска
- Критично: >90% диска

PromQL:

```
pg_database_size_bytes{datname="ksk_database"}
```

4. Раздутие таблиц (Bloat)

Назначение: Выявление неэффективного использования дискового пространства

Метрики: `pg_stat_user_tables_n_dead_tup`, `pg_stat_user_tables_n_live_tup`

Источник данных:

```
SELECT
    schemaname,
    relname AS tablename,
    n_dead_tup,
    n_live_tup,
    ROUND(100.0 * n_dead_tup / NULLIF(n_live_tup + n_dead_tup, 0), 2) AS dead_pct
FROM pg_stat_user_tables
WHERE schemaname = 'upoa_ksk_reports'
  AND n_live_tup > 0
ORDER BY dead_pct DESC;
```

Пороги:

- 0-10%: норма
- 10-30%: предупреждение (планировать VACUUM)

- 30%: критично (выполнить VACUUM немедленно)

PromQL:

```
(pg_stat_user_tables_n_dead_tup{schemaname="upoa_ksk_reports"} /
(pg_stat_user_tables_n_live_tup{schemaname="upoa_ksk_reports"} +
pg_stat_user_tables_n_dead_tup{schemaname="upoa_ksk_reports"})) * 100
```

5. Использование индексов

Назначение: Оценка эффективности индексирования

Метрики: `pg_stat_user_tables_idx_scan`, `pg_stat_user_tables_seq_scan`

Источник данных:

```
SELECT
    schemaname,
    tablename,
    idx_scan,
    seq_scan,
    ROUND(100.0 * idx_scan / NULLIF(idx_scan + seq_scan, 0), 2) AS idx_usage_pct
FROM pg_stat_user_tables
WHERE schemaname = 'upoa_ksk_reports'
  AND (idx_scan + seq_scan) > 0
ORDER BY idx_usage_pct ASC;
```

Пороги:

- 90% idx_scan: оптимально
- 50-90% idx_scan: требуется анализ
- <50% idx_scan: критично (индексы не используются)

PromQL:

```
pg_stat_user_tables_idx_scan{schemaname="upoa_ksk_reports"}
pg_stat_user_tables_seq_scan{schemaname="upoa_ksk_reports"}
```

6. Cache Hit Ratio

Назначение: Эффективность использования RAM кеша

Метрики: `pg_stat_database_blks_hit`, `pg_stat_database_blks_read`

Источник данных:

```
SELECT
    datname,
```

```
    blks_hit,
    blks_read,
    ROUND(100.0 * blks_hit / NULLIF(blks_hit + blks_read, 0), 2) AS cache_hit_ratio
FROM pg_stat_database
WHERE datname = 'ksk_database';
```

Пороги:

- 99%: оптимально
- 95-99%: норма
- <95%: недостаточно RAM или плохие запросы

PromQL:

```
(pg_stat_database_blks_hit{datname="ksk_database"} /
(pg_stat_database_blks_hit{datname="ksk_database"} +
pg_stat_database_blks_read{datname="ksk_database"})) * 100
```

7. Блокировки (Locks)

Назначение: Обнаружение конфликтов и deadlocks

Метрики: `pg_locks`, `pg_stat_database_deadlocks`

Источник данных:

```
-- Активные блокировки
SELECT
    locktype,
    COUNT(*) AS lock_count
FROM pg_locks
WHERE granted = true
GROUP BY locktype
ORDER BY lock_count DESC;

-- Deadlocks
SELECT
    datname,
    deadlocks
FROM pg_stat_database
WHERE datname = 'ksk_database';
```

Пороги:

- 0-5 locks: норма
- 50 locks: требует проверки
- deadlocks >0: критично

PromQL:

```
pg_locks_count
rate(pg_stat_database_deadlocks{datname="ksk_database"}[5m])
```

8. Checkpoints

Назначение: Мониторинг производительности дисковой подсистемы

Метрики: `pg_stat_bgwriter_checkpoints_timed`, `pg_stat_bgwriter_checkpoints_req`

Источник данных:

```
SELECT
    checkpoints_timed,
    checkpoints_req,
    buffers_checkpoint,
    buffers_clean,
    buffers_backend
FROM pg_stat_bgwriter;
```

Пороги:

- checkpoints_req < 10% от checkpoints_timed: норма
- checkpoints_req > 30%: требуется увеличение checkpoint_timeout

PromQL:

```
pg_stat_bgwriter_checkpoints_timed
pg_stat_bgwriter_checkpoints_req
```

9. Tuple Operations

Назначение: Контроль операций вставки/обновления/удаления

Метрики: `pg_stat_user_tables_n_tup_ins`, `pg_stat_user_tables_n_tup_upd`, `pg_stat_user_tables_n_tup_del`

Источник данных:

```
SELECT
    schemaname,
    tablename,
    n_tup_ins,
    n_tup_upd,
    n_tup_del,
    n_tup_hot_upd
FROM pg_stat_user_tables
```

```
WHERE schemaname = 'upoa_ksk_reports'
ORDER BY n_tup_ins DESC;
```

Пороги:

- Зависит от бизнес-логики приложения
- Аномальные изменения требуют расследования

PromQL:

```
rate(pg_stat_user_tables_n_tup_ins{schemaname="upoa_ksk_reports"}[5m])
rate(pg_stat_user_tables_n_tup_upd{schemaname="upoa_ksk_reports"}[5m])
rate(pg_stat_user_tables_n_tup_del{schemaname="upoa_ksk_reports"}[5m])
```

10. Database Age

Назначение: Контроль transaction ID wraparound

Метрика: `pg_database_age`

Источник данных:

```
SELECT
    datname,
    age(datfrozenxid) AS age
FROM pg_database
WHERE datname = 'ksk_database';
```

Пороги:

- <1 billion: норма
- 1-1.5 billion: предупреждение
- 1.5 billion: критично (риск wraparound)

PromQL:

```
pg_database_age{datname="ksk_database"}
```

Пользователь для мониторинга

```
CREATE USER monitoring_user WITH PASSWORD 'secure_password';
GRANT CONNECT ON DATABASE ksk_database TO monitoring_user;
GRANT pg_monitor TO monitoring_user;
GRANT USAGE ON SCHEMA upoa_ksk_reports TO monitoring_user;
GRANT SELECT ON ALL TABLES IN SCHEMA upoa_ksk_reports TO monitoring_user;
```

```
ALTER DEFAULT PRIVILEGES IN SCHEMA upoa_ksk_reports
GRANT SELECT ON TABLES TO monitoring_user;
```

Grafana Dashboard Import

Готовые dashboard ID:

- PostgreSQL Database: **9628**
- PostgreSQL Exporter: **12485**

Импорт: Grafana → Dashboards → Import → указать ID

Alerting Rules (Prometheus)

```
groups:
  - name: ksk_postgresql
    interval: 30s
    rules:
      - alert: KSKPostgreSQLConnectionsHigh
        expr: pg_stat_database_numbackends{datname="ksk_database"} > 300
        for: 5m
        labels:
          severity: warning
        annotations:
          summary: "Высокое количество подключений"
          description: "{{ $value }} активных подключений (норма <300)"

      - alert: KSKPostgreSQLConnectionsCritical
        expr: pg_stat_database_numbackends{datname="ksk_database"} > 400
        for: 2m
        labels:
          severity: critical
        annotations:
          summary: "Критическое количество подключений"
          description: "{{ $value }} активных подключений (критично >400)"

      - alert: KSKPostgreSQLTPSLow
        expr: rate(pg_stat_database_xact_commit{datname="ksk_database"}[5m]) < 10
        for: 5m
        labels:
          severity: critical
        annotations:
          summary: "Низкий TPS"
          description: "{{ $value }} TPS (критично <10)"

      - alert: KSKPostgreSQLTPSCritical
        expr: rate(pg_stat_database_xact_commit{datname="ksk_database"}[5m]) > 1000
        for: 5m
        labels:
          severity: critical
        annotations:
          summary: "Аномально высокий TPS"
```

```
          description: "{{ $value }} TPS (критично >1000)"

      - alert: KSKPostgreSQLBloatHigh
        expr: |
          (pg_stat_user_tables_n_dead_tup /
          (pg_stat_user_tables_n_live_tup + pg_stat_user_tables_n_dead_tup)) * 100 > 30
        for: 30m
        labels:
          severity: warning
        annotations:
          summary: "Высокий bloat {{ $labels.tablename }}"
          description: "Dead tuples: {{ $value }}%"

      - alert: KSKPostgreSQLCacheHitLow
        expr: |
          (pg_stat_database_blks_hit /
          (pg_stat_database_blks_hit + pg_stat_database_blks_read)) * 100 < 95
        for: 10m
        labels:
          severity: warning
        annotations:
          summary: "Низкий cache hit ratio"
          description: "{{ $value }}% (норма >95%)"

      - alert: KSKPostgreSQLDeadlocks
        expr: rate(pg_stat_database_deadlocks{datname="ksk_database"}[5m]) > 0
        for: 1m
        labels:
          severity: critical
        annotations:
          summary: "Обнаружены deadlocks"
```

Минимальные требования

PostgreSQL: Версия 10+
Расширения: Не требуются
Prometheus: Версия 2.x+
Grafana: Версия 8.x+
postgres_exporter: Версия 0.11+

Дополнительные стандартные метрики

Autovacuum

```
SELECT
    schemaname,
    tablename,
    last_vacuum,
    last_autovacuum,
    vacuum_count,
    autovacuum_count
```

```
FROM pg_stat_user_tables
WHERE schemaname = 'upoa_ksk_reports';
```

PromQL: `pg_stat_user_tables_last_autovacuum`

Temporary Files

```
SELECT
    datname,
    temp_files,
    temp_bytes
FROM pg_stat_database
WHERE datname = 'ksk_database';
```

PromQL: `pg_stat_database_temp_bytes`

Примеры Grafana Dashboards

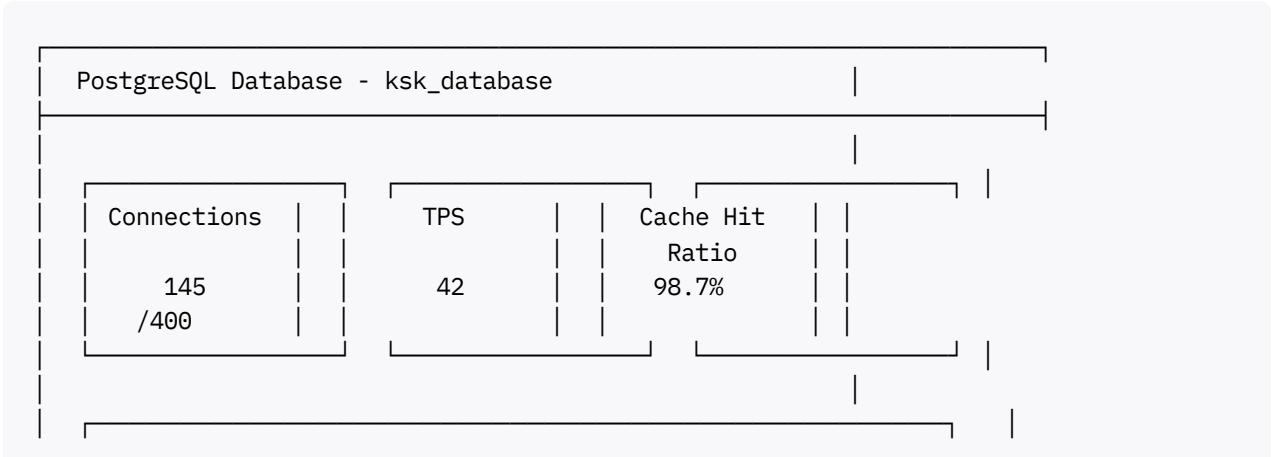
Dashboard 1: PostgreSQL Database Overview (ID: 9628)

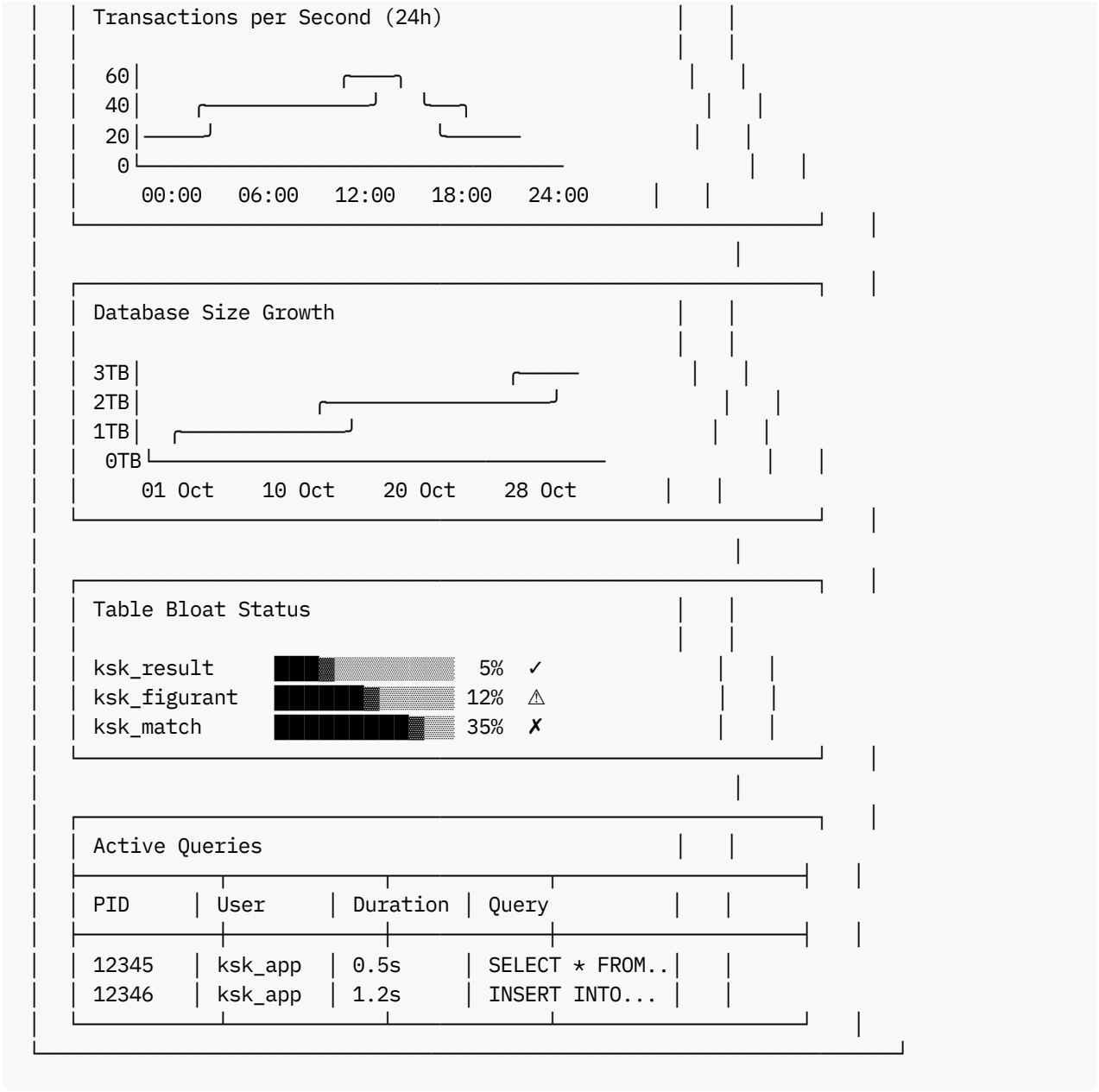
Назначение: Основной dashboard для мониторинга PostgreSQL

Компоненты:

- Database connections (gauge)
- TPS - Transactions per second (graph)
- Cache hit ratio (stat)
- Database size (graph + stat)
- Active queries (table)
- Table bloat % (bar gauge)
- Index usage statistics (table)
- Checkpoint stats (graph)

Панели:





Метрики:

- `pg_stat_database_numbackends`
- `rate(pg_stat_database_xact_commit[5m])`
- `pg_database_size_bytes`
- `pg_stat_user_tables_n_dead_tup`
- `pg_stat_activity`

Dashboard 2: PostgreSQL Exporter (ID: 12485)

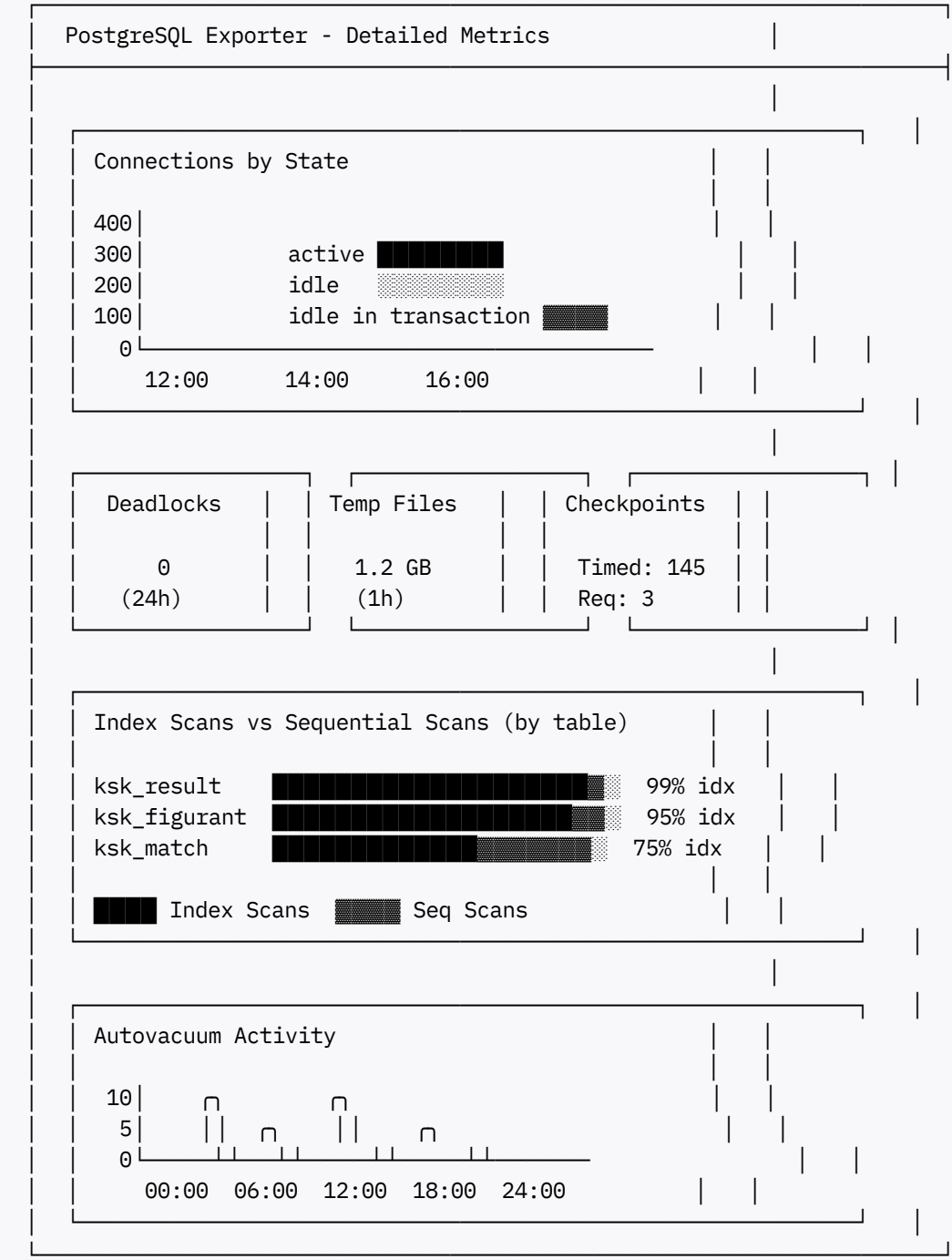
Назначение: Детальный мониторинг с query-level метриками

Компоненты:

- Database connections by state (stacked graph)

- Query execution time percentiles (heatmap)
- Lock types distribution (pie chart)
- Autovacuum activity (graph)
- Index scans vs sequential scans (stacked bar)
- Deadlocks counter (stat)

Панели:



Метрики:

- `pg_stat_activity_count` (by state)

- `rate(pg_stat_database_deadlocks[5m])`
- `pg_stat_database_temp_bytes`
- `pg_stat_bgwriter_checkpoints_timed`
- `pg_stat_user_tables_idx_scan`
- `pg_stat_user_tables_autovacuum_count`

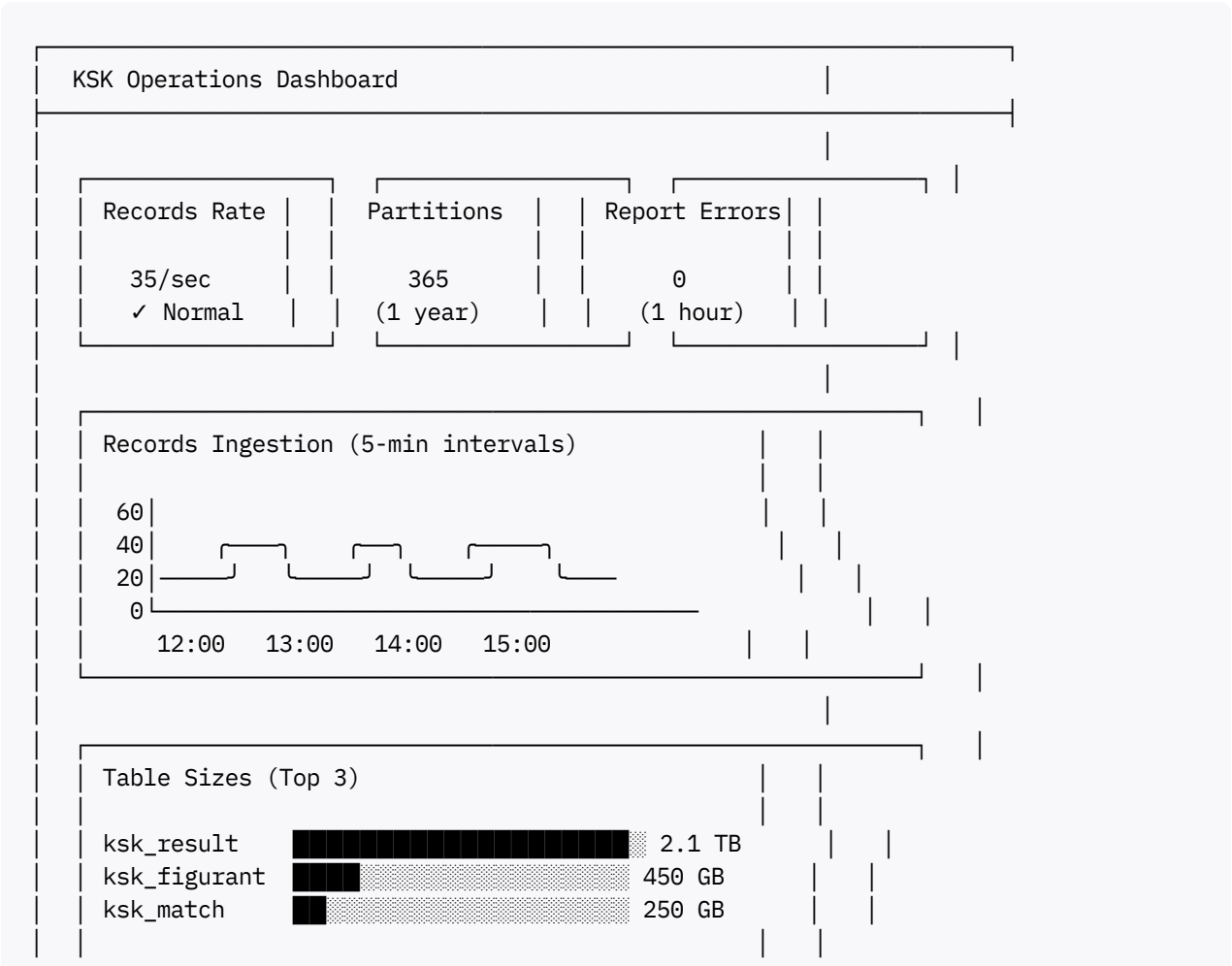
Dashboard 3: KSK Operations Custom

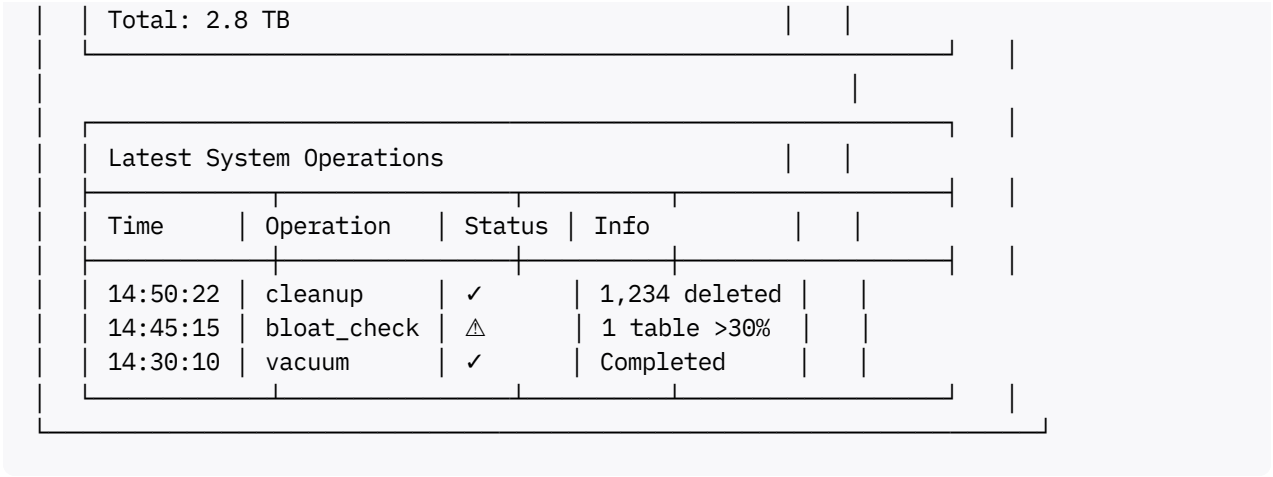
Назначение: Специализированный dashboard для КСК

Компоненты:

- KSK records ingestion rate (graph)
- Report generation status (pie chart)
- Partition count (stat)
- Table sizes for ksk_result, ksk_figurant, ksk_match (bar gauge)
- System operations log (table)
- Bloat alert status (stat panel)

Панели:





SQL для подключения к PostgreSQL в Grafana:

```
-- Records rate panel
SELECT
    $__timeGroup(output_timestamp, '5m') as time,
    COUNT(*) / 300.0 as records_per_sec
FROM upoa_ksk_reports.ksk_result
WHERE $__timeFilter(output_timestamp)
GROUP BY 1
ORDER BY 1;

-- Partition count panel
SELECT COUNT(*) as count
FROM pg_tables
WHERE schemaname = 'upoa_ksk_reports'
  AND tablename LIKE 'part_ksk_result_%';

-- Table sizes panel
SELECT
    tablename,
    pg_total_relation_size('upoa_ksk_reports.'||tablename) as bytes
FROM pg_tables
WHERE schemaname = 'upoa_ksk_reports'
  AND tablename IN ('ksk_result', 'ksk_figurant', 'ksk_match')
ORDER BY bytes DESC;
```

Импорт готовых дашбордов

Шаг 1: Grafana → Dashboards → Import
Шаг 2: Ввести Dashboard ID: **9628** или **12485**
Шаг 3: Выбрать Prometheus data source
Шаг 4: Нажать Import

Результат: Готовый dashboard с автоматической настройкой всех панелей для PostgreSQL метрик

⁂

1. https://grafana.com/solutions/postgresql/monitor/
2. https://grafana.com/grafana/dashboards/4164-vm-pgsql-box-a/
3. https://play.grafana.org
4. https://www.youtube.com/watch?v=Ufq3rUxVmCg
5. https://grafana.com/grafana/dashboards/7506-monitoring-postgresql-server-postgres-replication/
6. https://docs.oracle.com/en/learn/ocipgsql-promgra/index.html
7. https://www.reddit.com/r/webdev/comments/1o1hsxw/i_created_a_fully_selfhosted_realtime_monitoring/
8. https://grafana.com/grafana/dashboards/