План лекції.

Тема 8. Моніторинг баз даних та систем логування.

- Вступ. Роль баз даних і логування у сучасних IT-інфраструктурах
- Класифікація баз даних як об'єктів моніторингу
- Архітектура баз даних як об'єкта моніторингу
- Об'єкти моніторингу у системах баз даних
- Ключові метрики моніторингу баз даних
- Моніторинг запитів і навантаження
- Моніторинг реплікацій та кластерів баз даних
- Моніторинг бекапів та відновлення
- Логування як об'єкт моніторингу
- Агрегація та централізований збір логів
- Інструменти систем логування та SIEM
- Кореляція метрик, логів та подій
- Безпека моніторингу баз даних і логів
- Побудова системи моніторингу БД та логування
- Типові проблеми та помилки
- Практичні сценарії та кейси
- Місце моніторингу баз даних і логування в загальній системі спостережуваності

Вступ. Роль баз даних і логування у сучасних IT-інфраструктурах

У сучасних IT-інфраструктурах бази даних є центральним елементом, навколо якого фактично будується робота більшості інформаційних систем. Веб-додатки, корпоративні сервіси, мобільні застосунки, системи аналітики, фінансові платформи — усі вони у тій чи іншій формі покладаються на збереження, обробку та доступ до даних. Саме база даних забезпечує цілісність, актуальність і доступність інформації, яка є ключовим ресурсом для бізнесу.

На відміну від багатьох інших компонентів інфраструктури, база даних зазвичай зберігає стан системи. Якщо веб-сервер або контейнер можна швидко перезапустити чи масштабувати, то втрата або пошкодження даних у базі даних має значно серйозніші наслідки. Тому бази даних майже завжди належать до критичних об'єктів IT-інфраструктури, а їх стабільна робота є обов'язковою умовою безперервності сервісів.

Практика експлуатації IT-систем показує, що проблеми з продуктивністю або доступністю дуже часто зводяться саме до рівня бази даних. Це пояснюється кількома факторами. По-перше, база даних обслуговує запити від багатьох клієнтів одночасно, виконуючи складні операції читання, запису та обробки транзакцій. По-друге, вона тісно пов'язана з дисковою підсистемою, мережевими затримками та ефективністю використання пам'яті.

Навіть незначні помилки в проєктуванні запитів, індексів або схем даних можуть призводити до лавиноподібного зростання часу відповіді. Крім того, бази даних часто стають точкою концентрації ризиків: перевантаження, блокування, конфлікти транзакцій, проблеми реплікації або збої під час резервного копіювання. Саме тому моніторинг баз даних має бути глибшим і більш деталізованим, ніж просте спостереження за станом сервісу «працює / не працює».

Рис.8.01. Моніторинг БД і логів — основа керованої, надійної та безпечної системи.

Логування є ще одним фундаментальним елементом сучасних IT-систем. Логи фіксують події, які відбуваються в системі: запити користувачів, помилки, попередження, системні повідомлення, дії адміністраторів, зміни конфігурацій. На відміну від метрик, які зазвичай показують агреговану картину, логи дозволяють побачити деталі конкретних подій у часі.

З точки зору експлуатації, логи є незамінним джерелом інформації під час розслідування інцидентів. Вони дозволяють відтворити послідовність подій, зрозуміти, що саме пішло не так, і визначити першопричину проблеми. Окрім цього, логування відіграє важливу роль у питаннях безпеки та аудиту, оскільки саме журнали подій часто використовуються для виявлення несанкціонованого доступу, атак або порушень політик безпеки.

Сучасний підхід до моніторингу все частіше описується через концепцію спостережуваності (Observability), яка ґрунтується на трьох основних складових: метриках, логах і трасах. У цьому контексті бази даних і системи логування займають особливе місце.

Метрики баз даних дозволяють оцінити загальний стан і продуктивність: навантаження, затримки, використання ресурсів, стан реплікації. Логи доповнюють цю картину деталями, показуючи конкретні помилки, повільні запити, збої або аномальні події. Разом вони формують цілісне уявлення про поведінку системи та дозволяють не лише реагувати на інциденти, а й проактивно виявляти потенційні проблеми.

Таким чином, моніторинг баз даних і логування не є ізольованими задачами, а інтегруються в загальну систему спостережуваності інфраструктури.

Для кінцевого користувача проблеми з базою даних зазвичай проявляються у вигляді повільної роботи веб-додатка, помилок, таймаутів або повної недоступності сервісу. Тому ефективний моніторинг БД безпосередньо впливає на користувацький досвід та якість сервісу.

Моніторинг баз даних повинен бути тісно пов'язаний з моніторингом веб-додатків і сервісів. Наприклад, зростання часу відповіді API може корелювати з появою повільних SQL-запитів або перевантаженням дискової підсистеми БД. Аналіз логів дозволяє простежити цей ланцюг від дії користувача до конкретної операції в базі даних.

Саме такий комплексний підхід дає змогу перейти від реактивного усунення проблем до усвідомленого управління продуктивністю, надійністю та безпекою IT-систем.

Класифікація баз даних як об'єктів моніторингу

Бази даних як об'єкти моніторингу суттєво відрізняються між собою за архітектурою, моделлю зберігання даних, сценаріями використання та вимогами до продуктивності й надійності. Розуміння цієї класифікації є важливим для правильного вибору метрик, інструментів і підходів до моніторингу.

Рис.8.02. Класифікація баз даних як об'єктів моніторингу.

Традиційною основою корпоративних інформаційних систем залишаються реляційні бази даних. Такі СУБД, як MySQL або MariaDB, PostgreSQL та Oracle Database, використовують табличну модель даних, підтримують складні SQL-запити, транзакції та механізми забезпечення цілісності даних. Для них характерна чітка структура, наявність схем, індексів і розвинених механізмів блокування та керування транзакціями. Відповідно, моніторинг реляційних баз даних зосереджується на аналізі запитів, транзакцій, блокувань, використання індексів та стану реплікації.

Паралельно з реляційними СУБД широко застосовуються NoSQL та документно-орієнтовані бази даних, зокрема MongoDB. Вони орієнтовані на роботу з напівструктурованими даними, горизонтальне масштабування та високу доступність. Для таких систем характерні інші підходи до зберігання та доступу до даних, що впливає і на підхід до моніторингу. Тут більший акцент робиться на стан кластера, балансування даних, затримки між вузлами та стабільність реплікації, ніж на класичні SQL-запити.

Ще одним важливим аспектом класифікації є поділ баз даних за типом навантаження. OLTP (Online Transaction Processing) — це системи оперативної обробки транзакцій, які орієнтовані на велику кількість коротких операцій у режимі реального часу. Для таких систем характерні часті операції читання та запису, робота з невеликими обсягами даних у межах окремих транзакцій і високі вимоги до швидкості відповіді. Тому для OLTP-баз даних критичними є мінімальні затримки, стабільність роботи під навантаженням і коректна обробка паралельних запитів. Моніторинг у цьому випадку зосереджується на часі відповіді, кількості транзакцій, блокуваннях, конфліктах та загальній пропускній здатності системи.

OLAP (Online Analytical Processing), навпаки, призначені для аналітичної обробки даних. Такі системи виконують складні запити, які можуть обробляти великі масиви інформації та працювати протягом тривалого часу. OLAP-навантаження менш чутливе до затримок окремих операцій, проте значно сильніше впливає на використання процесорних ресурсів, пам'яті та дискової підсистеми. Відповідно, моніторинг OLAP-баз даних орієнтований на контроль ресурсів, ефективність виконання складних запитів і оцінку їхнього впливу на загальну стабільність системи.

Навіть якщо використовується одна й та сама система управління базами даних, характер навантаження — транзакційний або аналітичний — суттєво змінює вимоги до моніторингу, набір ключових метрик і підходи до аналізу продуктивності.

З точки зору експлуатації важливу роль відіграє середовище, у якому працює база даних. On-premise бази даних надають повний контроль над інфраструктурою, конфігураціями та внутрішніми механізмами роботи СУБД, однак водночас вимагають детального моніторингу апаратних ресурсів, дискових підсистем, мережі та операційної системи. У такому середовищі відповідальність за доступність, продуктивність, резервне копіювання та відновлення повністю лежить на адміністраторах.

Хмарні та керовані бази даних (DBaaS, Database as a Service) реалізують інший підхід, за якого база даних надається як готовий сервіс. Провайдер хмарної платформи бере на себе розгортання СУБД, оновлення, масштабування, резервне копіювання та базові механізми відмовостійкості. Це значно знижує операційне навантаження на команди експлуатації, однак одночасно обмежує доступ до внутрішніх метрик, логів і низькорівневих параметрів роботи бази даних. У результаті моніторинг у DBaaS-середовищах здебільшого зосереджується на доступності сервісу, часу відповіді, продуктивності запитів, кількості підключень та стані реплікації в межах інтерфейсів, які надає провайдер.

Окремо слід розглядати архітектуру розгортання баз даних: одиночні інстанси, репліковані конфігурації та кластерні системи. Одиночна база даних є відносно простою з точки зору моніторингу, проте водночас виступає єдиною точкою відмови. Реплікація та кластерні рішення підвищують доступність і масштабованість, але істотно ускладнюють моніторинг, оскільки вимагають постійного контролю синхронізації даних, затримок між вузлами та коректної роботи механізмів автоматичного перемикання.

Таким чином, класифікація баз даних як об'єктів моніторингу дозволяє усвідомлено підходити до вибору метрик, інструментів і сценаріїв контролю, враховуючи не лише тип СУБД і характер навантаження, а й особливості середовища розгортання та архітектури всієї системи.

Архітектура баз даних як об'єкта моніторингу

Для ефективного моніторингу бази даних необхідно розуміти її внутрішню архітектуру та взаємодію окремих компонентів. База даних не є монолітним сервісом — вона складається з кількох логічних і фізичних шарів, кожен з яких по-різному впливає на продуктивність, стабільність

і надійність системи. Відповідно, проблеми можуть виникати на будь-якому з цих рівнів, а завдання моніторингу полягає у своєчасному виявленні таких відхилень.

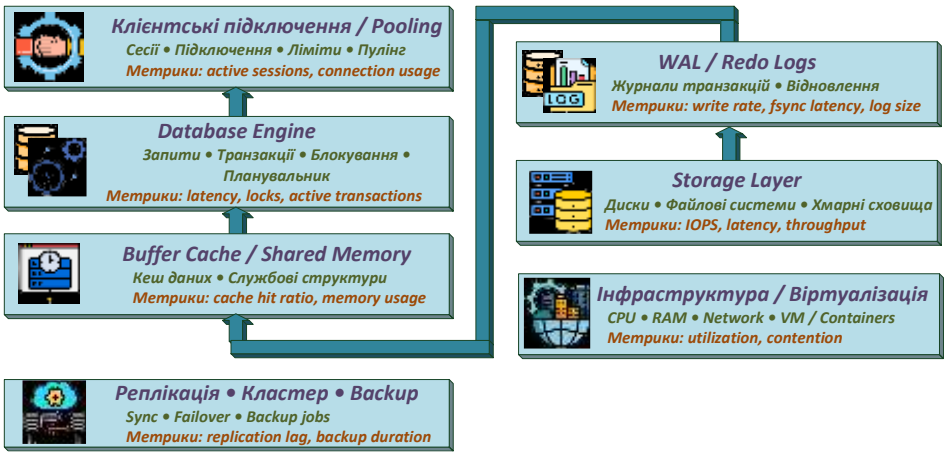

Рис.8.03. Архітектура баз даних як об'єкта моніторингу.

У центрі архітектури будь-якої системи управління базами даних знаходиться движок бази даних (database engine). Саме він відповідає за обробку запитів, планування їх виконання, управління транзакціями, блокуваннями та контролем цілісності даних. З точки зору моніторингу, стан engine безпосередньо відображається у таких показниках, як час виконання запитів, кількість активних транзакцій, наявність блокувань і конфліктів. Деградація роботи движка зазвичай швидко проявляється на рівні прикладних сервісів.

Важливим елементом є буферний кеш та спільна пам'ять (buffer cache / shared memory), які використовуються для зберігання часто запитуваних даних і службових структур у оперативній пам'яті. Ефективна робота кешу суттєво знижує навантаження на дискову підсистему та прискорює обробку запитів. Моніторинг у цій частині зосереджується на коефіцієнті попадань у кеш, використанні пам'яті та витісненні даних. Зниження ефективності кешу часто сигналізує про нестачу ресурсів або неефективні запити.

Окрему роль відіграють журнали транзакцій — WAL (Write-Ahead Log) або redo logs, залежно від конкретної СУБД. WAL — це механізм, за якого всі зміни даних спочатку фіксуються у спеціальному журналі, і лише після цього застосовуються до основних структур зберігання. Такий підхід гарантує, що у разі збою система зможе відновити цілісність даних, повторивши зафіксовані в журналі операції. Аналогічну роль у деяких СУБД виконують redo logs, які містять інформацію про всі зміни, необхідні для повторного відтворення завершених транзакцій після перезапуску системи.

Завдяки журналам транзакцій забезпечується надійність роботи бази даних і можливість відновлення після аварійного завершення, відмови живлення або збоїв обладнання. Водночас інтенсивна робота з WAL або redo logs може створювати значне навантаження на дискову підсистему, оскільки запис у журнал відбувається для кожної транзакції. Тому моніторинг швидкості запису, затримок операцій введення-виведення та заповнення журналів є критично важливим, особливо для систем із високою транзакційною активністю.

Наступним рівнем є шар зберігання даних (storage layer), який включає файлові системи, дискові масиви або хмарні сховища. Навіть оптимально налаштована СУБД не зможе працювати стабільно при проблемах зі зберіганням даних. Моніторинг цього шару охоплює показники введення-виведення, затримки доступу до диска, пропускну здатність і доступність сховища. Проблеми на рівні storage часто мають відкладений ефект, поступово погіршуючи продуктивність бази даних.

Важливою складовою архітектури є клієнтські підключення та механізми пулінгу з'єднань. Кількість одночасних підключень, їхній життєвий цикл і ефективність повторного використання з'єднань безпосередньо впливають на навантаження СУБД. Моніторинг дозволяє виявляти ситуації, коли база даних перевантажена надмірною кількістю сесій або коли неправильне налаштування пулінгу призводить до зайвих витрат ресурсів.

У сучасних середовищах бази даних рідко працюють як ізольовані інстанси. Реплікація та кластерні конфігурації забезпечують високу доступність і масштабованість, але значно ускладнюють архітектуру. З точки зору моніторингу важливо контролювати стан кожного вузла, затримки реплікації, узгодженість даних і коректність роботи механізмів автоматичного перемикання. Проблеми в реплікації можуть тривалий час залишатися непоміченими, але мати критичні наслідки у разі відмови основного вузла.

Окремим, але невід'ємним елементом екосистеми є бекап-інфраструктура. Процеси резервного копіювання та відновлення тісно інтегровані з роботою бази даних і можуть створювати додаткове навантаження. Моніторинг повинен охоплювати не лише факт виконання бекапів, а й їхню тривалість, вплив на продуктивність та коректність завершення, оскільки помилки у цій частині часто виявляються лише під час аварійного відновлення.

Робота бази даних завжди тісно пов'язана з апаратними або віртуалізованими ресурсами, на яких вона розгорнута. Процесорні ресурси, обсяг оперативної пам'яті, продуктивність дискової підсистеми та характеристики мережі безпосередньо визначають межі продуктивності СУБД. У віртуалізованих та контейнеризованих середовищах додатково з'являються фактори спільного використання ресурсів, що може призводити до нестабільної продуктивності.

З точки зору моніторингу це означає необхідність кореляції метрик бази даних з метриками інфраструктури. Проблеми, які на перший погляд виглядають як помилки СУБД, насправді можуть бути наслідком нестачі ресурсів, перевантаження хоста або конфліктів у віртуальному середовищі.

Таким чином, архітектура бази даних як об'єкта моніторингу охоплює не лише внутрішні механізми СУБД, а й усю інфраструктуру навколо неї. Лише комплексний підхід дозволяє отримати повну картину стану бази даних і забезпечити її стабільну та передбачувану роботу.

Об'єкти моніторингу у системах баз даних

Моніторинг баз даних не обмежується перевіркою факту їхньої доступності. Для отримання повної та достовірної картини стану системи необхідно розглядати базу даних як сукупність взаємопов'язаних об'єктів, кожен з яких може бути джерелом проблем або, навпаки, індикатором стабільної роботи. Виділення цих об'єктів дозволяє структурувати моніторинг і спрямувати увагу на найбільш критичні зони.

Першим і базовим об'єктом моніторингу є сервер бази даних або інстанс СУБД. Саме на цьому рівні контролюється загальний стан сервісу, його доступність, час відповіді на тестові запити та споживання ресурсів. Відмова інстансу або його нестабільна робота одразу призводять до недоступності прикладних сервісів, тому цей рівень є основою будь-якої системи моніторингу.

Тісно пов'язаними з інстансом є користувацькі сесії та підключення. Кількість активних і очікуючих з'єднань, тривалість сесій та частота відкриття нових підключень безпосередньо впливають на навантаження СУБД. Надмірна кількість сесій або неправильна робота пулінгу з'єднань часто стають причиною деградації продуктивності, навіть якщо апаратні ресурси ще не вичерпані.

Окрему категорію об'єктів становлять запити та транзакції, які фактично формують реальне навантаження на базу даних. Моніторинг на цьому рівні дозволяє виявляти повільні або неефективні запити, часті блокування, конфлікти транзакцій і аномальні сценарії доступу до даних. Саме аналіз запитів найчастіше дає відповідь на питання, чому база даних працює повільно або нестабільно.

Наступним важливим об'єктом є структури даних — таблиці, індекси та схеми. Їхній стан безпосередньо впливає на швидкість виконання запитів і ефективність використання ресурсів. Розростання таблиць, фрагментація індексів або некоректна структура схем можуть призводити до поступового погіршення продуктивності, яке не завжди помітне без цілеспрямованого моніторингу.

У розподілених середовищах критичними об'єктами стають репліки та вузли кластерів баз даних. Кожен вузол має власний стан, навантаження та показники доступності, а порушення синхронізації між ними може призвести до втрати актуальності даних або проблем під час перемикання у разі відмови. Тому моніторинг повинен охоплювати як окремі вузли, так і їхню взаємодію в межах кластера.

Не менш важливими об'єктами моніторингу є бекапи та задачі обслуговування бази даних. Регулярне резервне копіювання, перевірка цілісності даних, оновлення статистик і обслуговування індексів часто виконуються у фоновому режимі, але можуть істотно впливати на продуктивність. Відсутність контролю за цими процесами створює ризик виявлення проблем лише під час аварійного відновлення.

Окрему категорію становлять журнали транзакцій та системні логи бази даних. Вони відображають внутрішні події СУБД, помилки, попередження та службову інформацію. Аналіз логів дозволяє виявляти проблеми, які ще не проявилися на рівні метрик, а також використовувати їх для детального розслідування інцидентів і аудиту дій.

Таким чином, об'єкти моніторингу у системах баз даних охоплюють як високорівневі компоненти, так і внутрішні механізми СУБД. Комплексний підхід до їхнього контролю є необхідною умовою стабільної, продуктивної та безпечної роботи баз даних у сучасних IT-інфраструктурах.

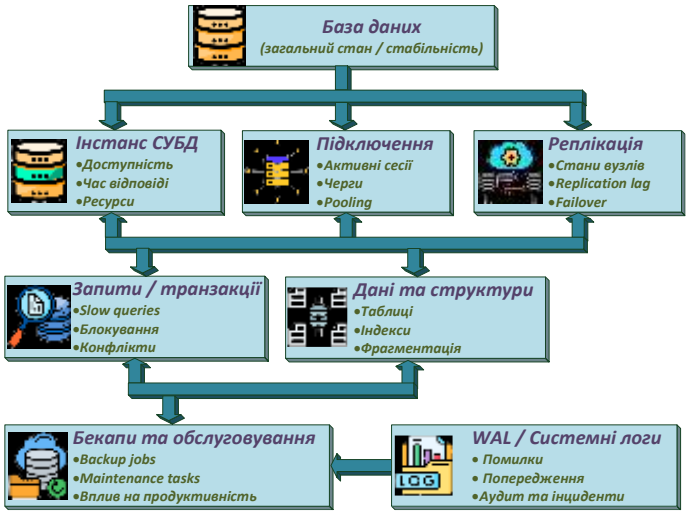

Рис.8.04. Об'єкти моніторингу у системах БД.

Ключові метрики моніторингу баз даних

Метрики є основою будь-якої системи моніторингу баз даних, оскільки саме вони дозволяють кількісно оцінити стан, продуктивність і стабільність роботи СУБД. На відміну від логів, які фіксують окремі події, метрики формують узагальнену картину поведінки бази даних у часі та дають змогу відстежувати тенденції, виявляти аномалії й будувати ефективний алертинг.

Важливо розуміти, що не існує універсального набору метрик, який однаково добре підходить для всіх типів баз даних і сценаріїв використання. Набір ключових показників залежить від архітектури СУБД, типу навантаження, вимог до доступності та ролі бази даних у загальній IT-інфраструктурі. Водночас можна виділити кілька базових категорій метрик, які є спільними для більшості систем і з яких доцільно починати побудову моніторингу.

Першою та найбільш фундаментальною категорією є метрики доступності, оскільки вони відповідають на головне питання: чи доступна база даних для прикладних сервісів у поточний момент.

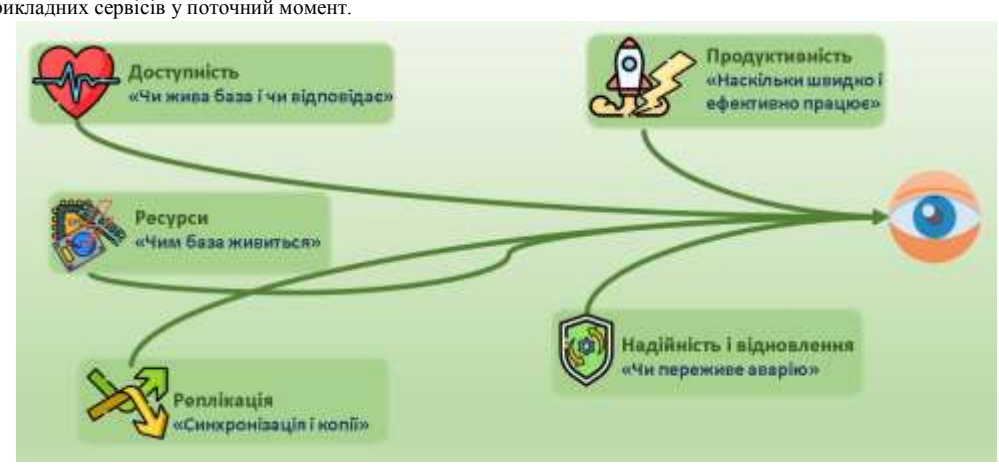

Рис.8.05. Об'єкти моніторингу у системах БД.

- ***Метрики доступності***

Метрики доступності відображають здатність бази даних приймати підключення та обробляти запити. Вони є першою лінією контролю і часто використовуються для побудови критичних алертів, оскільки проблеми на цьому рівні зазвичай одразу впливають на роботу бізнес-додатків.

Однією з базових метрик є статус сервісу бази даних. Він показує, чи запущений інстанс СУБД і чи здатний він відповідати на запити. Втім, простого контролю процесу або порту недостатньо: база даних може бути формально запущеною, але перебувати у некоректному або деградованому стані. Тому на практиці статус сервісу часто перевіряється через виконання простого службового запиту.

Важливим доповненням до статусу є час відповіді на тестові запити. Ця метрика дозволяє оцінити не лише факт доступності, а й реальну швидкість реакції бази даних. Зростання часу відповіді може бути раннім індикатором перевантаження, проблем із ресурсами або внутрішніх блокувань, навіть якщо система ще формально працює.

Ще одним ключовим показником доступності є кількість активних та очікуючих підключень. Активні підключення відображають поточне навантаження на СУБД, тоді як зростання кількості очікуючих або відхилених з'єднань часто сигналізує про вичерпання доступних ресурсів або некоректну конфігурацію пулінгу. Без контролю цієї метрики база даних може стати недоступною навіть за відсутності явних апаратних обмежень.

Таким чином, метрики доступності формують базовий рівень моніторингу бази даних і дозволяють своєчасно виявляти критичні проблеми, що безпосередньо впливають на стабільність роботи сервісів.

- ***Метрики продуктивності***

Метрики продуктивності показують, наскільки ефективно база даних обробляє робоче навантаження. Якщо метрики доступності відповідають на питання «чи працює база даних», то метрики продуктивності дозволяють оцінити, як саме вона працює і чи відповідає її поведінка очікуванням прикладних сервісів і користувачів.

Одними з базових показників продуктивності є QPS (Queries Per Second) та TPS (Transactions Per Second). QPS відображає кількість запитів, які база даних обробляє за одиницю часу, тоді як TPS показує кількість завершених транзакцій. Ці метрики дозволяють оцінити загальний рівень навантаження та динаміку роботи системи. Різкі зміни QPS або TPS можуть свідчити як про зростання бізнес-активності, так і про аномальні сценарії, наприклад, помилки в додатку або некоректну поведінку клієнтів.

Не менш важливою є метрика часу виконання запитів. У практиці моніторингу рідко обмежуються лише середнім значенням, оскільки воно може приховувати проблеми з окремими запитами. Тому широко використовуються перцентилі часу виконання, які показують, як поводиться більшість запитів і чи існують значні відхилення. Зростання часу виконання навіть для невеликої частини запитів може призводити до деградації сервісу та накопичення черг на рівні додатків.

Окрему увагу приділяють повільним запитам (slow queries). Це запити, виконання яких перевищує заданий поріг часу. Аналіз таких запитів є одним із найефективніших інструментів оптимізації продуктивності, оскільки саме вони часто створюють непропорційно велике навантаження на базу даних. Моніторинг повільних запитів дозволяє виявляти неефективні індекси, некоректні SQL-конструкції або проблеми зі структурою даних.

Ще одним важливим аспектом продуктивності є статистика блокувань і взаємних блокувань (lock та deadlock). Блокування є нормальним механізмом забезпечення цілісності даних, однак їх надмірна кількість або тривале утримання може суттєво сповільнювати роботу системи. Deadlock-и, у свою чергу, свідчать про конфлікти між транзакціями та неправильну організацію доступу до даних. Моніторинг цих показників дозволяє вчасно виявляти проблемні сценарії конкурентного доступу та запобігати деградації продуктивності.

Таким чином, метрики продуктивності є ключовим інструментом для розуміння реального навантаження на базу даних і пошуку причин уповільнення роботи, навіть за умови формальної доступності сервісу.

- ***Метрики ресурсів***

Метрики ресурсів відображають, якими обчислювальними та апаратними можливостями користується база даних і наскільки ефективно ці ресурси використовуються. Навіть за оптимальної конфігурації СУБД та коректних запитів нестача або перевантаження ресурсів неминуче призводить до зниження продуктивності. Тому моніторинг ресурсів є необхідним доповненням до аналізу запитів і транзакцій.

Одними з базових показників є використання CPU, оперативної пам'яті та дискової підсистеми (Disk I/O). Високе завантаження процесора може свідчити про складні або неефективні запити, відсутність індексів чи пікове навантаження з боку додатків. Контроль використання оперативної пам'яті дозволяє виявляти ситуації, коли база даних змушена часто звертатися до диска через нестачу кешу. Метрики дискового введення-виведення, зокрема затримки та пропускна здатність, є критично важливими, оскільки саме дискова підсистема часто стає обмежувальним фактором для продуктивності СУБД.

Окреме місце серед ресурсних показників займає cache hit ratio — співвідношення запитів, оброблених з кешу, до загальної кількості звернень до даних. Високе значення цього показника свідчить про ефективне використання оперативної пам'яті та зменшення навантаження на дискову підсистему. Зниження cache hit ratio часто вказує на зростання обсягів даних, зміну характеру запитів або недостатній обсяг пам'яті, що безпосередньо впливає на час відповіді бази даних.

Не менш важливою є метрика використання сховища даних (storage). Контроль обсягу зайнятого простору, темпів його зростання та доступного резерву дозволяє запобігти критичним ситуаціям, пов'язаним із заповненням диска. Переповнення сховища може призвести не лише до зупинки запису даних, а й до порушення роботи СУБД в цілому, включно з відмовами транзакцій та помилками сервісів.

У сукупності метрики ресурсів дозволяють зрозуміти, чи відповідає поточна інфраструктура вимогам навантаження та чи є проблеми, пов'язані не з логікою роботи бази даних, а з обмеженнями середовища, у якому вона працює.

- ***Метрики реплікації***

Метрики реплікації є критично важливими для систем, у яких база даних розгорнута у вигляді реплікованих конфігурацій або кластерів. Реплікація використовується для підвищення доступності, масштабування читання та забезпечення відмовостійкості, однак вона суттєво ускладнює експлуатацію. Саме тому коректний моніторинг реплікаційних процесів є необхідною умовою стабільної роботи таких систем.

Однією з ключових метрик є replication lag — затримка між моментом фіксації змін на основному вузлі та їх застосуванням на репліках. Невелика затримка є допустимою для більшості сценаріїв, проте її зростання може призводити до читання застарілих даних, порушення бізнес-логіки додатків і проблем із узгодженістю. Моніторинг replication lag дозволяє своєчасно виявляти перевантаження реплік, проблеми з мережею або неефективну обробку журналів транзакцій.

Не менш важливим є статус реплік. Він показує, чи знаходяться репліки в актуальному стані, чи отримують вони зміни з основного вузла та чи готові обслуговувати запити. Втрата синхронізації, зупинка реплікаційного потоку або переведення репліки у деградований стан можуть залишатися непоміченими без належного моніторингу, особливо якщо основний вузол продовжує працювати без явних збоїв.

Окрему увагу слід приділяти конфліктам і помилкам синхронізації. У різних СУБД вони можуть виникати з різних причин: паралельні записи, некоректна конфігурація реплікації, помилки мережі або проблеми з дисковою підсистемою. Накопичення таких помилок часто призводить до повної зупинки реплікації або потреби в ручному втручанні адміністратора. Моніторинг помилок і попереджень реплікаційних механізмів дозволяє оперативно реагувати на проблеми ще до того, як вони вплинуть на доступність або цілісність даних.

У підсумку, метрики реплікації дають змогу контролювати не лише факт наявності реплік, а й реальну якість та актуальність даних у розподіленій архітектурі бази даних, що є критично важливим для сучасних високонавантажених систем.

- ***Метрики надійності та відновлення***

Окрім показників доступності, продуктивності та реплікації, для систем баз даних критично важливими є метрики, що характеризують надійність зберігання даних і здатність системи до відновлення після збоїв. Саме ці показники визначають, наскільки база даних готова до аварійних ситуацій, помилок адміністрування або відмов інфраструктури.

Однією з ключових груп таких метрик є стан і результати резервного копіювання. Моніторинг має охоплювати факт успішного виконання бекапів, їхню періодичність, тривалість створення та наявність помилок. Відсутність актуального резервного копіювання часто не проявляється у штатному режимі роботи, але стає критичною у момент інциденту, тому контроль цих метрик є обов'язковим.

Важливе місце займають показники RPO (Recovery Point Objective) та RTO (Recovery Time Objective). RPO визначає максимально допустиму втрату даних у часі, тоді як RTO — допустимий час простою системи до повного відновлення працездатності. Хоча ці значення часто задаються на рівні вимог бізнесу, саме моніторинг дозволяє оцінити, чи відповідає реальна система встановленим цілям.

Окрему увагу слід приділяти процесам відновлення та перевірки цілісності бекапів. Успішно створений резервний файл ще не гарантує можливість коректного відновлення. Тому важливими є метрики, пов'язані з тестовими відновленнями, помилками при відновленні та перевіркою консистентності даних.

Таким чином, метрики надійності та відновлення доповнюють загальну картину стану бази даних і дозволяють оцінювати її не лише з точки зору поточної роботи, а й з позиції готовності до кризових сценаріїв. У зрілих системах моніторингу ці показники є невід'ємною частиною контролю критичних сервісів.

Моніторинг запитів і навантаження

Моніторинг запитів і навантаження є одним із ключових аспектів контролю стану баз даних, оскільки саме на рівні виконання запитів найчастіше проявляються проблеми продуктивності, масштабованості та стабільності роботи системи. Навіть за достатніх апаратних ресурсів і коректно налаштованої СУБД неефективні або некоректно сформовані запити можуть стати основною причиною деградації сервісу.

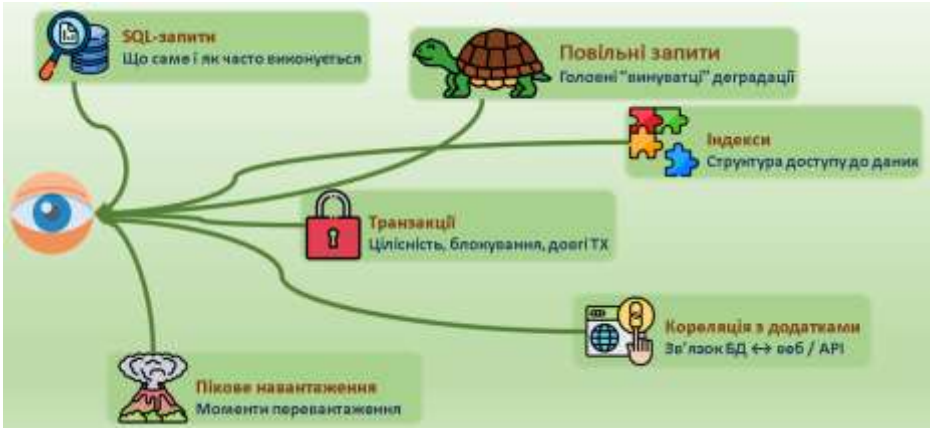

Рис.8.06. Де і як виникає навантаження в БД

Аналіз SQL-запитів дозволяє зрозуміти, як саме база даних використовується прикладними системами. Моніторинг охоплює частоту виконання запитів, їхню тривалість, структуру та споживання ресурсів. Особливу увагу приділяють запитам, які виконуються найчастіше або мають найбільший вплив на загальне навантаження. Саме вони зазвичай формують основну частину затримок і створюють пікові навантаження.

Більшість сучасних СУБД підтримують механізм журналювання повільних запитів (slow query log). У цьому журналі фіксуються запити, час виконання яких перевищує заданий поріг. Аналіз таких записів дозволяє виявляти проблемні ділянки логіки додатків, неоптимальні SQL-конструкції та відсутність необхідних індексів. Важливо, що slow query log є не лише інструментом оптимізації, а й важливим джерелом даних для довготривалого аналізу тенденцій у навантаженні.

Моніторинг бази даних повинен охоплювати не лише самі запити, а й ефективність використання індексів. Неефективні або надлишкові індекси можуть уповільнювати операції запису та збільшувати споживання дискового простору, тоді як відсутність необхідних індексів призводить до повного сканування таблиць і різкого зростання часу виконання запитів. Аналіз планів виконання та статистики використання індексів дозволяє знаходити баланс між швидкістю читання та витратами на обслуговування структури даних.

Транзакції є основою цілісності даних у більшості СУБД, тому їхній стан і поведінка мають безпосередній вплив на стабільність системи. Моніторинг включає контроль тривалості транзакцій, кількості відкритих транзакцій та частоти їх завершення. Довготривалі транзакції можуть блокувати ресурси, викликати затримки та призводити до зростання журналів транзакцій, що робить їх важливим об'єктом спостереження.

Аналіз пікових навантажень дозволяє визначити, у які моменти база даних працює на межі своїх можливостей. Такі піки можуть бути пов'язані з бізнес-подіями, масовими операціями, фоновими задачами або некоректною поведінкою додатків. Моніторинг дає змогу не лише фіксувати сам факт піку, а й аналізувати його причини, що є основою для планування масштабування або оптимізації архітектури.

Окремо слід виділити кореляцію між запитами до бази даних і діями на рівні веб-додатків. Проблеми продуктивності, помітні користувачам, часто мають коріння саме у базі даних, але проявляються у вигляді повільних сторінок або помилок сервісів. Поєднання даних моніторингу БД з логами та метриками додатків дозволяє будувати цілісну картину і швидше знаходити першопричини інцидентів.

Моніторинг реплікацій та кластерів баз даних

Реплікація та кластеризація баз даних широко застосовуються для підвищення доступності, масштабованості та відмовостійкості IT-систем. Водночас такі архітектури суттєво ускладнюють експлуатацію, оскільки стабільна робота залежить не лише від стану окремих інстансів, а й від коректної взаємодії між ними. Саме тому моніторинг реплікацій і кластерів є критично важливим елементом контролю баз даних.

У класичних схемах Primary / Replica один вузол обробляє операції запису, а інші отримують копії змін і зазвичай використовуються для читання або резервування. Моніторинг у таких системах зосереджується на доступності основного вузла, коректності роботи реплік і своєчасності передачі змін. Втрата або деградація primary-вузла без належного контролю може призвести до простою сервісів або втрати даних.

У multi-master архітектурах декілька вузлів можуть одночасно приймати операції запису. Такий підхід підвищує доступність і зменшує залежність від одного інстансу, але водночас створює додаткові ризики конфліктів даних. Моніторинг у цьому випадку має охоплювати стан кожного вузла, механізми вирішення конфліктів та узгодженість даних між усіма учасниками кластера.

Однією з ключових метрик у реплікованих системах є replication lag — затримка між виконанням змін на основному вузлі та їх появою на репліках. Зростання цієї затримки може призводити до застарілих даних на репліках і некоректної роботи додатків, які використовують їх для читання. Моніторинг дозволяє вчасно виявляти такі ситуації та аналізувати причини — від нестачі ресурсів до проблем із мережею або дисковою підсистемою.

Особливо небезпечними є сценарії split-brain, коли вузли кластера втрачають зв'язок між собою і починають працювати як незалежні primary-інстанси. У таких умовах виникає ризик розходження даних і складних для виправлення помилок синхронізації. Моніторинг має фіксувати втрату кворуму, порушення зв'язності між вузлами та аномалії у процесах реплікації, що дозволяє оперативно реагувати на подібні інциденти.

У високодоступних середовищах важливо контролювати процеси failover та switchover — автоматичне або кероване переключення ролей між вузлами. Моніторинг дозволяє відстежувати факт переключення, його тривалість і коректність завершення. Неповний або некоректний failover може призвести до тривалих простоїв або неконсистентного стану системи.

Бази даних часто працюють у зв'язці з зовнішніми HA-рішеннями, такими як кластери високої доступності, балансувальники навантаження або оркестратори. У цьому випадку моніторинг повинен враховувати не лише внутрішній стан СУБД, а й сигнали від HA-систем, оскільки саме вони приймають рішення про переключення ролей і маршрутизацію трафіку. Узгоджений моніторинг дозволяє уникати помилкових спрацьовувань і забезпечує стабільну роботу всієї інфраструктури.

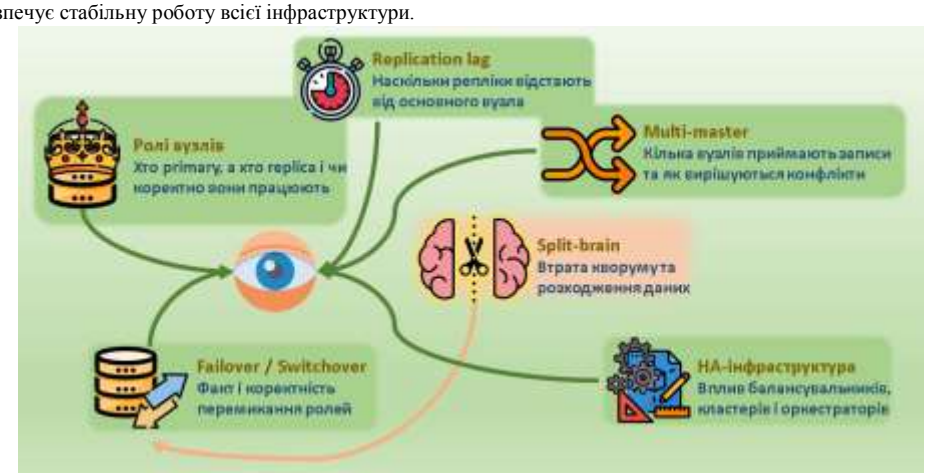

Рис.8.07. Де ламаються репліковані бази даних.

Моніторинг бекапів та відновлення

Резервне копіювання є одним із базових механізмів забезпечення надійності баз даних, однак саме по собі його налаштування ще не гарантує захисту даних. Без постійного моніторингу процесів бекапу та відновлення система залишається вразливою до збоїв, помилок конфігурації та людського фактора. Тому контроль резервного копіювання повинен розглядатися як невід'ємна частина загальної системи моніторингу баз даних.

Рис.8.08. Моніторинг бекапів та відновлення.

Першим і найпростішим завданням є перевірка регулярності створення резервних копій. Моніторинг має фіксувати, чи виконуються бекапи відповідно до заданого розкладу, та сигналізувати про будь-які пропуски. Навіть короткочасна відсутність актуального бекапу може призвести до значних втрат даних у разі інциденту, особливо для транзакційних систем.

Не менш важливим є контроль результатів виконання бекапів. Процес може формально запускатися, але завершуватися з помилками або створювати некоректні резервні копії. Моніторинг повинен аналізувати статуси завершення, повідомлення про помилки та попередження, що дозволяє своєчасно виявляти проблеми з доступом до сховищ, нестачею ресурсів або некоректними налаштуваннями.

Час створення резервної копії також є важливим показником. Зростання тривалості бекапу може свідчити про збільшення обсягу даних, деградацію продуктивності дискової підсистеми або конкуренцію з робочими навантаженнями. Моніторинг цього параметра дозволяє оцінювати вплив бекапів на продуктивність бази даних і своєчасно коригувати розклад або архітектуру резервного копіювання.

Окрему увагу слід приділяти контролю зберігання резервних копій та їх ротації. Моніторинг має відстежувати обсяг зайнятого сховища, дотримання політик зберігання та своєчасне видалення застарілих копій. Порушення цих процесів може призвести як до нестачі дискового простору, так і до зберігання надмірної кількості непотрібних даних.

Наявність резервної копії не гарантує можливість успішного відновлення. Тому важливою складовою є періодична перевірка процесів відновлення. Моніторинг може включати контроль тестових відновлень, перевірку цілісності даних та фіксацію помилок, що виникають під час restore-процесів. Такий підхід дозволяє виявляти проблеми ще до настання реальної аварійної ситуації.

Практика показує, що більшість проблем з бекапами пов'язані не з відсутністю інструментів, а з людськими помилками та некоректними припущеннями. До типових помилок належать відсутність моніторингу, ігнорування попереджень, зберігання бекапів у тій самій інфраструктурі, що й основна база даних, а також відсутність регулярних перевірок відновлення. Системний моніторинг дозволяє мінімізувати ці ризики та підвищити загальну надійність IT-системи.

Логування як об'єкт моніторингу

Логування є одним із фундаментальних механізмів спостереження за роботою IT-систем. На відміну від метрик, які відображають узагальнений стан системи, логи містять детальний опис подій, що відбуваються на рівні операцій, помилок і взаємодії компонентів. Саме тому логування розглядається не лише як допоміжний інструмент налагодження, а як повноцінний об'єкт моніторингу.

У типовій інфраструктурі можна виділити кілька основних категорій логів. Системні логи фіксують події на рівні операційної системи, служби запуску, мережевих компонентів і апаратних ресурсів. Вони дозволяють відстежувати збої, перевантаження та проблеми базового рівня.

Логи додатків відображають внутрішню логіку роботи прикладних систем, включаючи помилки виконання, винятки та бізнес-події. Саме ці логи часто є першим джерелом інформації під час аналізу проблем, з якими стикаються користувачі.

Логи баз даних містять інформацію про помилки СУБД, попередження, виконання запитів, транзакції та внутрішні процеси. Вони відіграють ключову роль у діагностиці проблем продуктивності, доступності та цілісності даних.

Окрему категорію складають аудит-логи, які фіксують дії користувачів, зміну прав доступу, виконання адміністративних операцій і спроби несанкціонованого доступу. Такі логи є важливими з точки зору безпеки, відповідності нормативним вимогам і розслідування інцидентів.

З точки зору обробки логів важливим є поділ на структуровані та неструктуровані. Неструктуровані логи зазвичай мають текстовий формат, орієнтований на людину, і складні для автоматизованого аналізу. Структуровані логи, навпаки, містять чітко визначені поля та значення, що значно спрощує їх пошук, фільтрацію та кореляцію.

Найпоширенішими форматами логів залишаються звичайний текст і JSON. Текстові логи прості у генерації та читанні, але погано підходять для масштабного аналізу. Формат JSON дозволяє зберігати логи у структурованому вигляді, що робить їх зручними для централізованих систем збору та аналітики, а також для інтеграції з системами моніторингу та безпеки.

Логи відіграють ключову роль у пошуку та розслідуванні інцидентів. Вони дозволяють відновити послідовність подій, встановити першопричину проблеми та оцінити її вплив на систему. У поєднанні з метриками та трасуванням логи формують повноцінну основу для спостережуваності, забезпечуючи глибоке розуміння поведінки системи у нормальних і аварійних режимах роботи.

Рис.8.09. Логування як об'єкт моніторингу.

Агрегація та централізований збір логів

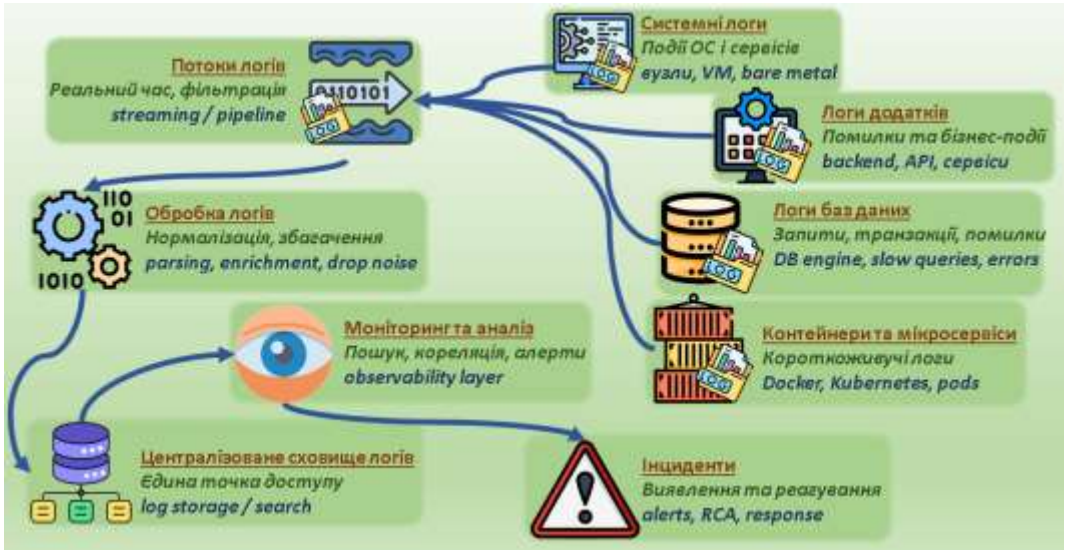

Рис.8.10. Централізоване логування в сучасних IT-системах.

У сучасних IT-інфраструктурах кількість компонентів і сервісів постійно зростає, що робить локальне логування малоефективним і складним в експлуатації. Коли логи зберігаються безпосередньо на окремих серверах або вузлах, їх аналіз потребує ручного доступу, ускладнює кореляцію подій і значно збільшує час реагування на інциденти. Саме тому агрегація та централізований збір логів стали стандартним підходом у побудові систем спостережуваності.

Локальне логування має низку обмежень. Логи можуть бути втрачені у разі збою вузла, перевстановлення системи або проблем зі сховищем. Крім того, аналіз подій у розподілених системах стає складним, оскільки інформація розпорошена між різними серверами, контейнерами та середовищами. Відсутність єдиної точки доступу до логів ускладнює пошук першопричин проблем і підвищує навантаження на команди експлуатації.

Централізоване логування (centralized logging) передбачає збір логів з усіх компонентів інфраструктури в єдине сховище або систему аналізу. Такий підхід дозволяє зберігати логи незалежно від життєвого циклу окремих сервісів, забезпечує швидкий пошук і створює основу для кореляції подій між різними рівнями системи. Централізація логів також є передумовою для автоматизації аналізу, побудови алертів і інтеграції з системами безпеки.

У системах централізованого логування важливо враховувати різноманіття джерел даних. Бази даних генерують логи помилок, повільних запитів, транзакцій та внутрішніх процесів. Веб-додатки створюють логи запитів, помилок виконання та бізнес-подій. Операційні системи постачають системні журнали, що відображають стан сервісів, ресурсів і апаратної частини. У сучасних середовищах значну роль відіграють також контейнери та мікросервіси, логи яких є короткоживучими і потребують негайного збору до централізованого сховища.

Централізований збір логів передбачає роботу з потоками логів, які надходять у режимі реального часу. На цьому етапі відбувається фільтрація, нормалізація та збагачення даних, що дозволяє зменшити обсяг шуму і підготувати логи до подальшого аналізу. Правильна обробка потоків логів є критичною для масштабованості системи та своєчасного виявлення аномалій і інцидентів.

Інструменти систем логування та SIEM

Централізований збір логів створює основу для спостереження за системою, але реальну цінність він набуває лише за наявності інструментів, здатних зберігати, обробляти, аналізувати та корелювати великі обсяги подій. Саме такі можливості надають платформи логування та SIEM-системи, які поєднують технічний моніторинг, аналітику та аспекти інформаційної безпеки.

У цьому розділі розглядаються найпоширеніші інструменти для роботи з логами — від універсальних платформ агрегації до рішень, орієнтованих на великі корпоративні середовища. Особлива увага приділяється їх архітектурі, ролям окремих компонентів і можливостям інтеграції з системами моніторингу.

- ***Elastic Stack (ELK)***

Elastic Stack, також відомий як ELK-стек, є одним із найпопулярніших відкритих рішень для централізованого збору, зберігання та аналізу логів. Його популярність зумовлена гнучкою архітектурою, масштабованістю та можливістю адаптації під різні типи інфраструктур — від невеликих систем до великих розподілених середовищ.

Elasticsearch є ядром Elastic Stack і виконує роль пошукового та аналітичного рушія. Він забезпечує швидке індексування великих обсягів логів і дозволяє виконувати складні пошукові запити майже в реальному часі. Для моніторингу важливо, що Elasticsearch дозволяє ефективно зберігати часові ряди подій і виконувати агрегації, необхідні для аналізу тенденцій та виявлення аномалій.

Logstash відповідає за прийом, обробку та трансформацію логів. Він дозволяє збирати дані з різних джерел, фільтрувати їх, нормалізувати та збагачувати додатковою інформацією перед збереженням у Elasticsearch. У контексті моніторингу Logstash відіграє ключову роль у перетворенні сирих, неструктурованих логів у формат, придатний для подальшого аналізу.

Beats — це набір легковагових агентів, які встановлюються безпосередньо на сервери, контейнери або інші вузли інфраструктури. Вони відповідають за збір логів, метрик та інших подій і передачу їх до Logstash або безпосередньо в Elasticsearch. Завдяки мінімальному споживанню ресурсів Beats добре підходять для масштабних середовищ і динамічних інфраструктур.

Kibana забезпечує візуалізацію та аналітику зібраних логів. Вона дозволяє будувати дашборди, виконувати пошук, аналізувати події та створювати алерти. Для системного та мережевого моніторингу Kibana є інтерфейсом, який перетворює великі масиви логів на зрозумілу та наочну інформацію для адміністраторів і аналітиків.

- ***Graylog***

Graylog є популярною платформою централізованого логування, яка орієнтована на простоту впровадження та зручність щоденної роботи з логами. Вона поєднує можливості збору, зберігання, аналізу та алертингу, що робить її придатною як для середніх, так і для великих інфраструктур. У контексті моніторингу Graylog часто обирають за чітку архітектуру та зрозумілу модель обробки подій.

Архітектура Graylog побудована навколо центрального сервера, який відповідає за прийом і обробку логів, а також сховищ для збереження даних. Graylog приймає логи з різних джерел через інпут-інтерфейси, після чого передає їх на обробку та зберігання. Такий підхід дозволяє масштабувати систему, розділяючи прийом, обробку та зберігання даних, а також інтегрувати Graylog у вже існуючі інфраструктури логування.

Однією з ключових особливостей Graylog є пайплайни обробки логів. Вони дозволяють описувати правила трансформації подій у вигляді послідовних етапів, на яких логи можуть фільтруватися, змінюватися або збагачуватися додатковими полями. Такий механізм значно спрощує нормалізацію логів і підготовку їх до подальшого аналізу. Для моніторингу це означає можливість швидко виділяти важливі події та зменшувати обсяг непотрібної інформації.

Graylog підтримує механізми алертингу, які дозволяють реагувати на визначені події або умови у логах. Алерти можуть спрацьовувати на основі пошукових запитів, перевищення порогових значень або появи конкретних шаблонів у логах. Це робить Graylog корисним інструментом не лише для аналізу після інциденту, а й для оперативного виявлення проблем, таких як помилки додатків, збої сервісів або підозріла активність.

- ***Splunk***

Splunk є потужною комерційною платформою для збору, аналізу та кореляції машинних даних, яка широко використовується у великих корпоративних та критичних середовищах. На відміну від багатьох інших систем логування, Splunk із самого початку орієнтований на роботу з великими обсягами подій у режимі, близькому до реального часу, та на глибоку аналітику з акцентом на безпеку і відповідність вимогам.

В основі Splunk лежить ідея індексування всіх типів машинних даних, включаючи логи, події та метрики. Дані збираються з різних джерел, нормалізуються та зберігаються в індексах, що забезпечує швидкий доступ і пошук. Важливим елементом є уніфікована модель даних, яка дозволяє працювати з різнорідною інформацією в єдиному просторі аналітики.

Однією з ключових переваг Splunk є потужні можливості пошуку та кореляції подій. Платформа дозволяє об’єднувати дані з різних систем, виявляти взаємозв’язки між подіями та будувати складні аналітичні запити. Це особливо важливо під час розслідування інцидентів, коли необхідно відстежити ланцюжок подій між базами даних, додатками, мережевими компонентами та користувацькими діями.

Splunk широко застосовується у великих і географічно розподілених середовищах, де обсяги логів можуть сягати терабайтів на добу. Платформа підтримує масштабування, розподілене зберігання та високодоступні конфігурації, що робить її придатною для критичних систем.

Завдяки розвиненим механізмам аналітики та інтеграції з SIEM-модулями Splunk часто використовується як центральний елемент спостережуваності та безпеки в корпоративних інфраструктурах.

- ***Wazuh як SIEM та система безпекового моніторингу***

Wazuh є відкритою платформою класу SIEM, яка використовує логування як основу для виявлення інцидентів, порушень політик безпеки та аномальної поведінки в IT-інфраструктурі. На відміну від універсальних систем агрегації логів, Wazuh спочатку орієнтований на завдання інформаційної безпеки, поєднуючи моніторинг подій, аналіз логів і контроль стану систем.

Архітектурно Wazuh працює за агентною моделлю, коли спеціальні агенти встановлюються на сервери, робочі станції, бази даних або контейнерні вузли. Вони збирають логи, інформацію про системні події, конфігурації та стан безпеки і передають ці дані до центрального сервера для подальшого аналізу. Такий підхід дозволяє отримувати детальну картину подій безпосередньо з джерела.

Важливою особливістю Wazuh є кореляція подій на основі правил і політик безпеки. Платформа аналізує логи операційних систем, додатків і баз даних, порівнюючи їх із вбудованими правилами, що описують відомі шаблони атак, порушення доступу або небезпечні дії. Це дозволяє переходити від простого збору логів до активного виявлення інцидентів.

Wazuh тісно інтегрується з Elastic Stack, використовуючи Elasticsearch для зберігання даних і Kibana для візуалізації. Завдяки цьому користувач отримує потужні можливості пошуку, побудови дашбордів і аналізу подій у реальному часі. У контексті моніторингу баз даних це дозволяє відстежувати підозрілі запити, помилки доступу, зміни конфігурацій та інші критичні події.

Таким чином, Wazuh займає проміжне місце між класичними системами логування та повноцінними безпековими платформами. Його використання демонструє еволюцію підходів до логування — від пасивного зберігання подій до активного аналізу та реагування, що є важливим аспектом сучасної спостережуваності та захисту IT-інфраструктури.

Таблиця 8.01

Інструменти логування та SIEM: порівняння підходів

Інструмент	Клас системи	Архітектура	Основні компоненти	Що збирає	Основні функції	Методи інтеграції	Типове використання
Elastic Stack (ELK)	Log platform	Модульна, масштабована	Elasticsearch, Logstash, Beats, Kibana	Логи, метрики, події	Пошук, агрегація, аналітика, візуалізація	API, Beats, Logstash input	Observability, моніторинг продуктивності, troubleshooting
Graylog	Log platform	Центральний сервер + сховище	Graylog server, MongoDB, Elasticsearch, Inputs	Логи, події, метрики	Централізація, фільтрація, нормалізація, алерти	Syslog, GELF, Beats	Troubleshooting, середні та великі інфраструктури
Splunk	Log / SIEM	Розподілена, індексована	Indexer, Forwarder, Search Head	Машинні дані, логи, події, метрики	Пошук, кореляція подій, аналітика, дашборди	Forwarder, API, SDK	Enterprise, SOC, security analytics, large-scale log management
Wazuh	SIEM / Security monitoring	Агент-сервер	Wazuh server, агенти, Elasticsearch, Kibana	Логи, події безпеки, конфігурації	Виявлення інцидентів, кореляція правил, алерти	Agents, API, Elastic Stack	Security monitoring, incident detection, compliance, SOC

- ***Зв'язок теоретичного матеріалу з практичним курсом***

Розглянуті інструменти логування та SIEM формують цілісну концептуальну основу для практичної роботи з системами безпекового та операційного моніторингу. Теоретичний матеріал пояснює, чому сучасні IT-інфраструктури потребують централізованого збору логів, кореляції подій і автоматизованого виявлення інцидентів, тоді як лабораторні роботи дозволяють реалізувати ці підходи на практиці.

Розгортання Wazuh Server і підключення агентів ілюструють агентну модель збору подій і принципи централізованого логування. Аналіз системних логів та виявлення інцидентів демонструють роботу правил кореляції та алертингу. Використання Splunk дає змогу побачити альтернативний підхід до зберігання, пошуку та візуалізації логів, а інтеграція Wazuh зі Splunk показує, як різні інструменти можуть доповнювати одне одного в межах єдиної екосистеми моніторингу та безпеки.

Таким чином, практичний курс не просто навчає роботі з конкретними продуктами, а закріплює загальні принципи, розглянуті в лекції, формуючи системне бачення роботи з логами та подіями.

- ***Інтеграція логів з системами моніторингу***

Ізольоване використання логування або метрик значно обмежує можливості аналізу стану системи. Тому важливим етапом розвитку інфраструктури моніторингу є інтеграція систем збору логів із класичними системами моніторингу метрик і доступності. Такий підхід дозволяє поєднати числові показники продуктивності з контекстом подій, зафіксованих у логах.

На практиці це означає кореляцію алертів за метриками (наприклад, зростання часу відповіді чи навантаження на CPU) з відповідними подіями в логах баз даних, операційних систем або додатків. Інтеграція може реалізовуватися через спільні дашборди, передачу подій між системами, використання webhook-ів або API. У результаті оператор або інженер отримує не просто сигнал про проблему, а й інформацію про її можливу причину.

Такий підхід є ключовим для побудови повноцінної observability-платформи, де метрики відповідають на питання «що відбувається», а логи — «чому це відбувається». Саме поєднання цих джерел даних дозволяє швидше локалізувати інциденти, зменшити час реагування та підвищити загальну надійність і безпеку IT-систем.

Кореляція метрик, логів та подій

У сучасних IT-інфраструктурах жодне джерело даних саме по собі не дає повної відповіді на питання про стан системи. Метрики показують кількісні характеристики роботи, логи фіксують події та контекст, а події й алерти сигналізують про відхилення від норми. Лише їх поєднання дозволяє перейти від простого спостереження до глибокого розуміння причин проблем. Саме тому кореляція метрик, логів і подій є центральним елементом підходу observability.

Моніторинг баз даних традиційно зосереджується на метриках продуктивності, доступності та використання ресурсів. Проте самі по собі ці показники часто не пояснюють, *чому* виникла проблема. Логування доповнює метрики інформацією про виконувані запити, помилки, винятки та внутрішні події СУБД. Кореляція цих даних дозволяє пов'язати зміну числових показників із конкретними діями або збоями, що відбуваються всередині системи.

Наприклад, зростання часу відповіді бази даних стає значно зрозумілішим, якщо одночасно в логах з'являються записи про повільні запити, блокування або помилки доступу до дискової підсистеми. У такому випадку метрики вказують на симптом, а логи — на потенційну причину.

Типовим прикладом є ситуація, коли фіксується різке зростання навантаження на CPU сервера бази даних. Сама по собі ця метрика не пояснює джерело проблеми. Однак аналіз логів або статистики запитів може показати появу повільних або неефективних SQL-запитів, зміну планів виконання або запуск ресурсоємних аналітичних операцій. Кореляція цих даних дозволяє пов'язати пікове навантаження з конкретними діями в системі.

Інший поширений сценарій — деградація сервісу з точки зору користувачів. Метрики доступності та часу відповіді сигналізують про проблему, але саме логи додатків або бази даних можуть містити повідомлення про помилки з'єднання, тайм-аути чи відмови реплік. Поєднання цих джерел дозволяє швидко зрозуміти, чи проблема має інфраструктурний, прикладний або логічний характер.

Root cause analysis — пошук першопричини інциденту — практично неможливий без використання логів. Метрики зазвичай показують лише наслідки проблеми, тоді як логи фіксують послідовність подій, що до неї призвела. Аналіз часових міток, повідомлень про помилки та контекст виконання операцій дозволяє відновити хронологію інциденту.

У поєднанні з метриками це дає змогу не лише усунути поточну проблему, а й зрозуміти, які зміни в конфігурації, коді або навантаженні стали її причиною. Такий підхід є основою для підвищення стабільності систем і запобігання повторенню інцидентів у майбутньому.

APM-системи (Application Performance Monitoring) відіграють важливу роль у кореляції подій на рівні додатків, баз даних і інфраструктури. Вони дозволяють пов'язати дії конкретного користувача або запит веб-додатку з транзакціями в базі даних, використанням ресурсів і записами в логах. Така наскрізна видимість забезпечує розуміння поведінки системи від зовнішнього запиту до внутрішніх компонентів.

У результаті кореляція метрик, логів і подій перетворює моніторинг із набору розрізнених інструментів на цілісну систему спостережуваності. Вона дозволяє швидше реагувати на інциденти, точніше визначати їх причини та приймати обгрунтовані рішення щодо оптимізації, масштабування й підвищення надійності IT-інфраструктури.

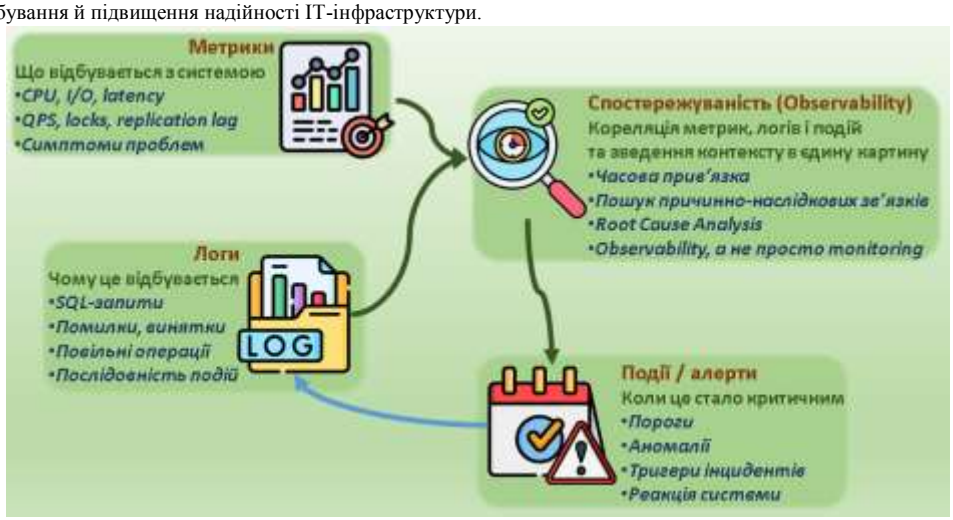

Рис.8.11. Кореляція метрик, логів та подій.

Безпека моніторингу баз даних і логів

Рис.8.12. Безпека моніторингу баз даних і логів.

Моніторинг баз даних і систем логування значно підвищує прозорість роботи IT-інфраструктури, але водночас створює нові ризики з точки зору безпеки. Метрики, логи та аналітичні дані часто містять чутливу інформацію, доступ до якої повинен бути жорстко контрольований. Тому безпека моніторингу є невід'ємною складовою загальної стратегії захисту інформаційних систем.

Одним із базових принципів безпеки є принцип мінімально необхідних привілеїв. Не всі користувачі, які мають доступ до систем моніторингу, повинні бачити всі метрики та всі логи. Адміністратори баз даних, розробники, спеціалісти з безпеки та оператори можуть мати різні ролі та рівні доступу.

Системи моніторингу та логування повинні підтримувати рольову модель доступу, яка дозволяє розмежовувати права на перегляд, пошук, експорт і зміну конфігурацій. Це знижує ризик витоку інформації та несанкціонованих дій з боку внутрішніх користувачів.

Логи баз даних і додатків часто містять персональні дані, облікові записи, токени доступу або фрагменти SQL-запитів. Зберігання таких даних у відкритому вигляді створює серйозні ризики безпеки та проблеми з відповідністю регуляторним вимогам.

Тому важливим елементом безпечного логування є маскування або анонімізація чутливих даних ще на етапі збору або обробки логів. Це дозволяє зберігати діагностичну цінність логів, не розкриваючи критично важливу інформацію навіть у разі компрометації системи логування.

Аудит доступу є ключовим механізмом контролю дій користувачів і сервісів у базах даних. Він дозволяє фіксувати, хто, коли і до яких об'єктів БД звертався, а також які операції виконувалися. Такі аудит-логи мають особливу цінність у розслідуванні інцидентів і забезпеченні відповідності стандартам безпеки.

Моніторинг аудиту доступу дає змогу виявляти аномальні дії, наприклад спроби доступу поза робочим часом, масові вибірки даних або незвичні операції з привілейованих облікових записів. У поєднанні з іншими метриками це формує основу для поведінкового аналізу.

Системні та прикладні логи є одним із головних джерел інформації для виявлення атак і зловмисної активності. Помилки автентифікації, спроби ескалації привілеїв, підозрілі запити до бази даних або аномальні шаблони доступу можуть бути зафіксовані саме в логах.

Регулярний аналіз таких подій дозволяє виявляти як зовнішні атаки, так і внутрішні загрози. У цьому контексті моніторинг логів перестає бути лише інструментом експлуатації та стає важливою складовою системи інформаційної безпеки.

Інтеграція логів баз даних і систем моніторингу з SIEM-платформами дозволяє підняти рівень безпекового контролю на новий рівень. SIEM-системи виконують кореляцію подій із різних джерел, застосовують правила виявлення загроз і формують централізовану картину безпеки.

У такій архітектурі логи БД стають частиною загального потоку безпекових подій поряд із логами операційних систем, мережевих пристроїв і додатків. Це дозволяє оперативно реагувати на інциденти, автоматизувати сповіщення та зменшувати час виявлення атак.

Побудова системи моніторингу БД та логування

Побудова ефективної системи моніторингу баз даних і логування потребує комплексного підходу, який враховує масштаб інфраструктури, критичність сервісів, вимоги до безпеки та експлуатаційні особливості. Важливо не лише зібрати метрики й логи, а й організувати їх так, щоб система залишалася керованою, масштабованою та корисною в реальних сценаріях.

Рис.8.13. Побудова системи моніторингу БД та логування.

Архітектура системи моніторингу зазвичай будується за принципом централізації збору даних із розподілених джерел. Бази даних, сервери, контейнери та додатки виступають джерелами метрик і логів, які передаються до централізованих систем зберігання й аналізу. Важливо враховувати відмовостійкість самих компонентів моніторингу, оскільки їхня недоступність у критичний момент знижує цінність усієї системи спостережуваності.

Добра архітектура передбачає розділення шарів збору, обробки, зберігання та візуалізації, що спрощує масштабування та супровід. Такий підхід дозволяє адаптувати систему під різні типи баз даних і навантажень без повної перебудови інфраструктури.

Одним із ключових рішень є вибір між агентним і агентless підходами. Агентний підхід передбачає встановлення спеціальних компонентів на серверах баз даних або операційних системах. Це забезпечує глибокий доступ до метрик і логів, але збільшує складність адміністрування та вимоги до безпеки.

Агентless підхід базується на опитуванні сервісів через стандартні інтерфейси та протоколи. Він спрощує розгортання, але може обмежувати глибину моніторингу. На практиці часто використовується комбінований підхід, коли для критичних баз даних застосовуються агенти, а для менш важливих компонентів — агентless механізми.

Системи логування мають бути готові до постійного зростання обсягів даних. Масштабування досягається за рахунок горизонтального розширення компонентів збору, обробки та зберігання логів. Важливо також враховувати пікові навантаження, які можуть виникати під час інцидентів або масових помилок у додатках.

Грамотне масштабування дозволяє уникнути втрати логів і зберігати стабільність системи навіть за різкого зростання кількості подій. Це особливо критично для баз даних і систем безпеки, де втрата логів може ускладнити аналіз інцидентів.

Окрему увагу слід приділяти політикам зберігання та ротації логів. Логи мають зберігатися достатньо довго для аналізу інцидентів, аудиту та відповідності вимогам регуляторів, але не безконтрольно. Ротація дозволяє обмежувати обсяг сховищ і підтримувати стабільну продуктивність систем логування.

Для баз даних важливо враховувати різну цінність логів: транзакційні та аудит-логи зазвичай зберігаються довше, ніж діагностичні або відладочні записи. Такий підхід дозволяє оптимізувати витрати на зберігання без втрати критично важливої інформації.

Алертинг є завершальним, але надзвичайно важливим елементом системи моніторингу. Правильно налаштовані пороги дозволяють виявляти проблеми на ранніх етапах і реагувати до того, як вони вплинуть на користувачів. Водночас надмірна кількість алертів призводить до перевантаження операторів і зниження ефективності реагування.

Тому при побудові алертингу важливо поєднувати метрики баз даних, події з логів і контекст сервісів. Це дозволяє формувати більш точні сповіщення, які відображають реальний стан системи, а не окремі локальні відхилення.

Типові проблеми та помилки

Навіть за наявності сучасних інструментів моніторингу та логування системи баз даних часто експлуатуються з помилками, які знижують ефективність спостережуваності. Більшість із них пов'язані не з відсутністю технологій, а з неправильними підходами до їх використання. Усвідомлення типових проблем дозволяє будувати більш надійні та керовані системи.

Однією з найпоширеніших помилок є зосередження виключно на доступності бази даних. Факт того, що сервіс відповідає на запити, ще не означає його коректну роботу. База даних може бути формально доступною, але при цьому працювати з великими затримками або не справлятися з навантаженням.

Ігнорування метрик продуктивності призводить до ситуацій, коли проблеми стають помітними лише після скарг користувачів. Ефективний моніторинг повинен поєднувати контроль доступності з аналізом часу відповіді, транзакційної активності та використання ресурсів.

Ще однією типовою проблемою є неконтрольоване накопичення логів. Надмірна деталізація логування без чіткої мети призводить до зростання витрат на зберігання та ускладнює аналіз подій. У результаті важливі повідомлення губляться серед другорядної інформації.

Раціональний підхід передбачає баланс між повнотою логів і їх практичною цінністю. Логи повинні бути достатньо інформативними для діагностики та безпеки, але не перевантажувати систему та операторів.

У реплікованих і кластерних базах даних часто недооцінюється важливість моніторингу реплікаційних процесів. Відсутність алертів на затримки реплікації або помилки синхронізації може призвести до непомітної деградації системи та втрати актуальності даних.

Такі проблеми особливо небезпечні в системах, де репліки використовуються для читання або як резервні вузли. Моніторинг реплікації повинен бути обов'язковою частиною загальної системи спостережуваності.

Наявність налаштованих резервних копій часто сприймається як достатня умова безпеки даних. Проте відсутність моніторингу процесів бекапування і перевірок можливості відновлення робить такі бекапи формальністю.

Типовою помилкою є відсутність алертів на збої резервного копіювання або неконтрольоване зростання часу виконання бекапів. Моніторинг бекапів має бути таким же обов'язковим, як і моніторинг основної роботи бази даних.

Ізольований моніторинг баз даних без урахування стану додатків, мережі та інфраструктури значно обмежує можливості аналізу. Проблеми часто виникають на стику компонентів, і без кореляції даних з різних шарів знайти їхню причину стає складно.

Ефективна система моніторингу повинна інтегрувати метрики й логи баз даних з інформацією про роботу веб-додатків, контейнерів, віртуальних середовищ і мережі. Лише такий підхід дозволяє отримати цілісне уявлення про стан системи та швидко реагувати на інциденти.

Рис.8.14. Типові проблеми та помилки моніторингу БД.

Практичні сценарії та кейси

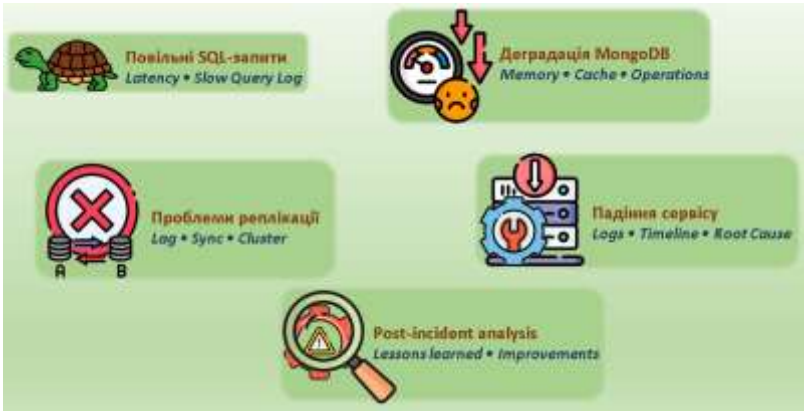

Рис.8.15. Кілька практичних сценаріїв та кейсів

Практичні сценарії дозволяють побачити, як теоретичні принципи моніторингу баз даних і логування працюють у реальних умовах. У повсякденній експлуатації проблеми рідко проявляються ізольовано, тому аналіз завжди базується на поєднанні метрик, логів і подій. Розглянемо типові кейси, з якими стикаються фахівці з підтримки та адміністрування.

Одним із найпоширеніших сценаріїв є поступове або раптове зростання часу відповіді бази даних. Моніторинг продуктивності дозволяє зафіксувати збільшення середнього або перцентильного часу виконання запитів. Далі в роботу вступає аналіз slow query log або статистики запитів, які показують, які саме операції споживають найбільше ресурсів.

Кореляція цих даних із метриками CPU, дискових операцій і кешу дозволяє визначити, чи проблема пов'язана з неефективними запитами, відсутністю індексів або зміною характеру навантаження. Такий підхід є типовим для оптимізації продуктивності транзакційних систем.

Наступний сценарій, що ми розглянемо - аналіз деградації MongoDB. У документно-орієнтованих базах даних, таких як MongoDB, деградація часто проявляється у вигляді збільшення затримок операцій або нестабільної роботи під навантаженням. Моніторинг показників використання пам'яті, роботи кешу та кількості активних операцій дозволяє виявити аномалії.

Логи MongoDB доповнюють картину повідомленнями про повільні операції, блокування або проблеми з реплікацією. Аналіз цих даних у сукупності дозволяє зрозуміти, чи деградація пов'язана зі структурою даних, розміром документів або конфігурацією кластера.

Тепер про пошук причин падіння сервісу. У випадках, коли сервіс стає недоступним або нестабільним, логи часто є єдиним джерелом інформації про причини інциденту. Аналіз системних, прикладних і базових логів дозволяє відновити послідовність подій, що передували збою.

Кореляція часових міток у логах із метриками доступності та навантаження дає змогу відокремити першопричину від вторинних наслідків. Такий підхід значно скорочує час діагностики та дозволяє уникнути повторення інциденту.

Не можна не згадати про реплікаційні проблеми. Проблеми з реплікацією часто не призводять до негайної відмови сервісу, але створюють приховані ризики. Моніторинг реплікаційних затримок і статусу реплік дозволяє вчасно зафіксувати відхилення.

Логи бази даних і кластерних компонентів допомагають з'ясувати причини проблем, наприклад помилки мережі, конфлікти записів або перевантаження вузлів. Практичне реагування включає не лише усунення поточного збою, а й аналіз умов, за яких він виник.

Після усунення інциденту важливо провести постінцидентний аналіз, який базується переважно на логах. Вони дозволяють відновити повну картину подій, оцінити ефективність реагування та виявити слабкі місця в системі моніторингу.

Такий аналіз є основою для покращення алертів, уточнення порогів і вдосконалення архітектури моніторингу. У результаті кожен інцидент стає джерелом знань і сприяє підвищенню загальної надійності IT-інфраструктури.

Місце моніторингу баз даних і логування в загальній системі спостережуваності

Моніторинг баз даних і систем логування займає центральне місце в сучасній системі спостережуваності, оскільки саме ці компоненти акумулюють ключову інформацію про реальну поведінку IT-систем. Бази даних є ядром більшості сервісів, а логи — хронікою всіх подій, що відбуваються в інфраструктурі. Їх поєднання дозволяє перейти від реактивного реагування до проактивного управління стабільністю та безпекою.

У межах концепції observability моніторинг баз даних і логування тісно пов'язані з усіма трьома основними складовими: метриками, логами та трасуванням. Метрики баз даних показують кількісні характеристики роботи, логи забезпечують контекст і деталізацію подій, а трасування дозволяє пов'язати операції бази даних із запитами додатків і діями користувачів.

Рис.8.16. Місце моніторингу БД і логування в системі моніторингу

Разом ці компоненти формують повний цикл спостережуваності, у якому проблеми не лише фіксуються, а й пояснюються. Це особливо важливо для складних розподілених систем, де без кореляції даних з різних джерел неможливо швидко знайти першопричину інциденту.

Моніторинг баз даних тісно пов'язаний із контролем стану інфраструктури. Продуктивність СУБД безпосередньо залежить від ресурсів серверів, мережі та систем зберігання. Інфраструктурні метрики дозволяють зрозуміти, чи проблеми бази даних викликані дефіцитом ресурсів, апаратними збоями або мережевими затримками.

У такій взаємодії бази даних перестають розглядатися ізольовано і стають частиною єдиної екосистеми, де кожен рівень впливає на загальний результат.

Веб-моніторинг відображає стан сервісів з точки зору кінцевих користувачів. Затримки, помилки або недоступність веб-додатків часто мають свої корені саме в роботі баз даних. Кореляція показників веб-моніторингу з метриками та логами БД дозволяє швидко пов'язати погіршення користувацького досвіду з внутрішніми технічними причинами.

Такий підхід допомагає фокусуватися не лише на технічних показниках, а й на реальному впливі проблем на бізнес-процеси.

APM-системи (Application Performance Monitoring) відіграють роль «містка» між додатками та базами даних. Вони дозволяють простежити шлях запиту від користувацького інтерфейсу до конкретної операції в СУБД. Це особливо цінно для аналізу складних транзакцій, де час виконання формується сукупністю дій на різних рівнях системи.

У взаємодії з APM моніторинг баз даних стає частиною наскрізного аналізу продуктивності, що значно спрощує оптимізацію та пошук вузьких місць.

Логування та моніторинг баз даних є важливим джерелом даних для SIEM-систем. Аудит-логи, помилки доступу, аномальні запити та підозрілі дії користувачів дозволяють виявляти загрози на ранніх етапах. У межах SIEM ці події корелюються з інформацією з інших джерел, формуючи цілісну картину безпеки.

Таким чином, моніторинг БД і логування виконують не лише експлуатаційну, а й безпекову функцію, що особливо важливо для критичних і регульованих середовищ.

Моніторинг баз даних і логування логічно завершують цей модуль, об'єднуючи метрики, події та контекст у єдину систему спостережуваності. Саме на цьому рівні стає очевидно, як технічні показники трансформуються у реальний вплив на сервіси, користувачів і бізнес у цілому.