УДК 004.658.2

DOI: 10.18372/2073-4751.75.18018

Перекрест А.Л., д.т.н.,
orcid.org/0000-0002-7728-9020,
Бахарєв В.С., д.т.н.,
orcid.org/0000-0001-9312-654X,
Вадурін К.О.,
orcid.org/0000-0001-7781-5783,
Дерієнко А.І.,
Іващенко А.В.,
Шкарупа С.А.

РОЗРОБКА БАЗИ ДАНИХ ДЛЯ ЗБЕРІГАННЯ ПОКАЗНИКІВ СТАНУ АТМОСФЕРНОГО ПОВІТРЯ З ДОСЛІДНИХ СТАНЦІЙ КОМУНАЛЬНОГО ПІДПРИЄМСТВА

Кременчуцький національний університет імені Михайла Остроградського

pksg13@gmail.com,
v.s.baharev@gmail.com,
kir3337@gmail.com,
andrey.deriyenko@gmail.com,
a.ivascheko2020@gmail.com,
sergii.shkarupa@gmail.com

Вступ

На сьогоднішній день у світі гостро стоїть проблема збереження стабільної чистоти атмосферного повітря, зважаючи на високий рівень урбанізації, зростання числа підприємств з викидами в атмосферу та збільшення кількості особистих транспортних засобів. Україна також стикається з серйозними незворотними змінами, викликаними війною та руйнуванням Каховської ГЕС. Результатом цих змін є обміління Каховського водосховища, відселення населення з зруйнованих територій, переміщення підприємств, зменшується кількість каналів для зрошення полів у південній частині України. Все це збільшує навантаження на організації, що займаються екологічними дослідженнями, оскільки потрібно аналізувати стан атмосферного повітря в умовах релокації виробничих підприємств та населення з інших областей. Ці зміни впливають на щільність та інтенсивність викидів, які були зафіксовані та прогнозувалися статистично ще до початку війни та екологічної катастрофи на Каховській ГЕС.

Також на території України діє велика кількість автоматичних станцій та постів екологічного моніторингу, що належать до різних рівнів та суб'єктів моніторингу. Значна частина таких систем чи постів під'єднані до мережі Інтернет та зберігають екологічну інформацію у власних базах даних.

Для проведення детальних досліджень поточного екологічного стану атмосферного повітря в межах Кременчуччини та інших зонах досліджень, накопичення знань про тренди змін вимірюваних показників забрудненості повітря та прогнозування майбутніх змін екології, актуальною задачею є створення сукупної бази даних для автоматичного збереження показів як з постів та станцій досліджень, що належать комунальному підприємству-замовнику, так і онлайн-ресурсів партнерів.

Аналіз джерел та постановка проблеми

Дослідження пов'язані з екологічним станом атмосферного повітря Кременчуччини, використовуючи методи комп'ютерного аналізу даних, описані у великій кількості наукових праць.

У роботі [1] запропоновано розширити перелік вихідних умов для епізодичного та оперативного контролю за станом

атмосферного повітря в місті з використанням ПМЕЛ (постійних моніторингових екологічних лабораторій). Такими умовами можуть бути результати вимірювань станцій громадського моніторингу з урахуванням вимог діючих нормативних документів. Наприклад, станція громадської мережі моніторингу може фіксувати протягом трьох поспіль годин концентрацію *NO2* у значеннях, що перевищують небезпечні пороги. Інформаційно-аналітична система муніципального моніторингу формує сповіщення для відділу оперативного контролю, а також ініціює відповідні дії за певною схемою. Таке рішення дозволить підтверджувати дані індикативних станцій за допомогою повіреного обладнання муніципальної лабораторії, що, в свою чергу, збільшить можливості виявлення реальних забруднювачів атмосферного повітря в місті.

Але у дослідженні не наведено конкретних способів реалізації запропонованої пропозиції. Також реалізація інформаційно-аналітичної системи потребує бази даних, що дозволить формувати звіти за різні проміжки часу, щоб виявити що спричинило перевищення рівня показника.

У дослідженні [2] подано результати теоретичних та практичних досліджень, що стосуються розробки концептуального підходу до формування звітів за результатами комплексного післяпроєктного екологічного моніторингу як частини процедури оцінки впливу на довкілля техногенних об'єктів. Запропоновано забезпечувати комплексність проведених моніторингових досліджень шляхом інтеграції різних методів спостережень за станом основних компонентів довкілля, на які може впливати імовірний негативний вплив. Науковий аспект аналізу результатів комплексу натурних спостережень забезпечується застосуванням методів порівняльного аналізу та верифікації результатів хімічного та фізико-хімічного кількісного аналізу за допомогою методів біомоніторингу.

У даному дослідженні також не наведено практичного прикладу реалізації доповнюваної екологічної бази даних за інформацією з якої формуються звіти, що було б корисним для реалізації подібних проєктів у віддалених зонах екологічних досліджень.

У роботі [3] акцентовано увагу на необхідності об'єктивного інформування населення через засоби масової інформації про рівень радіаційного фону в місті Кременчук у безперервному режимі. Для аналізу даних, у роботі використано програмне забезпечення *Microsoft Power BI*, в якому реалізовано завантаження даних з архівів за трьома областями, виділення певних станцій у кожній з областей, перетворення даних часу в формат дата-час, агрегація за годинними, добовими, місячними, квартальними та річними показниками, відображення екологічних параметрів за окремими станціями, містами та областями.

Але масиви даних, при розширенні системи інформування та зберіганні ретроспективних даних, потребують спеціалізованого рішення базованого на системі керування базами даних для стабільної роботи з ними, оскільки *Microsoft Power BI* має обмеження обсягу даних, який можна завантажити та обробити, та кількість джерел даних.

Отже, екологічні дослідження стану атмосферного повітря у Кременчуці недостатньо уваги приділяють питанням зберігання та роботи з великими обсягами даних, що зумовлює необхідність проведення досліджень в цьому напрямі.

Мета та задачі дослідження

Об'єктом дослідження є процес обробки та збереження даних від автоматичних станцій та постів спостереження.

Предмет дослідження повністю збігається з метою дослідження.

Метою цього дослідження є створення та впровадження структурованої бази даних показів з автоматичних станцій та постів спостереження, що дозволить зберегти великі об'єми даних про стан повітря в одному просторі, швидко отримувати доступ до збережених показів, та

стане основою для створення служби з автоматизованого формування звітних відомостей, а також підтримуватиме розширення у вигляді додання нових станцій чи даних з онлайн-ресурсів.

Основні задачі, які необхідно виконати у даному дослідженні наступні:

- проаналізувати існуючі моделі баз даних. Обрати вид бази даних що відповідатиме вимогам поставленим до зберігання та оновлення екологічних даних;
- проаналізувати системи керування базами даних. Обрати систему керування базами даних що забезпечить контроль за надлишковістю даних та підтримуватиме її цілісність;
- спроєктувати структуру бази даних згідно з системою класифікації та кодування, та реєстру постів спостереження;
- реалізувати базу даних за допомогою обраної системи керування базами даних.

Наукова новизна дослідження

Наукова новизна дослідження полягає у наступному:

- уперше розроблено модель бази даних, для зберігання екологічних показників, що містить такі таблиці: *migrations*; *failed_jobs*; *users*; *password_reset_tokens*; *personal_access_tokens*; *station*1756; *station*1748; *station*1753; *data_eco_city*_1755; *station_T*3950716, *station_V*0440346; *station_T*3950713; *split_measurement_ua*171, та призначена для обслуговування локального екологічного сервісу;
- уперше розроблено програмну частину моделі бази даних для зберігання екологічних показників, виходячи з конфігурації апаратного забезпечення КП «НДЦ», основою якої є реляційна, однокористувацька централізована *SQL* модель що працює на локальних ресурсах сервера, зберігається у дисковому просторі та призначена для аналітики екологічних даних. Мовою програмування для реалізації серверного функціоналу та роботи з базою даних обрано *PHP*.

Матеріали дослідження

Вихідні екологічні показники для заповнення бази даних у реальному часі отримано з трьох станцій *Vaisala* з сенсорами *AQT*530 та *WXT*530.

Станції *Vaisala* з серійними номерами *T*3950713 та *T*3950716 містять обидва сенсора. Станція з серійним номером *V*0440346 має тільки один сенсор *AQT*530.

*AQT*530 – газоаналізатор, що вимірює такий набір показів: атмосферний тиск; температура повітря; монооксид вуглецю *CO*; сірководень H_2S; діоксид азоту NO_2; мікроскопічні тверді частинки *PM*10, *PM*1, *PM*2.5; відносну вологість повітря; діоксид сірки SO_2.

*WXT*530 – метеостанція, що вимірює такий набір показів: швидкість вітру; напрям вітру; інтенсивність осадів; відносну вологість; температуру повітря; накопичення опадів; атмосферний тиск; максимальна швидкість вітру.

Також базу даних заповнено ретроспективними даними від чотирьох станцій *Air Fresh Max* з особистого кабінету дослідника в *EcoCity*, що надходять у двох форматах: від станції *station*1756 сирі дані отримуються кожну хвилину, а від станцій *station*1748, *station*1753, *station*1755 надходять усередненні дані за кожні 20 хвилин. Номери станцій вказують на конкретну станцію, наприклад, код 1748, 17 відповідає номеру області досліджень, а 48 – номеру станції.

Air Fresh Max – це автоматична станція, що дозволяє здійснювати контроль та реєстрацію стану повітря навколишнього середовища, а саме показників: температури; вологості; тиску; концентрації пилу *PM*2.5 і *PM*10; озону O_3; аміаку NH_3; діоксиду азоту NO_2; монооксиду вуглецю *CO*; вуглекислого газу CO_2; в режимі реального часу.

Методи дослідження

Для досягнення поставленої мети було використано ряд методів дослідження:

- проведено дослідження наукових публікацій, статей, джерел, документів та інформаційних онлайн-ресурсів, що

стосуються баз даних, систем керування базами даних, методів зберігання та аналізу екологічних даних;

- вивчено потреби комунального підприємства та поставлені вимоги до бази даних, визначено функціональні та нефункціональні вимоги до системи;
- аналіз різних типів моделей баз даних, таких як реляційні, ієрархічні, мережеві, *NoSQL* тощо, і вибір найбільш підходящої моделі для проєкту;
- визначення мов програмування для реалізації серверного функціоналу та вибір системи керування базами даних для роботи з великими обсягами даних;
- розробка схеми бази даних, визначення таблиць, полів та їх зв'язків, визначення індексів та унікальних ключів;
- проведення послідовних міграцій бази даних для створення необхідних таблиць та забезпечення актуальності структури.

Визначення та поняття баз даних

База даних – це структурована колекція даних [4], яка зберігається і управляється з допомогою спеціального програмного забезпечення, відомого як система керування базами даних (СКБД). База даних дозволяє організовувати, зберігати, оновлювати та аналізувати великий обсяг даних для забезпечення ефективного доступу до них з допомогою різних додатків і користувачів.

Основні поняття пов'язані з базами даних такі.

Таблиця – це основна структура даних в реляційній моделі баз даних. Вона складається з рядків (записів) і стовпців (полів), де кожен рядок представляє один об'єкт або запис, а кожен стовпець містить певний атрибут цих об'єктів.

Рядок – це один окремий запис у таблиці, що містить дані про певний об'єкт або сутність. Кожен рядок має значення для кожного поля таблиці.

Стовпець – це один атрибут або характеристика, яка визначає дані, що зберігаються у таблиці. Кожен стовпець має унікальне ім'я та тип даних, який визначає допустимий формат даних для цього поля.

Ключ – це унікальний ідентифікатор для кожного запису в таблиці. Він дозволяє унікально ідентифікувати кожен рядок і забезпечує швидкий доступ до даних.

Запит – це команда або запит, який надсилається до бази даних для отримання, оновлення або видалення даних. *SQL* (*Structured Query Language*) є найпоширенішою мовою для створення запитів до реляційних баз даних.

Нормалізація – це процес організації даних у базі даних для усунення дублікатів та забезпечення ефективності зберігання. Нормалізація допомагає забезпечити цілісність даних та зменшити аномалії при оновленні даних.

СКБД – це програмне забезпечення, яке дозволяє зберігати, управляти та обробляти дані в базі даних. Відомими СКБД є: *MySQL, PostgreSQL, Microsoft SQL Server, Oracle, MongoDB.*

Індекс – це структура даних, яка прискорює пошук і доступ до даних у таблиці. Вона створюється на основі значень одного або декількох стовпців таблиці.

Транзакція – це послідовність одного або декількох запитів, які виконуються як єдине ціле. Транзакції дозволяють забезпечити атомарність, цілісність, ізоляцію та стійкість даних у базі даних.

Бекап – це копія бази даних, яка створюється для забезпечення захисту даних в разі аварій або втрати інформації. Регулярне створення бекапів є важливою практикою для забезпечення безпеки даних.

Класифікація баз даних

Бази даних можна класифікувати за різними критеріями.

1. За моделлю даних:

- реляційна модель – це найпоширеніший і широко використовуваний тип моделі організації даних. Вона базується на математичній теорії множин і заснована на поняттях таблиць з рядками і стовпцями. В кожному рядку зберігаються дані про певний об'єкт, а кожен стовпець представляє певний атрибут цих об'єктів. Спрощена структура реляційних баз даних дозволяє

легко маніпулювати даними та здійснювати запити з мовою *SQL*;

- ієрархічна модель – це модель організованих даних у вигляді ієрархії батьківських і дочірніх вузлів, яка нагадує структуру дерева. Кожен запис може мати багато дочірніх записів, але тільки один батьківський. Ця модель широко використовувалась в давніх старих СКБД, таких як *IMS* (*Information Management System*);
- мережева модель – це розширення ієрархічної моделі, де кожен запис може мати багато батьківських та дочірніх записів. В цій моделі дочірні записи пов'язані з батьківськими через «відносини» (ребра графа). Це дозволяє більш гнучко організовувати дані, але також призводить до складнішого управління даними;
- об'єктно-орієнтована модель – це модель організації даних, що використовує об'єктно-орієнтовані підходи для зберігання і обробки даних. Вона дозволяє зберігати дані як об'єкти з властивостями (атрибутами) та методами (функціями) для обробки цих даних. Цей підхід особливо корисний, коли необхідно працювати зі складними структурами даних, наприклад, в програмуванні та розробці програмного забезпечення;
- графова модель – це модель організації даних, що базується на математичній теорії графів і використовує графи для представлення структури даних. Вона ідеально підходить для моделювання взаємозв'язків між об'єктами, де вузли представляють об'єкти, а ребра показують зв'язки між ними. Графові бази даних здатні ефективно опрацьовувати складні структури даних та зв'язки.

2. За кількістю користувачів:

- однокористувацькі бази даних, що призначені для використання одним користувачем або додатком;
- багатокористувацькі бази даних, що призначені для одночасного використання багатьма користувачами або додатками.

3. За розміщенням даних:

- локальна база даних – це база даних, що знаходиться і зберігається на локальному пристрої або комп'ютері. Доступ до даних відбувається безпосередньо на цьому пристрої, іноді через локальну мережу. Такі бази даних зазвичай використовуються для особистих проектів, малих бізнесів або навіть для розгортування та тестування на окремому комп'ютері;
- розподілена база даних – базах даних в якій дані фізично знаходяться на різних комп'ютерах або серверах, що розташовані у різних локаціях. Ці сервери можуть бути підключені до однієї мережі або взаємодіють через Інтернет. Розподілені бази даних дозволяють зберігати великі обсяги даних, забезпечувати доступ та резервне копіювання даних;
- централізована база даних – це база даних де всі дані знаходяться на одному центральному сервері або в централізованому сховищі. Користувачі та програми отримують доступ до даних через центральний вузол. Цей тип баз даних використовується в багатьох традиційних організаціях та підприємствах;
- розподілена централізована база даних – це комбінація підходів розподіленої та централізованої бази даних. Дані розподілені між різними локаціями, але управління та контроль щодо даних здійснюється через центральний вузол;
- хмарна база даних – це база даних де дані зберігаються та обробляються на хмарних обчислювальних платформах. Цей підхід дозволяє легко масштабувати ресурси та забезпечує зручний доступ до даних через Інтернет;
- мобільні бази даних – це бази даних, які забезпечують зберігання та доступ до даних на мобільних пристроях, таких як смартфони і планшети. Такі бази даних можуть працювати автономно або синхронізуватися з серверами, що дозволяє забезпечувати доступ до даних навіть за відсутності Інтернет-з'єднання;

4. За метою використання:

- операційні бази даних, що використовуються для повсякденної операційної діяльності, такої як обробка транзакцій та запитів в режимі реального часу;

- аналітичні бази даних, що призначені для аналізу великого обсягу даних, використовуються для створення звітів та бізнес-аналітики.

5. За технологією фізичного зберігання:

- дискова база даних – це найпоширеніший тип баз даних, де дані зберігаються на фізичних дисках комп'ютерів або серверів. Вони можуть бути жорсткими дисками (*HDD*) або твердотілими дисками (*SSD*). Дискові бази даних забезпечують високу міцність і постійний доступ до даних;
- ініційована у пам'яті (*In-Memory*) база даних – цей тип баз даних зберігає всі дані у оперативній пам'яті (*RAM*) комп'ютера, а не на диску. Такий підхід зазвичай забезпечує значно швидший доступ до даних порівняно з диском, що робить ініційовану у пам'яті базу даних привабливою для додатків, які потребують високої продуктивності та відгук у реальному часі;
- колоційовані бази даних – у таких базах даних дані зберігаються на серверах, що розташовані на тому ж фізичному носії або в одному дата-центрі. Це знижує затримку доступу до даних і дозволяє більш ефективно обмінюватися даними між серверами;
- розподілені бази даних – це бази даних де дані фізично розподілені між різними серверами або вузлами, які можуть бути розташовані на різних локаціях. Розподілена база даних може використовувати різні технології фізичного зберігання для кожного вузла.
- хмарна база даних – ц база даних де дані зберігаються у хмарних обчислювальних платформах, таких як *Amazon Web Services* (*AWS*), *Microsoft Azure* або *Google Cloud Platform*. Цей підхід надає гнучкість та легкий доступ до даних через Інтернет;
- файлові системи – цей тип організації даних використовує файлову систему операційної системи для зберігання та управління даними. Він може включати текстові файли, бінарні файли, *XML*-файли та інші формати;
- об'єктно-орієнтовані бази даних – у цих базах даних дані зберігаються у вигляді об'єктів з властивостями та методами, подібно до того, як об'єкти використовуються в об'єктно-орієнтованому програмуванні.

6. За архітектурою:

- централізовані бази даних, у яких всі дані знаходяться на одному центральному сервері або в централізованому сховищі;
- розподілені централізовані бази даних, у яких дані розподілені між різними локаціями, але управління та контроль здійснюється через центральний вузол.

7. За ступенем роздільності даних:

- фізично-незалежні бази даних, у яких можна змінювати структуру не змінюючи програми, які з ними працюють;
- фізично-залежні бази даних, зміни в структурі яких можуть впливати на програми, які з ними працюють.

У реальних проектах бази даних можуть поєднувати різні характеристики та особливості, щоб відповідати конкретним вимогам та потребам.

Існує кілька мов програмування та запитів [5], які використовуються для роботи з базами даних. Вибір конкретної мови залежить від типу бази даних, яка використовується, а також вимог до програмного продукту. Далі розглянуто популярні мови, які часто використовуються для роботи з базами даних.

1. *SQL* є стандартною мовою для роботи з реляційними базами даних. Вона дозволяє виконувати різні операції з даними, такі як створення таблиць, вставка, оновлення, видалення, вибірка, об'єднання таблиць, групування, сортування та багато інших. *SQL* можна використовувати в багатьох реляційних СУБД, таких як *MySQL, PostgreSQL, Microsoft SQL Server, Oracle* і *SQLite*.

2. *Python* є популярною мовою програмування з великими засобами для роботи з базами даних. Вона має різні бібліотеки та модулі, які дозволяють підключатися до баз даних і виконувати запити. Наприклад, бібліотека *sqlite*3 в *Python*

дозволяє взаємодіяти з базами даних *SQLite*, а *psycopg*2 допомагає працювати з *PostgreSQL*.

3. *Java* також є популярною мовою програмування для роботи з базами даних. Вона має багато бібліотек і *API* для підключення до різних СКБД. *JDBC* (*Java Database Connectivity*) – це стандартний *API* для роботи з реляційними базами даних у *Java*.

4. *C#* є мовою програмування, яка часто використовується для розробки додатків на платформі *.NET*. Вона підтримує роботу з реляційними базами даних за допомогою *ADO.NET*.

5. *PHP* є широко використовуваною мовою програмування для веб-розробки. Вона має вбудовану підтримку для багатьох реляційних баз даних, таких як *MySQL, PostgreSQL* і *SQLite*.

6. *Ruby* має розширення для роботи з базами даних, таким як *ActiveRecord* для реляційних баз даних або *MongoMapper* для роботи з базами даних *MongoDB*.

JavaScript також може використовуватися для роботи з базами даних, особливо в контексті веб-розробки. Наприклад, з допомогою *Node.js* можна взаємодіяти з реляційними та *NoSQL* базами даних.

Вибір моделі бази даних

Ключовими критеріями при розробці бази даних для збереження показників якості повітря з автоматизованих постів та станцій моніторингу є:

1. Інформація про тип поста чи станції моніторингу, які використовуються для збору даних. Це дозволить ідентифікувати дані, які були зібрані з кожного поста або станції, а також оцінити точність даних.

2. Місце розташування постів та станцій моніторингу, що дозволить відстежувати зміни якості повітря в різних частинах міста або регіону.

3. Час збору даних, що дозволить відстежувати зміни якості повітря в часі.

4. Показники якості повітря, що дозволяють оцінити якість повітря в різних частинах міста або регіону.

5. Точність даних, що дозволяє оцінити надійність даних.

Окрім цих ключових критеріїв, база даних може містити також інші дані, такі як:

1. Ім’я постів та станцій моніторингу, що дозволить користувачам легко знаходити потрібні дані.

2. Опис постів та станцій моніторингу, що дозволить користувачам дізнатися більше про пости та станції моніторингу, які були використані для збору даних.

3. Технічні характеристики постів та станцій моніторингу, що дозволить користувачам оцінити можливості постів та станцій моніторингу, які були використані для збору даних.

4. Стандарти якості повітря, що дозволить користувачам оцінити якість повітря на основі стандартів якості повітря.

Розробка бази даних для збереження показників якості повітря з різних автоматизованих постів та станцій моніторингу є важливим кроком у забезпеченні ефективного моніторингу якості повітря. База даних повинна містити інформацію, яка є точною, надійною та доступною для користувачів.

Враховуючи зазначені критерії, додаткові дані про пости та станції досліджень повітря та надані за технічним завданням набори сирих даних для обробки та збереження у базі даних, обрано модель бази даних, яку можна класифікувати наступним чином за наступними критеріями:

1. За моделлю даних обрано реляційну модель *SQL*, що є найпоширенішим типом моделі організації даних, а отже наявна велика кількість бібліотек і методів при роботі з такими базами даних, що зменшує час на реалізацію проєкту.

2. За кількістю користувачів обрано однокористувацьку базу даних, оскільки дані туди записуватимуться/зчитуватимуться комплексом програм, що фа4ктично являються одним додатком.

3. За розміщенням даних обрано локальну базу даних, що буде встановлена на локальному сервері, разом з серверним додатком, який працюватиме з базою даних та обслуговуватиме клієнтів.

4. За метою використання визначено що база даних відноситиметься до аналітичних, оскільки має зберігати великі об'єми сирих даних, їх зрізи та результуючі дані.

5. За технологією фізичного зберігання база даних відноситиметься до дискових, адже з такою базою даних легко буде працювати та вона постійно буде доступна для роботи, а дані у ній надійно зберігатимуться на фізичному носії з можливістю відновлення.

6. За архітектурою обрано централізовану базу даних у якій всі дані зберігатимуться на одному сервері, що обслуговуватиме клієнтів та отримуватиме дані зі станцій.

7. За ступенем роздільності даних обрано фізично-незалежну базу даних, що дозволить удосконалення структури та розширення функціоналу додатку, що працюватиме з базою даних, без зміни раніше реалізованого функціоналу.

Мовою програмування для реалізації серверного функціоналу обрано *PHP*, оскільки ця мова широко використовується для веб-розробки та має крім вбудованої підтримки багатьох реляційних баз даних, таких як *MySQL, PostgreSQL* і *SQLite*, велику кількість потужних фреймворків, що значно пришвидшують написання та підтримання баз даних моделі *SQL*.

Аналіз систем керування базами даних

Популярні СКБД з моделлю даних *SQL*, що використовуються для реалізації проєктів середнього розміру [6] такі.

- *PostgreSQL* – це потужна відкрита об'єктно-реляційна система управління базами даних (СКБД), яка підтримує реляційну модель даних та має широкий набір функцій [7]. Вона відрізняється високою надійністю, можливістю зберігання геоданих через розширення *PostGIS* та додатковими опціями, такими як *JSON, XML*, та *hstore*. *PostgreSQL* є добрим вибором для проектів, де потрібна гнучка і надійна система управління базами даних з активною спільнотою розробників.
- *MySQL* – це відкрита реляційна система управління базами даних, розроблена компанією *Oracle*, яка використовується у різних застосунках, зокрема веб-додатках [8]. Вона відрізняється простотою у використанні, високою швидкодією, та легковажністю, що дозволяє її використовувати в багатьох веб-серверах та простих додатках. Існує безкоштовна версія *MySQL Community Edition* з відкритим кодом, яка є популярним вибором для малих та середніх проектів з обмеженими бюджетами.
- *Microsoft SQL Server* – це комерційна реляційна система управління базами даних, розроблена компанією *Microsoft* [9]. Вона є однією з найпоширеніших СКБД в корпоративному середовищі та пропонує різноманітні можливості для зберігання, обробки та аналізу великих обсягів даних. *SQL Server* має високий рівень безпеки, підтримує реплікацію та шкальованість, що робить його ідеальним вибором для критичних застосунків з вимогами до надійності та продуктивності. Крім того, він має широкий набір інструментів та інтеграцію з екосистемою *Microsoft*, що дозволяє забезпечити гарну сумісність з іншими продуктами *Microsoft*.
- *Oracle Database* – це одна з найпоширеніших комерційних реляційних СКБД, розроблена компанією *Oracle* [10]. Вона володіє великим рівнем надійності, масштабованості та продуктивності, що робить її популярним вибором для великих корпоративних проектів. *Oracle Database* пропонує широкий набір функцій, включаючи продвинуту аналітику, високий рівень безпеки та реплікацію даних. Компанія *Oracle* також надає підтримку та регулярні оновлення для забезпечення стабільної та ефективної роботи з базами даних. Велика спільнота користувачів та партнерів підтримує розширений екосистема *Oracle*, що дозволяє знайти рішення для різноманітних бізнес-викликів.
- *SQLite* – це легковажна вбудовувана реляційна система управління базами даних [11], яка має невеликий розмір та низький рівень вимог до ресурсів. Вона здатна працювати без сервера баз даних, що робить її ідеальним вибором для

вбудовування в мобільні додатки, веб-браузери, *IoT*-пристрої та інші застосунки, які вимагають швидкості та ефективності. *SQLite* підтримує більшість стандартних операцій *SQL*, так що використання її є досить зручним для розробників, що знайомі з мовою *SQL*. Її простота у використанні та широкий коло застосувань робить *SQLite* одним з популярних виборів для малих та середніх проектів.

- *MariaDB* – це відкритий форк системи керування базами даних *MySQL* з акцентом на відкритий код та активну спільноту розробників [12]. Вона зберігає сумісність з *MySQL*, але також пропонує додаткові функції, покращення та оптимізації, що роблять її привабливим вибором для розробників, які шукають альтернативу *MySQL*. *MariaDB* підтримує геодані за допомогою розширення *MariaDB GIS*, а також пропонує підтримку для шифрування даних та інших функцій безпеки. Її легковажність та зручний інтерфейс роблять *MariaDB* популярним вибором для веб-додатків, мобільних додатків та інших проектів з вимогами до продуктивності та безпеки.

- *IBM Db*2 – це потужна та високопродуктивна система управління базами даних (СКБД), розроблена компанією *IBM* [13]. Вона відзначається широким набором функцій, які забезпечують надійну та ефективну роботу з великими обсягами даних. *IBM Db*2 підтримує реляційну модель даних та надає розширені можливості для аналітики та обробки даних в режимі реального часу. Вона також пропонує інструменти для оптимізації продуктивності, масштабованості та безпеки, що робить її популярним вибором для великих корпоративних застосунків. Крім того, *IBM Db*2 має велику спільноту користувачів та партнерів, що підтримує розширений екосистему, що дозволяє знайти рішення для різноманітних бізнес-викликів та інтеграцію з іншими системами та сервісами.

Вибір системи керування базою даних

MySQL є кращим варіантом для збереження даних про показники екологічного стану атмосферного повітря з кількох причин.

По-перше, вона володіє великою спільнотою користувачів і розробників, що забезпечують наявність ресурсів, бібліотек, модулів та підтримки у разі виникнення проблем з роботою БД.

По-друге, *MySQL* володіє гарними показниками продуктивності та масштабованості, що дозволяє їй ефективно працювати з великими обсягами сирих даних, що збираються з автоматизованих постів та станцій екологічного моніторингу, а також накопичується за рік та можуть бути внесені вручну. Об’єм опрацьовуваних сирих даних становить не менше двох гігабайт. Її оптимізоване ядро забезпечує швидкий доступ до інформації, що є важливим аспектом для систем, які мають обробляти та аналізувати великі обсяги даних у реальному часі. Також виникає необхідність зберігати у базі даних проміжні зрізи, що формуються при первинній обробці сирих даних. *MySQL* забезпечує достатню стабільність та продуктивність роботи системи за таких умов.

По-третє, *MySQL* є легкою СКБД, яка не потребує великих обчислювальних ресурсів для свого функціонування. Це дозволяє ефективно використовувати її на різних платформах, включаючи мобільні пристрої та *IoT*-пристрої, що можуть бути використані в мобільних додатках та системах моніторингу стану повітря. Більш того, зручність у використанні та невисока вартість роблять *MySQL* привабливим вибором для проєктів з обмеженими бюджетами або підприємств, які шукають ефективне та доступне рішення для збереження та аналізу даних про екологічний стан повітря.

Розробка структури бази даних

Розроблена структура бази даних представлена на рис. 1.

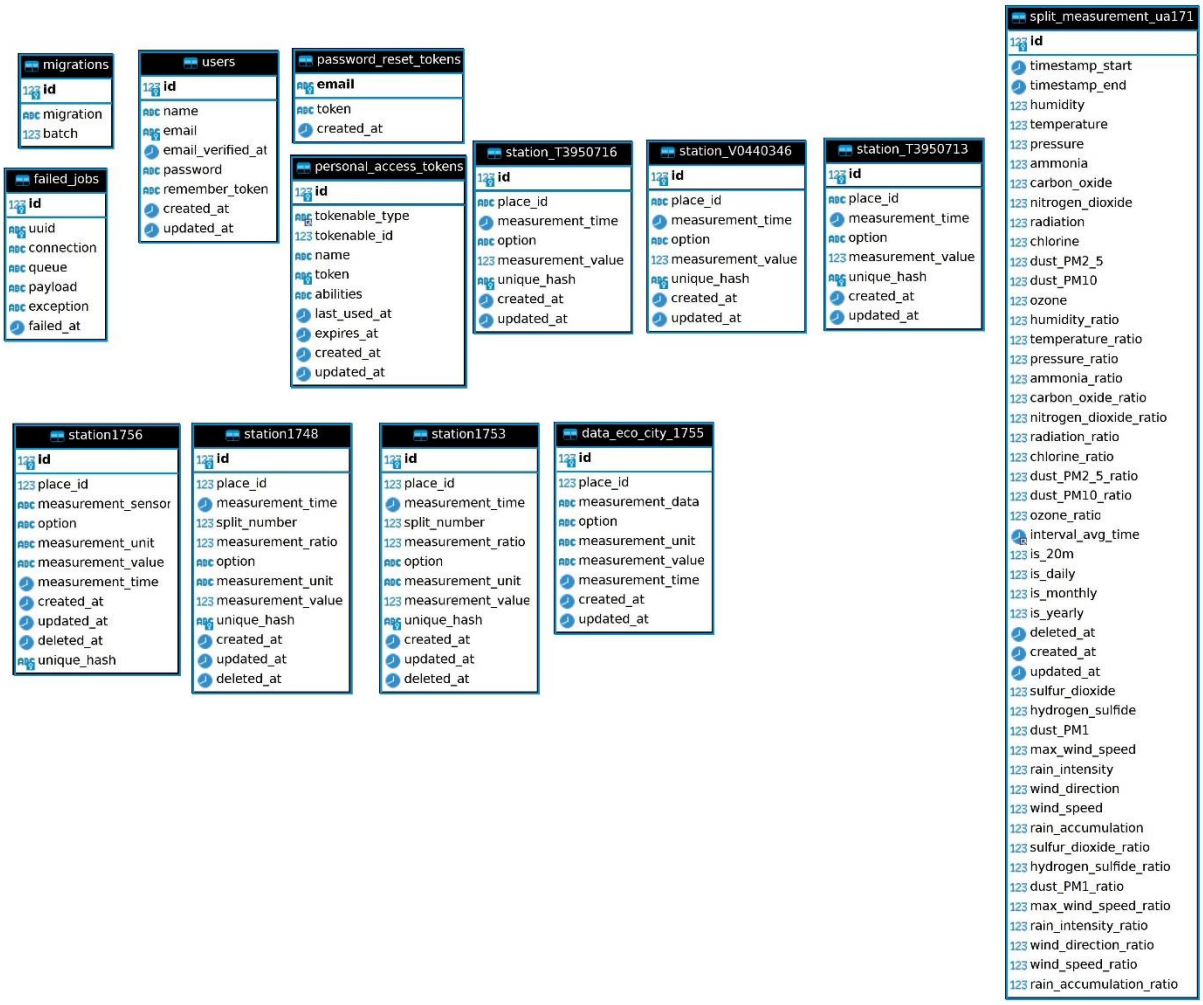

Рис. 1. Розроблена структура бази даних

База даних включає такі набори даних:

migrations – інформація про міграції – це процес створення та оновлення структури бази даних відповідно до змін у додатку. Міграції зазвичай пишуться як *PHP*-файли, які містять *SQL*-запити, які потрібно виконати для створення або оновлення таблиць бази даних;

failed_jobs – це таблиця, яка зберігає записи про невдалі завдання в черзі для роботи з станціями *Vaisala*. Ця таблиця використовується для логування невдалих завдань, щоб їх можна було переглядати та виправляти;

users – зберігає записи про користувачів зареєстрованих на сервері. Ця таблиця використовується для зберігання інформації про користувачів, таких як їхні імена, електронні адреси, паролі та інші дані;

password_reset_tokens – зберігає записи про токени для скидання пароля доступу до додаткових можливостей сервера;

personal_access_tokens – зберігає записи про токени доступу до станцій *Vaisala* за допомогою *API*. Ця таблиця використовується для зберігання інформації про особисті токени доступу, ідентифікатор користувача, який створив токен, мету токена та дату закінчення дії токена;

*station*1756, *station*1748, *station*1753, *data_eco_city*_1755 – таблиці даних станцій екологічного моніторингу. Даній від станції надходять у двох форматах: від станції *station*1756 сирі дані отримуються кожну хвилину, а від станцій *station*1748, *station*1753, *data_eco_city*_1755 надходять усередненні дані за кожні 20 хвилин. Номери станцій вказують на конкретну станцію, наприклад, маємо код 1748, 17

відповідає номеру області досліджень, а 48 номеру станції;

*station_T*3950716, *station_V*0440346, *station_T*3950713 – дані з станцій *Vaisala*. Станції *Vaisala* мають власні коди, що відповідають їх серійному номеру, для спрощення комунікації з ними за допомогою *API*. Зважаючи на особливості конструкції станцій та їх різну комплектацію для кожної зроблено власну таблицю;

*split_measurement_ua*171 – таблиця даних після обробки, що включає середні значення за визначені проміжки часу, повноту даних та додаткові метеорологічні та екологічні показники, що наразі знаходяться в процесі інтеграції в систему. Код *ua*171 ідентифікує таблицю в межах глобальної області дослідження: 17 відповідає за область дослідження, а 1 вказує на номер міста в досліджуваній області.

Визначення вмісту пов'язаних таблиць бази даних

Розроблені таблиці в базі даних мають наступний вміст:

- *migrations*:

а) *migration*: ім'я файлу міграції;

б) *batch*: номер пакету міграції;

- *failed_jobs*:

а) *uuid*: унікальний *UUID* завдання;

б) *connection*: ім'я з'єднання, яке було використано для виконання завдання;

в) *queue*: черга, до якої було додано завдання;

г) *payload*: порожнє тіло завдання;

д) *exception*: виняток, який стався під час виконання завдання;

е) *failed_at*: дата та час, коли завдання не вдалося;

- *users*:

а) *name*: ім'я користувача;

б) *email*: електронна адреса користувача;

в) *email_verified_at*: дата підтвердження електронної адреси;

г) *password*: пароль користувача;

д) *remember_token*: токен для запам'ятовування користувача;

е) *created_at*: дата та час створення користувача;

є) *updated_at*: дата та час оновлення користувача;

- *password_reset_tokens*:

а) *token*: токен для скидання пароля;

б) *created_at*: дата та час створення токена для скидання пароля;

- *personal_access_tokens*:

а) *tokenable_type*: тип активації токена;

б) *tokenable_id*: ідентифікатор активації токена;

в) *name*: найменування особистого токена доступу;

г) *token*: токен доступу;

д) *abilities*: дозволи особистого токена доступу;

е) *lust_used_at*: дата та час останнього використання;

є) *expires_at*: дата та час закінчення дії особистого токена доступу;

ж) *created_at*: дата та час створення особистого токена доступу;

з) *updated_at*: дата оновлення токена;

- *station*1756, *station*1748, *station*1753, *data_eco_city_*1755, *station_T*3950716, *station_V*0440346, *station_T*3950713:

а) *place_id*: надлишковість у вигляді ідентифікатору станції;

б) *measurement_time*: час вимірювання показника станцією;

в) *measurement_data*: дата вимірювання показника станцією;

г) *option*: назва вимірюваного показника;

д) *measurement_unit*: одиниці виміру показника;

е) *measurement_value*: виміряне значення показника;

є) *measurement_ratio*: повнота даних отриманих від станцій;

ж) *created_at*: дата та час запису рядка у базу даних;

з) *updated _at*: дата та час оновлення рядка у базу даних;

и) *deleted_at*: дата та час видалення рядка з набору для обробки;

і) *unique_hash*: унікальний хеш рядка даних для запобігання їх повторення;

- *split_measurement_ua*171:

а) *timestamp_start*: початок обраного часового проміжку для генерації звіту;

б) *timestamp_end*: кінець обраного часового проміжку для генерації звіту;

в) *humidity, temperature, pressure, ammonia, carbon_oxide, nitrogen_dioxide, radiation, chlorine, dust_PM*2_5, *dust_PM*10, *ozone, sulfur_dioxide, hydrogen_sulfide, dust_PMl, max_wind_speed, rain_intensity, wind_direction, wind_speed, rain_accumulation*: показники стану атмосферного повітря та метеорологічні показники приведені до однакових одиниць виміру за певний проміжок часу;

г) *humidity_ratio, temperature_ratio, pressure_ratio, ammonia_ratio, carbon_oxide_ratio, nitrogen_dioxide_ratio, radiation_ratio, chlohne_ratio, dust_PM*2_5*_ratio, dust_PM*10*_ratio, ozone_ratio, sulfur_dioxide_ratio, hydrogen_sulfide_ratio, dust_PMl_ratio, max_wind_speed_ratio, rain_intensity_ratio, wind_direction_ratio, wind_speed_ratio, rain_accumulation_ratio*: повнота даних відповідник показників у базі даних;

д) *is_*20*m, is_daily, is_monthly, is_yearly*: проміжок часу за який обрається середнє значення показника для формування звіту;

е) *interval_avg_time*: середнє значення часу для побудови графіків змін показників;

є) *deleted_at*: дата та час видалення таблиці;

ж) *created_at*: дата та час створення таблиці;

з) *updated_at*: дата та час оновлення таблиці.

Програмні засоби для реалізації бази даних

Для запису у базу даних показників, що отримуються за допомогою *API* від станцій *Vaisala* [14] з сенсорами *AQT*530 та *WXT*530, вміст отриманих повідомлень проаналізовано з використанням *Postman API Platform*.

Postman API Platform – це інтегрована платформа для розробки та управління *API* [15], яка надає засоби для спрощення процесу створення, тестування, документування та взаємодії з *API*.

Першим ключовим компонентом *Postman API Platform* є сам *Postman*, який є потужним клієнтом для тестування та взаємодії з *API*. Він дозволяє розробникам відправляти *HTTP*-запити до *API*, перевіряти результати, а також легко візуалізувати та аналізувати дані з відповідей. Крім того, *Postman* забезпечує можливість автоматизації тестування *API* за допомогою зручних засобів, що спрощують процес тестування та дозволяють виявляти та виправляти помилки швидше.

Другий компонент – *Postman API Network*, є ресурсом для спільноти розробників, де можна знайти тисячі готових колекцій *API* та прикладів, які можна безпосередньо використовувати для розробки власних проектів. Це сприяє підвищенню продуктивності та покращенню якості розробки *API*, оскільки розробники можуть використовувати перевірені рішення та підходи.

Третій компонент – *Postman API Monitoring*, дозволяє розробникам контролювати продуктивність своїх *API* в реальному часі. Завдяки цьому інструменту, розробники можуть відстежувати метрики продуктивності, виявляти перевантаження та проблеми, що виникають при взаємодії з *API*, тим самим допомагаючи підтримувати високу доступність та надійність сервісів для користувачів.

Для розробки основного функціонала *php* веб-сервера було використано фреймворк *Laravel* та розширення *PDO*, а також обрану систему керування базами даних *MySQL*.

Laravel – це популярний веб-фреймворк для мови програмування *PHP*, який створений з метою спростити та прискорити процес розробки веб-додатків [16]. Завдяки своїй зручній та оновленому синтаксису, *Laravel* дозволяє розробникам швидко створювати якісні та потужні вебдодатки з різноманітними функціональними можливостями. Фреймворк включає в себе багато готових компонентів, таких

як аутентифікація, маршрутизація, сесії, кешування та інші, що спрощує розробку та забезпечує більшу продуктивність роботи розробників.

Однією з особливостей *Laravel* є його *ORM* (*Object-Relational Mapping*) – *Eloquent* [17], який дозволяє легко взаємодіяти з базами даних за допомогою об'єктно-орієнтованого підходу. *Eloquent* дозволяє розробникам працювати з базами даних, використовуючи синтаксис *PHP*-класів, що робить роботу з даними більш зрозумілою та інтуїтивною. Крім того, *Laravel* забезпечує зручний механізм міграцій, що дозволяє легко змінювати структуру бази даних та керувати версіями схеми.

Ще однією перевагою *Laravel* є його активна спільнота розробників, яка постійно поповнює фреймворк новими розширеннями та інструментами. Завдяки цій спільноті, *Laravel* знаходиться в постійному розвитку, що дозволяє розробникам використовувати найновіші технології та найкращі практики у своїх проектах

PHP PDO (*PHP Data Objects*) – це розширення мови програмування *PHP*, яке надає інтерфейс для взаємодії з реляційними базами даних [18]. Воно є альтернативою застарілій функції *MySQL* та дозволяє розробникам працювати з різними базами даних, такими як *MySQL, PostgreSQL, SQLite* та іншими, використовуючи єдиний інтерфейс.

Першим ключовим перевагою *PHP PDO* є його кросс-платформеність та гнучкість. Розробники можуть писати універсальний код для роботи з базами даних, інакше кажучи, вони можуть переключатись між різними системами управління базами даних без необхідності переписувати весь код. Це робить *PHP PDO* ідеальним вибором для проектів, які можуть зазнавати змін в майбутньому.

Другою важливою особливістю *PHP PDO* є його підтримка параметризованих запитів. Це дозволяє розробникам виконувати запити до баз даних з використанням параметрів, що дозволяє уникнути *SQL*-ін'єкцій, забезпечуючи безпеку взаємодії з базою даних. Запити з параметрами також покращують продуктивність, оскільки база даних може кешувати операції та перевикористовувати скомпільовані запити для покращення швидкості виконання.

Також, *PDO* надає розробникам можливість використовувати транзакції для керування операціями з базою даних. Транзакції дозволяють виконувати групу запитів як один атомарний блок, тобто або всі запити успішно виконуються, або ж жоден з них не застосовується. Це дозволяє забезпечити цілісність даних та уникнути неконсистентних станів бази даних, що є критично важливим для багатьох додатків з великою кількістю одночасних взаємодій користувачів.

Для реалізації веб-серверу, що працюватиме з базою даних заплановано використати *Nginx*.

Nginx – це веб-сервер та проксі-сервер з високою продуктивністю, який розроблений з метою забезпечення швидкої та ефективної обробки *HTTP*-запитів [19]. Цей сервер написаний на мові *C* і заснований на події «*asynchronous event-driven*» архітектурі, що дозволяє ефективно взаємодіяти з багатьма запитами одночасно. Найпоширеніше використання *Nginx* – це обслуговування веб-сайтів, зокрема статичних файлів, а також виконання функцій проксі-сервера для розподілу навантаження на додаткові ресурси.

Однією з основних переваг *Nginx* є його низький рівень споживання ресурсів системи. Він демонструє високу продуктивність при одночасній обробці тисяч або навіть мільйонів з'єднань, що робить його особливо привабливим для високонавантажених проектів. Крім того, *Nginx* відзначається надійністю та стабільністю роботи, що забезпечує безперебійну роботу веб-додатків навіть при великому навантаженні.

Ще однією важливою особливістю *Nginx* є його гнучкість та можливості конфігурації. Він підтримує різні види обробки запитів, включаючи статичний контент, обробку *PHP* та інші мови програмування, а також проксі-послуги та зворотній

проксі-сервер для балансування навантаження між різними серверами. Інтерфейс конфігурації *Nginx* зрозумілий та дозволяє легко налаштувати його для відповідності потребам конкретного проєкту.

А операційною системою для запуску *Nginx* обрано *Ubuntu* [20].

Ubuntu – це один із найпопулярніших та безкоштовних дистрибутивів операційної системи *Linux*. Розроблений компанією *Canonical*, *Ubuntu* базується на дистрибутиві *Debian* і відомий своєю простотою в установці та використанні, що робить його привабливим для широкого кола користувачів. Основними цілями проєкту *Ubuntu* є створення доступного та відкритого операційного середовища для роботи з персональними комп'ютерами, серверами та хмарними обчисленнями.

Один із ключових аспектів *Ubuntu* – це активна спільнота розробників і користувачів, яка підтримує його постійний розвиток і вдосконалення. Завдяки відкритому вихідному коду та доступності інструментів для розробки, користувачі можуть активно приєднуватись до проекту, долучатись до розробки, вносити зміни та вирішувати проблеми.

Апаратні засоби реалізації бази даних

Реалізацію сервера заплановано здійснити за допомогою технологій, що пропонуються хостингами.

Хостинг – це послуга надання простору на серверах для розміщення веб-сайтів та забезпечення їх доступності в Інтернеті [21]. Це важлива інфраструктура, яка дозволяє публікувати веб-сторінки та надавати їх відвідувачам за допомогою глобальної мережі. Хостинг-провайдери надають різні типи хостингу, такі як спільний хостинг, віртуальний приватний сервер (*VPS*), область хмарних обчислень та власний фізичний сервер. Кожен тип хостингу має свої переваги та обмеження, що дозволяє забезпечити індивідуальні потреби різних користувачів.

Спільний хостинг – це найпоширеніший тип хостингу, де кілька веб-сайтів розміщуються на одному фізичному сервері та ділять обчислювальні ресурси. Це зручний та економічний варіант для невеликих та середніх проектів, але може призвести до обмеження продуктивності у разі великої кількості відвідувачів на одному хостинг-аккаунті. Віртуальний приватний сервер (*VPS*) [22] надає більшу незалежність та контроль над ресурсами, оскільки кожен *VPS* має своє власне віртуальне середовище з власними обчислювальними ресурсами та окремою операційною системою. Це дозволяє використовувати *VPS* для більших та більш складних проектів з більшим обсягом відвідувачів та ресурсів.

Область хмарних обчислень є новим та перспективним рішенням для хостингу, де ресурси розподіляються між кількома серверами в режимі реального часу. Хостинг на хмарних серверах надає більшу масштабованість та гнучкість, оскільки ресурси можуть збільшуватись або зменшуватись в залежності від потреб проекту. Це дозволяє забезпечити оптимальну продуктивність навіть при різкому зростанні відвідувачів. Хмарний хостинг також відрізняється високим рівнем доступності, оскільки він базується на розподілених серверах, що забезпечує безперебійну роботу веб-додатків навіть у разі збоїв на одному з серверів.

Зважаючи на вартість місячної оренди ресурсів хостингу, для першої ітерації реалізації проєкту доцільно спинитися на спільному хостингові. За 300 грн/міс. хостинг пропонує 20 Гб дискового простору та 1024 Мб оперативної пам'яті. Зважаючи на отриманий об'єм сирих даних та досвід роботи з аналізу даних, передбачено, що для зберігання сирих даних зі станцій та їх оброблених зрізів за проміжок часу з 2021 по 2024 потрібно 20 Гб дискового простору. За рік з однієї станції накопичується близько 4,5 Гб даних. Розроблене програмне забезпечення сервера, що включає *frontend* та *backend*, без врахування бази даних, займає 360 Мб простору. 2,3 Гб необхідно для встановлення операційної системи *Ubuntu* на сервер. При роботі операційної системи *Ubuntu* їй потрібно 700 Мб оперативної пам'яті для стабільної роботи.

Передбачається можливість видалення сирих даних та проміжних зрізів з серверу, залишаючи лише кінцеві усереднені дані, що звільнить значну частину дискового простору для отримання нових даних більшу кількість років чи дозволить підключити більше станцій, за умов щоб залишитися в межах виділених 20 Гб. Фоновий процес роботи *php* веб-сервера займає у оперативній пам'яті близько 20 Мб. Враховуючи, що для обслуговування сервером одного клієнта у оперативній пам'яті необхідно виділити мінімум 30 Мб, то одночасно з даними зможуть працювати до 10 клієнтів. Враховуючи, що зазвичай клієнти не надсилають запити одночасно і завжди в мережах присутня деяка затримка, то сервер зможе обслугувати більшу кількість клієнтів, але час їх обслуговування може збільшитися.

Розроблені моделі вмісту бази даних

Оскільки робота з базами даних базується на використанні функцій *Laravel* та його модуля *Eloquent*, то спершу для таблиць бази даних було створено їх моделі даних, що зберігаються у папці *app/Models*.

Наприклад код моделі для *Station*1748 має наступний вміст.

```
<?php
namespace App\Models;
use
Illuminate\Database\Eloquent\Factories\Has
Factory;
use Illuminate\Database\Eloquent\Model;
use
Illuminate\Database\Eloquent\SoftDeletes;
class Station1748 extends Model
{
    use HasFactory;
    use SoftDeletes;
    protected $table = 'station1748';
    protected $fillable = [
        'place_id',
        'measurement_time',
        'split_number',
        'measurement_ratio',
        'option',
        'measurement_unit',
        'measurement_value',
        'unique_hash',
    ];
    protected static function boot()
    {
        parent::boot();
        static::creating(function ($model) {
            $model->unique_hash = md5(
                $model->place_id
                . $model->split_number
                . $model->option
                . $model->measurement_value
                . $model->measurement_time
            );
        });
    }
}
```

У ході роботи розроблено моделі таких таблиць: *migrations*; *failed_jobs*; *users*; *password_reset_tokens*; *personal_access_tokens*; *station*1756; *station*1748; *station*1753; *data_eco_city_*1755; *station_T*3950716, *station_V*0440346; *station_T*3950713; *split_measurement_ua*171.

Міграції бази даних

Migrations у *Laravel* – це механізм, що дозволяє розробникам легко та структуровано керувати базою даних у процесі розвитку веб-додатків [23]. Цей інструмент допомагає автоматично створювати та змінювати таблиці бази даних, зберігаючи їх структуру та стан у вигляді коду. *Migrations* використовує мову *PHP* для опису структури бази даних, що робить його зрозумілим та доступним для розробників з різним рівнем досвіду.

Один із основних принципів *Migrations* – це версіонування бази даних. Завдяки цьому, розробники можуть контролювати та відслідковувати зміни структури бази даних у часі. Кожна міграція має свій унікальний ідентифікатор, і при запуску механізму *Migrations* виконує лише ті міграції, які ще не були застосовані до бази даних. Це дозволяє плавно розвивати схему бази даних без проблем із синхронізацією між розробниками.

Migrations також забезпечує зручність при використанні. Завдяки

командному інтерфейсу *Artisan*, розробники можуть легко створювати нові міграції, проглядати історію застосованих міграцій, а також відміняти та перезастосовувати міграції. Це значно спрощує процес розвитку та допомагає уникнути помилок під час внесення змін у базу даних.

У ході розробки веб-серверу було проведено 20 міграцій бази даних, які збережені у файлах, що знаходяться за шляхом *database*/*migrations*.

Також серед міграцій відображуються стандартні файли *Laravel*, що мають назви:

- 2014_10_12_000000_*create_users_table*;
- 2014_10_12_100000_*create_password_reset_tokens_table*;
- 2019_08_19_000000_*create_failed_jobs_table*;
- 2019_12_14_000001_*create_personal_access_tokens_table*.

Приклад вмісту першого файлу проведеної міграції 2023_05_09_080651_*create_eco_city_data_table*, що має наступний вміст.

Обговорення результатів створення структурованої бази екологічних даних

Дослідження виконане в межах договору № 29/05 від 29.05.2023 року [24], ТОВ «ЛЕМПДЕВ» – КП «НДЦ» [25], «Адміністрування програмного забезпечення» ДК 021:2015 7226000-5 Послуги, пов'язані з програмним забезпеченням. Договір передбачає три етапи виконання науково-технічних робіт:

- розробка системи класифікації та кодування постів спостереження за станом атмосферного повітря;
- розробка та впровадження структурованої бази даних згідно системи класифікації та кодування, та реєстру постів спостереження;
- розробка, впровадження служби з автоматизованого формування звітних відомостей про стан атмосферного повітря згідно Постанови №827 КМУ [26] та тестування автоматизованого робочого місця оператора з віддаленим керуванням та налагодженням постів спостереження за станом атмосферного повітря.

Матеріали першого етапу роботи за договором описано у внутрішньому звіті. Короткі результати першого етапу наступні:

- проаналізовано нормативну базу для класифікації та кодування постів спостереження за атмосферним повітрям. Визначено, що постанова №827 КМУ від 14 серпня 2019 р. на ресурсі Верховної ради України пов'язана з Наказом №300 МВС України від 21.04.2021. У наказі визначено основні вимоги та умови розміщення пунктів спостережень за забрудненням атмосферного повітря в зонах та агломераціях, їх мінімальну кількість для проведення фіксованих вимірювань, правила кодування, документування щодо їх розміщення на території України;
- сформовано код за вимогами наведеними у Наказі №300 МВС України від 21.04.2021 [27]. Отриманий код проаналізовано та виявлено істотну недосконаліть системи кодування. Наразі код слабко репрезентує організацію-власника пункту спостереження, а також має лише умовні вказівки на географічне розміщення пункту, у вигляді двох цифр;
- удосконалено систему створення кодів для використання при створенні бази даних, виходячи з вимог визначених нормативних документах. Удосконалення системи полягає у доданні коду з 18 знаків, що відповідають за поточні координати пункту, з точністю до десятитисячної долі градуса, та абревіатуру організації-власника пункту з необмеженою довжиною.

У роботі висвітлено результати другого етапу дослідження, у ході якого створено функціональну базу даних, яка дозволяє зберігати, аналізувати та отримувати доступ до екологічних даних про стан повітря з різних станцій доступних комунальному підприємству-замовнику.

Вибір реляційної моделі бази даних з системою керування *MySQL*, модулем *PDO* та фреймворком *Laravel* дозволяють зручно адмініструвати створену базу даних мінімізуючи витрати на її розробку та

обслуговування. Система запису проведених міграцій у самій базі даних дозволяє зберігати інформацію про проведені зміни її структури та відновити попередній вигляд бази даних за необхідності. Міграції бази даних до вигляду представленого у даній роботі здійснено за вимогами замовника з урахуванням технічних та нормативних аспектів виконання третього етапу дослідження з створення служби з автоматизованого формування звітних відомостей про стан атмосферного повітря.

Висновки

У ході виконання роботи досягнуто поставленої мети з створення структурованої бази даних згідно з системою класифікації та кодування, та реєстру постів спостереження, що дозволяє зберігати великі об'єми даних про стан повітря з станцій та постів спостережень в одному просторі, швидко отримувати доступ до збережених показів, будувати графіки в межах різних проміжків часу досліджень з різними інтервалами усереднення даних.

1. У процесі роботи проаналізовано визначення та поняття пов'язані з базами даних, розглянуто класифікацію баз даних та обрано оптимальну модель бази даних для реалізації проєкту.

2. За основу бази даних обрано реляційну, однокористувацьку централізовану *SQL* модель що працює на локальних ресурсах сервера, зберігається у дисковому просторі та призначена для аналітики екологічних даних. Мовою програмування для реалізації серверного функціоналу та роботи з базою даних обрано *PHP*.

3. Також, проведено аналіз систем керування базами даних, серед яких для роботи з базою даних обрано *MySQL*, оскільки дана система володіє гарними показниками продуктивності та масштабованості, що дозволяє їй ефективно працювати з великими обсягами сирих даних.

4. Потім розроблено структуру бази даних, що містить такі таблиці: *migrations*; *failed_jobs*; *users*; *password_reset_tokens*; *personal_access_tokens*; *station*1756; *station*1748; *station*1753; *data_eco_city*_1755; *station_T*3950716, *station_V*0440346; *station_T*3950713; *split_measurement_ua*171.

5. Щоб досягти фінального вигляду запланованої структури бази даних базуючись на вихідних моделях таблиць було проведено 20 міграцій.

Проведені міграції закладають основу до подальшого удосконалення функціоналу веб-сервера та збільшення кількості станцій з яких можна отримувати дані.

Література

1. Бахарєв В.С., Перекрест А.Л., Корцова О.Л., Міхєєва П.Д. Зміни функціональної схеми оперативного контролю за забрудненням атмосферного повітря в місті з використанням громадської мережі станцій моніторингу. *Екологічно сталий розвиток урбосистем*. 2022. С. 82–85.

2. Корцова О.Л., Бахарєв В.С. Теоретико-практичні аспекти реалізації післяпроектного комплексного екологічного моніторингу в процедурі оцінки впливу на довкілля. *Вісник КрНУ імені Михайла Остроградського*. 2021. В. 6(131). С. 54–59.

3. Бахарєв В.С., Перекрест А.Л., Корцова О.Л. Оповіщення населення про стан радіаційно-техногенної безпеки з використанням можливо-стей громадської мережі моніторингу *EcoCity*. *Ольвійський форум-2023: стратегії країн Причорноморського регіону в геополітичному просторі*. 2023. С. 10–15.

4. Meier A., Kaufmann M. SQL & NoSQL Databases. Springer Fachmedien Wiesbaden, 2019. 229 p.

5. Chavan D.H., Shaikh P.S. Introduction to DBMS: Designing and Implementing Databases from Scratch for Absolute Beginners (English Edition). BPB Publications, 2022. 276 p.

6. Blokdyk G. Database Management System Technologies a Complete Guide – 2019 Edition. (n.p.). Emereo Pty Limited, 2019. 316 p.

7. PostgreSQL. PostgreSQL. URL: https://www.postgresql.org/.

8. MySQL. MySQL. URL: https://www.mysql.com/.

9. Учасники проектів Вікімедіа. Microsoft SQL Server – Вікіпедія. Вікіпедія.

URL: https://uk.wikipedia.org/wiki/Microsoft_SQL_Server.

10. Oracle LiveLabs. URL: https://oracle.github.io/learning-library/data-management-library/database/db-quickstart/workshops/livelabs/?nav=open&lab=intro.

11. SQLite Home Page. SQLite Home Page. URL: https://www.sqlite.org/index.html.

12. MariaDB Foundation – MariaDB.org. MariaDB.org. URL: https://mariadb.org/.

13. IBM Db2. IBM – Deutschland | IBM. URL: https://www.ibm.com/products/db2.

14. Home. Vaisala. URL: https://www.vaisala.com/en.

15. Postman. Postman API Platform. URL: https://www.postman.com/.

16. Laravel – The PHP Framework For Web Artisans. Laravel – The PHP Framework For Web Artisans. URL: https://laravel.com/.

17. Laravel – The PHP Framework For Web Artisans. Eloquent: Getting Started. URL: https://laravel.com/docs/10.x/eloquent.

18. PHP: PDO – Manual. PHP: Hypertext Preprocessor. URL: https://www.php.net/manual/en/book.pdo.php.

19. nginx news. nginx news. URL: https://nginx.org/.

20. Enterprise Open Source and Linux | Ubuntu. Ubuntu. URL: https://ubuntu.com/.

21. Rockefeller J.D. Web Hosting Guide for Beginners. Createspace Independent Publishing Platform, 2016. 26 p.

22. LaCroix J. Mastering Ubuntu Server: Explore the Versatile, Powerful Linux Server Distribution Ubuntu 22.04 with This Comprehensive Guide. Packt Publishing, Limited, 2022. 584 p.

23. Laravel – The PHP Framework For Web Artisans. Database: Migrations. URL: https://laravel.com/docs/10.x/migrations.

24. Договір № 29/05 від 29.05.2023 року, ТОВ «ЛЕМПДЕВ» – КП «НДЦ», «Адміністрування програмного забезпечення», ДК 021:2015 7226000-5 Послуги, пов'язані з програмним забезпеченням.

25. Комунальне підприємство «Науковий центр еколого-соціальних досліджень» Кременчуцької міської ради Кременчуцького району Полтавської області. Портал відкритих даних. Головна сторінка – Data.gov.ua. URL: https://data.gov.ua/organization/komunalne-pidpryiemstvo-naukovyi-tsentr-ekoloho-sotsialnykh-doslidzhen-mkremenchuk-poltavskoyi.

26. Деякі питання здійснення державного моніторингу в галузі охорони атмосферного повітря : Постанова Каб. Міністрів України від 14.08.2019 р. № 827 : станом на 10 листоп. 2020 р. URL: https://zakon.rada.gov.ua/laws/show/827-2019-п#Text.

27. Про затвердження Порядку розміщення пунктів спостережень за забрудненням атмосферного повітря в зонах та агломераціях : Наказ М-ва внутр. справ України від 21.04.2021 р. № 300. URL: https://zakon.rada.gov.ua/laws/show/z0635-21#Text.

Перекрест А.Л., Бахарєв В.С., Вадурін К.О., Дерієнко А.І., Іващенко А.В., Шкарупа С.А.

РОЗРОБКА БАЗИ ДАНИХ ДЛЯ ЗБЕРІГАННЯ ПОКАЗНИКІВ СТАНУ АТМОСФЕРНОГО ПОВІТРЯ З ДОСЛІДНИХ СТАНЦІЙ КОМУНАЛЬНОГО ПІДПРИЄМСТВА

Станом на сьогодні проблема забруднення атмосферного повітря стоїть гостро, особливо в Україні, де відбуваються значні зміни екологічного стану довкілля внаслідок війни. Ці зміни призвели до необхідності релокації виробничих підприємств та населення, що збільшило навантаження на організації, що займаються екологічними дослідженнями. У м. Кременчук задачі дослідження стану компонентів довкілля виконує комунальне підприємство міської ради, що має відповідне технічне оснащення.

Для проведення детальних досліджень поточного екологічного стану атмосферного повітря в межах Кременчуччини та інших зонах досліджень, накопичення знань

про тренди змін вимірюваних показників забрудненості повітря та прогнозування майбутніх змін екології, актуальною задачею є створення сукупної бази даних для автоматичного збереження показів з постів та станцій досліджень.

У даному дослідженні проаналізовано моделі баз даних та системи керування базами даних з метою обрати оптимальну модель для проєкту. Основна увага приділена проєктуванню структури бази даних, включаючи розробку таблиць та їх зв'язків, що дозволило зберігати великі обсяги даних про стан повітря в одному просторі. Для реалізації бази даних використано PHP та MySQL – систему керування базами даних з гарними показниками продуктивності та масштабованості.

Окремий аспект дослідження становить можливість розширення бази даних за рахунок додавання нових станцій та інтеграція даних з онлайн-ресурсів. У ході роботи проведено міграції бази даних, які допомагають досягти фінальної структури бази даних, що закладає основу для подальшого розвитку функціоналу та збільшення кількості джерел отримання даних.

Створена база даних стане основою для створення служби автоматичного формування звітів та прогнозів, а також дозволить дослідникам та спеціалістам здійснювати детальний аналіз стану атмосферного повітря в різних регіонах України.

***Ключові слова**: екологічний моніторинг, атмосферне повітря, база даних, MySQL, Laravel, PHP, Vaisala..*

Perekrest A.L., Bakharev V.S., Vadurin K.O., Deriyenko A.I., Ivashchenko A.V., Shkarupa S.A.

DEVELOPMENT OF A DATABASE FOR STORING ATMOSPHERIC AIR QUALITY INDICATORS FROM THE RESEARCH STATIONS OF A UTILITY COMPANY

Today, the problem of air pollution is acute, especially in Ukraine, where significant changes in the environmental situation are taking place as a result of the war. These changes have led to the need to relocate production facilities and the population, which has increased the burden on organisations involved in environmental research. In Kremenchuk, the tasks of studying the state of environmental components are performed by a municipal enterprise of the city council, which has the appropriate technical equipment.

In order to conduct detailed studies of the current ecological state of the atmospheric air within the Kremenchuk region and other research areas, to accumulate knowledge about trends in measured air pollution indicators and to predict future environmental changes, it is important to create a comprehensive database for automatic storage of readings from research posts and stations.

This study analyses database models and database management systems in order to select the optimal model for the project. The main focus is on the design of the database structure, including the development of tables and their relationships, which allowed storing large amounts of air quality data in one space. PHP and MySQL, a database management system with good performance and scalability, were used to implement the database.

A separate aspect of the study is the possibility of expanding the database by adding new stations and integrating data from online resources. In the course of the work, the database was migrated, which helped to achieve the final structure of the database, laying the foundation for further development of the functionality and increasing the number of data sources.

The created database will become the basis for the creation of an automatic report and forecast generation service, as well as allow researchers and specialists to carry out a detailed analysis of the state of atmospheric air in different regions of Ukraine.

***Keywords**: environmental monitoring, atmospheric air, database, MySQL, Laravel, PHP, Vaisala.*