МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ АЕРОНАВІГАЦІЇ,
ЕЛЕКТРОНІКИ ТА ТЕЛЕКОМУНІКАЦІЙ
КАФЕДРА ТЕЛЕКОМУНІКАЦІЙНИХ ТА РАДІОЕЛЕКТРОННИХ СИСТЕМ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____ Роман ОДАРЧЕНКО
“_____”_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВР

Тема: «Система інформаційної безпеки СУБД Oracle»

Виконавиця: Ольга ПАВЛЕНКО
(підпис)

Керівник: Денис БАХТІЯРОВ
(підпис)

Нормоконтролер: Денис БАХТІЯРОВ
(підпис)

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій
Кафедра телекомунікаційних та радіоелектронних систем
Спеціальність 172 «Телекомунікації та радіотехніка»
Освітньо-професійна програма «Телекомунікаційні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри

Роман ОДАРЧЕНКО
“_____”_____2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Павленко Ольги Євгеніївни
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема кваліфікаційної роботи: «Система інформаційної безпеки СУБД Oracle» затверджена наказом ректора від «29» березня 2023 р. № 421/ст
2. Термін виконання роботи: з 22.05.2023 р. по 25.06.2023 р.
3. Вихідні дані до роботи: система управління базами даних Oracle
4. Зміст пояснювальної записки: вибір методу забезпечення безпеки; забезпечення безпеки СУБД Oracle; розробка бази даних;
5. Перелік обов'язкового графічного (ілюстративного) матеріалу: слайди презентації кваліфікаційної роботи в програмному пакеті Microsoft Power Point

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Розробити деталізований зміст розділів кваліфікаційної роботи	22.05.2023-24.05.2023	Виконано
2	Вступ	25.05.2023	Виконано
3	Вибір методу забезпечення безпеки	26.05.2023-29.05.2023	Виконано
4	Забезпечення безпеки СУБД Oracle	30.05.2023-07.06.2023	Виконано
5	Розробка бази даних	08.06.2023-14.06.2023	Виконано
6	Усунення недоліків та захист кваліфікаційної роботи	15.06.2023-25.06.2023	Виконано

7. Дата видачі завдання: “19" травня 2023 р.

Керівник кваліфікаційної роботи _____ Денис БАХТІЯРОВ
(підпис керівника) (П.І.Б.)

Завдання прийняла до виконання _____ Ольга ПАВЛЕНКО
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Кваліфікаційна робота «Система інформаційної безпеки СУБД Oracle» містить 64 сторінки, 21 рисунок, 5 таблиць, 20 використаних джерел.

СУБД ORACLE, БЕЗПЕКА ДАНИХ, КОНТРОЛЬ ДОСТУПУ, ШИФРУВАННЯ, АУДИТ БЕЗПЕКИ, ВТОРГНЕННЯ, ЗАПОБІГАННЯ ВТОРГНЕНЬ, ЖУРНАЛИ БЕЗПЕКИ, УПРАВЛІННЯ БЕЗПЕКОЮ, ПОЛІТИКИ БЕЗПЕКИ, УЯЗВИМОСТІ, ЗАХИСТ ІНФРАСТРУКТУРИ, РИЗИК БЕЗПЕКИ, ПРАВА ДОСТУПУ, СИГНАТУРИ АТАК, МОНІТОРИНГ БЕЗПЕКИ, ІДЕНТИФІКАЦІЯ ТА АУТЕНТИФІКАЦІЯ, УПРАВЛІННЯ ПРИВІЛЕЯМИ, ЗАХИСТ ВІД ВИТОКУ ДАНИХ, ВІДНОВЛЕННЯ ПІСЛЯ ІНЦИДЕНТІВ БЕЗПЕКИ.

Метою даної кваліфікаційної роботи є забезпечення безпеки системи управління базами даних на прикладі умовної бази даних. Для забезпечення багаторівневої безпеки як на рівні бази даних, так і на рівні сервера будуть використані такі компоненти, як Прозоре шифрування даних (TDE), протокол аутентифікації Radius, аудит бази даних.

Об’єктом дослідження – є сама система інформаційної безпеки, що використовується в СУБД Oracle.

Предметом дослідження – є безпека системи управління базами даних Oracle.

Практичне значення отриманих результатів. Отримані практичні результати дослідження системи інформаційної безпеки СУБД Oracle сприяють підвищенню рівня безпеки даних, зменшенню ризиків та створенню безпечного середовища для зберігання та обробки важливої інформації. Результати дослідження можуть бути використані для розробки комплексних політик безпеки, що стосуються використання СУБД Oracle. Ці політики включають рекомендації щодо паролів, прав доступу, шифрування, аудиту та інших аспектів безпеки. Вони допоможуть організації забезпечити єдині стандарти та керувати безпекою даних в СУБД Oracle.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

СУБД - Система управління базами даних.

БД - База даних.

ІБ - Інформаційна безпека.

ПЗ - Програмне забезпечення.

ACL - Access Control List (список контролю доступу).

RBAC - Role-Based Access Control (контроль доступу на основі ролей).

AES - Advanced Encryption Standard (розширений стандарт шифрування).

SSL - Secure Sockets Layer (протокол безпечного з'єднання).

IDS - Intrusion Detection System (система виявлення вторгнень).

IPS - Intrusion Prevention System (система запобігання вторгнень).

SIEM - Security Information and Event Management (система управління інформацією про безпеку та подіями).

DLP - Data Loss Prevention (попередження втрати даних).

PKI - Public Key Infrastructure (інфраструктура публічних ключів).

CVE - Common Vulnerabilities and Exposures (загальні уразливості та вразливості).

SQLi - SQL Injection (впровадження SQL-коду).

XSS - Cross-Site Scripting (міжсайтовий скриптинг).

CSRF - Cross-Site Request Forgery (міжсайтова подія фальсифікації запиту).

OTP - One-Time Password (одноразовий пароль).

MFA - Multi-Factor Authentication (багатофакторна аутентифікація).

GDPR - General Data Protection Regulation (Загальний регламент про захист персональних даних).

ВСТУП

Будь-яка діяльність у сучасному світі включає в себе безліч завдань забезпечення безпеки та виконання законодавчих норм.

Останнім часом було багато випадків крадіжок персональних даних та шахрайських операцій з використанням інформації кредитних карт, що призвело до збитків в десятки мільйонів доларів. Захист від таких загроз потребує ефективних рішень з безпеки. Університети та медичні організації підвищили рівень безпеки даних, використовуваних для ідентифікації особистості (PII), таких як номери соціального страхування. Компанії роздрібної торгівлі в даний час працюють над забезпеченням відповідності їх інформаційних систем вимогам PCI-DSS.

Найбільший кошмар будь-якої організації: хтось вкрав запасні копії вашої бази даних. Безумовно, була побудована захищена система, зашифровані найбільш конфіденційні ресурси, розміщені сервери баз даних за міжмережевими екранами. Але злодій вибрав доступний йому спосіб: він взяв запасні копії, щоб, ймовірно, скопіювати базу даних на іншому сервері, запустити інстанцію сервера цієї бази даних, а потім повільно переглядати всі ваші дані. Захист вмісту бази даних від такого крадіжки є не лише хорошою практикою, а й вимога багатьох нормативно-правових та нормативно-технічних документів [1-20].

Актуальність теми. Тема «Система інформаційної безпеки СУБД Oracle» залишається актуальною в наш час. Станом на 2023 рік Oracle є одним з провідних постачальників систем управління базами даних (СУБД) на ринку.

Захист інформації є надзвичайно важливим завданням для будь-якої організації, особливо коли мова йде про великі обсяги даних, такі як у випадку СУБД Oracle. З огляду на постійний розвиток технологій та загроз безпеці, важливо вдосконалювати та оновлювати систему інформаційної безпеки для забезпечення захисту даних від несанкціонованого доступу, витоку інформації, а також зловживання та атак.

Oracle постійно вдосконалює свої продукти та надає різноманітні засоби для забезпечення безпеки СУБД. Це включає такі функції, як контроль доступу до даних,

шифрування, аудит, виявлення та запобігання вторгненням, механізми автентифікації та авторизації. Крім того, Oracle регулярно випускає патчі та оновлення для виправлення виявлених вразливостей і захисту від нових загроз безпеки.

Загальні принципи і практики інформаційної безпеки залишаються незмінними, незалежно від конкретної СУБД, але кожна система має свої особливості та вимоги. Тому дослідження системи інформаційної безпеки СУБД Oracle є актуальним для організацій, які використовують або планують використовувати цю платформу для зберігання своїх даних.

Проблеми безпеки даних та кіберзлочинності продовжують еволюціонувати, тому важливо підтримувати актуальні знання про захист інформації у СУБД Oracle та впроваджувати відповідні заходи безпеки.

Мета і завдання дослідження.

Метою даної кваліфікаційної роботи є забезпечення безпеки системи управління базами даних на прикладі умовної бази даних. Для забезпечення багаторівневої безпеки як на рівні бази даних, так і на рівні сервера будуть використані такі компоненти, як Прозоре шифрування даних (TDE), протокол аутентифікації Radius, аудит бази даних.

Для досягнення поставленої мети вирішуються такі наукові завдання.

1. Аналіз функціональності системи інформаційної безпеки СУБД Oracle.
2. Аналіз вразливостей та загроз безпеці.
3. Впровадження аудиту безпеки.
4. Розробка політик безпеки.
5. Аналіз сучасних тенденцій у безпеці даних.

Об'єктом дослідження – є сама система інформаційної безпеки, що використовується в СУБД Oracle.

Предметом дослідження – є безпека системи управління базами даних Oracle.

При дослідженні теми «Система інформаційної безпеки СУБД Oracle» використовувались різні ***методи дослідження***, а саме:

- Літературний огляд: огляд наукової літератури, журналів, книг та наукових статей, що стосуються безпеки даних та СУБД Oracle.

- Експериментальне дослідження: експерименти з встановлення та конфігурації СУБД Oracle з метою вивчення різних аспектів безпеки; налаштування різних функцій безпеки, тестових сценаріїв та аналіз результатів для оцінки ефективності та надійності системи інформаційної безпеки.
- Кейс-стаді: вивчення реальних випадків використання СУБД Oracle та проблеми, пов'язаних з безпекою даних; аналіз цих кейсів, виявлення недоліків та пропозиції рішення для покращення безпеки в конкретних сценаріях використання.
- Аналіз журналів безпеки: вивчення журналів безпеки, які ведуться СУБД Oracle, для виявлення потенційних проблем безпеки, аномальних дій користувачів або спроб несанкціонованого доступу; аналіз цих даних та виокремлення ключових висновків для покращення безпеки системи.

Практичне значення отриманих результатів.

Отримані практичні результати дослідження системи інформаційної безпеки СУБД Oracle сприяють підвищенню рівня безпеки даних, зменшенню ризиків та створенню безпечного середовища для зберігання та обробки важливої інформації. Результати дослідження можуть бути використані для розробки комплексних політик безпеки, що стосуються використання СУБД Oracle. Ці політики включають рекомендації щодо паролів, прав доступу, шифрування, аудиту та інших аспектів безпеки. Вони допоможуть організації забезпечити єдині стандарти та керувати безпекою даних в СУБД Oracle.

Апробація отриманих результатів. Основні положення роботи доповідалися та обговорювалися на таких конференціях:

- Науково-практична конференція «Проблеми експлуатації та захисту інформаційно-комунікаційних систем», м. Київ, 2023 р.

РОЗДІЛ 1
ВИБІР МЕТОДУ ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ

1.1. Методи забезпечення безпеки СУБД

Шифрування даних є ключовим компонентом при реалізації принципу глибокого багаторівневого захисту, а також важливим елементом захисту даних під час передачі та зберігання. Вперше Oracle представив базу даних з програмним інтерфейсом (API) для шифрування даних в Oracle8i. На даний момент багато клієнтів використовують інтерфейси шифрування в базі даних Oracle для підвищення безпеки конфіденційних даних додатків. Досягнення прозорості читання/запису зашифрованих даних за допомогою використання крипто-API потребує впровадження функції викликів всередині самого додатка або використання попередньо встановленого тригера БД.

Представлення даних додатку також можуть потребувати розшифрування, перш ніж вони будуть передані до додатку. Крім того, керування ключами шифрування повинно проводитися програмним шляхом [1].

Oracle Advanced Security Transparent Data Encryption (TDE), вперше представлена в Oracle Database 10g Release 2, є найбільш передовим рішенням в галузі шифрування. TDE забезпечує вбудоване керування ключами шифрування та повну прозорість шифрування конфіденційних даних. Використаний при цьому механізм шифрування баз даних ґрунтується на використанні команд DDL, повністю виключаючи потребу в зміні додатків, створенні програмного забезпечення керування ключами шифрування, тригерів та представлень в базах даних [1].

Короткий опис Oracle Database Encryption представлено в таблиці 1.1.

Таблиця 1.1

Короткий опис Oracle Database Encryption

Комплекс функцій	Набір програм DBMS Obfuscation (Oracle 8i і вище) SE & EE	DBMS CRYPTO (Oracle database 10g R1 і вище) SE & EE	Oracle advanced security transparent data encryption EE Only Option
Алгоритми шифрування	DES, 3DES	DES, DES, AES, RC4,3DES_2KEY (1)	3DES, AES (128, 192, і 256 біт)
Додаткові форми	Не підтримується	PKCS5, нулі	PKCS5(2)
Режими з'єднання на основі блочного шифру	CBC	CBC, CFB, ECB, OFB	CBC (2)
Комплекс функцій	Набор программ DBMS Obfuscation(Oracle 8i і вище)SE & EE	DBMS CRYPTO (Oracle database 10gR1 і вище) SE & EE	Oracle advancedsecurity transparent data encryption EE Only Option
Режими з'єднання на основі блочного шифру	CBC	CBC, CFB, ECB,OFB	CBC (2)
Шифрувальні хеш-алгоритми	MD5	SHA-1, MD4(1), MD5(1)	SHA-1(2)
Хеш-алгоритмипо ключу (MAC)	none supported	HMAC_MD 5, HMAC_SH1	Не застосовуються
Шифрувальний псевдовипадковий генератор чисел	RAW, VARCHAR2	RAW, NUMBER, BINARY_INTEGE R	Не застосовуються
Типи баз даних	RAW, VARCHAR2	RAW, CLOB, BLOB	Все кроме: OBJ., ADT, LOB

Oracle Advanced Security (OAS) забезпечує конфіденційність та захист інформації.

Oracle Advanced Security складається з трьох основних частин:

- шифрування трафіку від клієнта до сервера додатків (SSL);
- шифрування трафіку від сервера додатків до бази даних (SQL Net);
- прозоре шифрування даних на диску (Transparent Data Encryption).

Усі з'єднання СУБД Oracle можуть бути зашифровані за допомогою OAS. Oracle Advanced Security виконує вимоги стандарту ISO 27001 у розділах A.10.8.4, A.10.9.2, A.11.4.2, A.12.3 [1].

OAS дозволяє захищати всі вхідні та вихідні з'єднання СУБД Oracle. Для кожного з'єднання створюється секретний ключ, що забезпечує безпеку всього мережевого трафіку. OAS робить неможливим скриту модифікацію, додавання або видалення частини передаваних даних [1].

Підтримуються такі алгоритми шифрування та забезпечення цілісності даних:

- AES (128, 192, 256).
- RC4 (40, 56, 128, 256).
- 3DES (2 і 3 ключа).
- MD5.
- SHA1.

Крім того, якщо в інформаційній системі підтримується інфраструктура відкритого ключа PKI, доступне шифрування SSL. Проста конфігурація дозволяє не змінювати додаток. Увімкнення шифрування та забезпечення цілісності даних виконується шляхом налаштування мережевих конфігурацій на стороні сервера та клієнта. Оскільки немає потреби змінювати додаток, ці методи забезпечення безпеки доступні практично для всіх, а їх конфігурування та використання дуже прості [1].

Сувора аутентифікація для Oracle Database 11g OAS забезпечує можливість для організації використовувати існуючу інфраструктуру безпеки, наприклад, Kerberos, PKI, RADIUS для здійснення суворої аутентифікації в СУБД Oracle 11g. Зв'язок протоколів аутентифікації та Oracle Advanced Security представлений на рисунку 1.1.

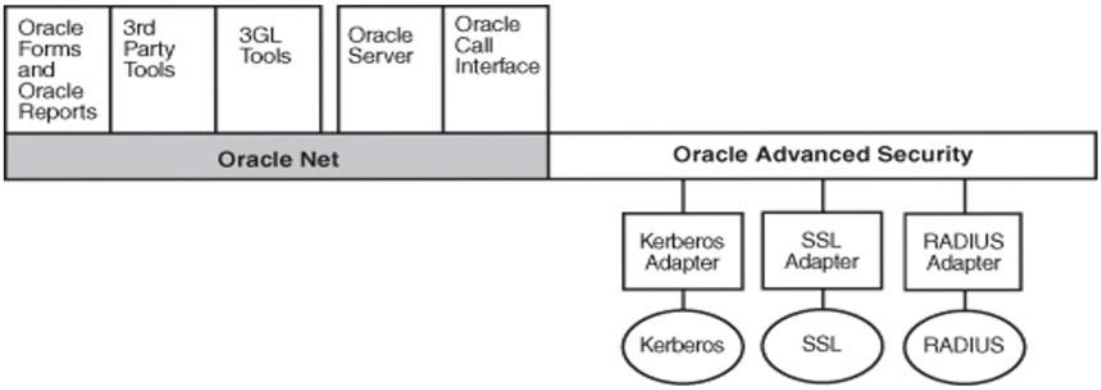

Рис. 1.1. Oracle Net Service з протоколами аутентицікації

1.2. Прозоре шифрування даних

За допомогою TDE забезпечується шифрування даних перед записом на диск та дешифрування даних до їх повернення до додатку. Процес шифрування та дешифрування виконується на рівні SQL та повністю прозорий для застосункових програм та користувачів. Резервні копії баз даних, записані на диск або магнітну стрічку, міститимуть ці дані в зашифрованому вигляді. TDE, за необхідності, може бути використано в поєднанні з Oracle RMAN для шифрування всієї СУБД Oracle під час резервного копіювання на диски [1].

Переваги прозорого шифрування даних [2]:

- вбудоване керування ключами шифрування;
- прозоре шифрування захищених даних (по стовпцях) в прикладних програмах;
- прозоре шифрування таблицевих просторів (вперше в 11g);
- прозоре шифрування файлів/LOBS (вперше в 11g);
- інтеграція апаратного модуля безпеки (HSM) (вперше в 11g).

1.2.1. Короткий опис управління ключами шифрування

TDE автоматично створює ключ шифрування при шифруванні даних колонки в таблиці бази даних. Кожна таблиця має унікальний ключ шифрування. Якщо в таблиці шифрується більше однієї колонки, то для кожної з них використовується той

самий ключ шифрування. Ключі шифрування для таблиць зберігаються в довіднику Oracle та шифруються за допомогою первинного ключа (майстер-ключа) шифрування TDE. Первинний ключ шифрування зберігається поза базою даних у «конверті для ключів» Oracle Wallet (файл формату PKCS#12), який шифрується за допомогою паролю, що встановлюється адміністратором безпеки або DBA під час створення.

Новинкою в Oracle Database 11g Advanced Security є можливість зберігання первинного ключа у пристрої HSM з використанням інтерфейсу PKCS#11 [1].

Короткий опис прозорого шифрування даних представлений на рисунку 1.2.

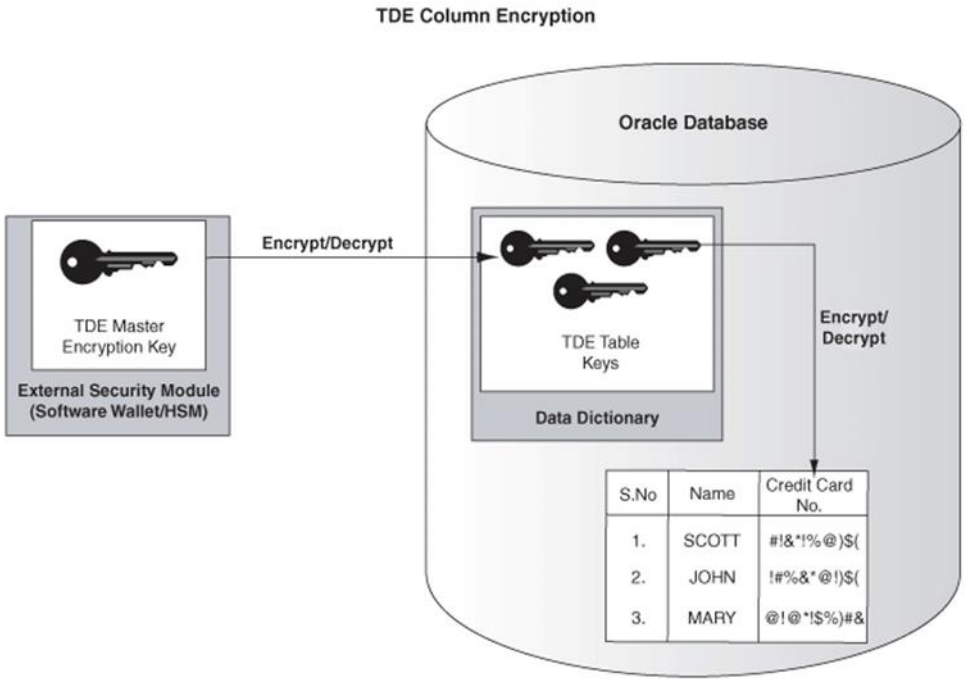

Рис. 1.2. Короткий опис прозорого шифрування даних

1.2.2. Етапи впровадження

У зв'язку з тим, що TDE є прозорим для існуючих додатків (не потрібно створювати тригери та представлення баз даних), процес шифрування став простішим порівняно з традиційними рішеннями на основі використання API. Наступні кроки можуть бути використані для застосування TDE [2]:

1) ініціалізація майстер-ключа;
2) визначення даних для шифрування (PII дані, кредитні картки);

3) перевірка: чи підтримує TDE вибраний тип даних, а також, чи не використовуються дані колонки як зовнішні ключі;

4) шифрування конфіденційних даних за допомогою TDE.

Ініціалізація майстер-ключа. Майстер-ключ для кожної бази даних свій. Незважаючи на це, будь-який майстер-ключ може бути скопійований у вторинну базу даних, якщо раніше він там не був використаний. Перш ніж зашифровувати дані таблиці, необхідно створити майстер-ключ. Для цього існує наступна команда:

```
SQL> alter system set key identified by "password";
```

Ця команда створює "тубус для ключів" (Oracle Wallet) та використовує вказаний у команді пароль для шифрування згідно зі стандартом PKCS#5. Oracle Wallet зберігає історію вже використаних майстер-ключів та робить їх доступними, коли дані, зашифровані старим ключем, зчитуються з резервних носіїв.

Перш ніж база даних зможе розшифрувати ключі таблиць для шифрування та розшифрування прикладних даних, потрібно відкрити "тубус для ключів" Wallet, який містить первинний ключ шифрування. Звичайно, можна запустити базу даних та працювати в ній без відкриття Wallet, проте при спробі доступу до зашифрованих даних база даних видаватиме повідомлення про помилку. Закривати Wallet (та обмежувати таким чином доступ до даних) слід тільки під час технічного обслуговування, коли доступ до бази даних дозволено спеціалісту, який займається підтримкою.

Майстер-ключ може бути змінений при повторному запуску команди "alter system".

```
SQL> alter system set key identified by "password";
```

При зміні майстер-ключа всі табличні ключі в каталозі Oracle будуть оновлені (розшифровані та зашифровані знову). Стандарт безпеки даних PCI (DSS) 1.1 вимагає "оновлювати ключі шифрування, не рідше одного разу на рік". Зміна майстер-ключа

призведе до оновлення зашифрованих ключів стовпців, використовуючи новий майстер-ключ, а зашифровані дані в таблицях залишаться без змін [2].

Пароль Wallet може бути змінений незалежно від майстер-ключа шифрування, він використовується виключно для шифрування даних файлу Wallet на диску. Для цього можна скористатися програмою Oracle Wallet Manager або утилітою 'orapki'.

Визначення конфіденційних даних. Процес пошуку конфіденційних даних, таких як номери соціального страхування та кредитних карток, може виявитися скрутним, особливо в комплексних додатках. Єдиний метод, яким можна скористатися, – це пошук у довіднику Oracle за назвою стовпця та типом даних, які часто використовуються для зберігання такої інформації.

```
SQL> select column_name, table_name, data_type from
dba_tab_cols where column_name like '%SOCIAL%' or column_name
like '%SSN%' or column_name like '%SECNUM%' or column_name
like "%SOC%' and owner='<owner>';
```

Типи даних, що підтримуються TDE

TDE підтримує найпоширеніші типи даних. Вони представлені у таблиці 1.2 [2].

Таблиця 1.2

Типи даних, що підтримуються TDE

VARCHAR2	CHAR	DATE
NUMBER	NVARCHAR2	NCHAR
RAW	RAW	SECUREFILES (LOBS)
BINARY_DOUBLE	BINARY_FLOAT	

TDE не може бути використаний для шифрування стовпців, які використовуються у зовнішньому ключі. Щоб перевірити, чи є колонка частиною зовнішнього ключа, вивчаються дані з довідника Oracle.

Шифрування даних за допомогою TDE. Щоб зашифрувати існуючі стовпці у таблиці [2]:

```
SQL> alter table figure_skater modify ( fs_name encrypt);
```

У ході виконання транзакції із шифрування підтримується послідовне читання даних.

Транзакції DML (вставити, оновити, видалити), які виконуються протягом транзакції зашифровування, будуть вимагати 'зміни online'.

Transparent Data Encryption підтримує роботу з індексами, мінімізуючи додатковий пошук у зашифрованому стовпці.

Коли стовпець з індексами готовий до шифрування, спочатку рекомендується видалити існуючі індекси, зашифрувати стовпець, а потім заново створити індекси.

Зміна ключа таблиці/стовпця. При виконанні команди "alter table" ключ таблиці або стовпця, розмір ключа та алгоритм можуть бути незалежно змінені [2]:

```
SQL> ALTER TABLE judge REKEY;
SQL> ALTER TABLE judge REKEY USING 'AES256'; SQL> ALTER
TABLE judge ENCRYPT USING 'AES128';
```

При зміні ключа таблиці або стовпця буде проведено повторне шифрування даних в таблиці.

Протягом часу існування таблиці її дані можуть фрагментуватися, розміщуватися в іншому порядку, розділятися за категоріями, збільшуватися в кількості та переміщуватися в межах табличного простору. Це може призвести до виникнення не використовуваних блоків зі старими, недоступними копіями даних в межах файлу бази даних. При шифруванні існуючого стовпця шифрується лише його найновіша "діюча" копія, при цьому дані можуть залишатися у незашифрованому вигляді в "старих" копіях даних стовпця [2].

Для мінімізації ризику залишити конфіденційні дані у незашифрованому вигляді при шифруванні, Oracle рекомендує створювати новий табличний простір, переміщувати таблиці додатків в нього та видаляти старий табличний простір.

1.3. Протокол автентифікації RADIUS

Oracle Advanced Security має протокол RADIUS, який дозволяє Oracle DataBase 11 дотримуватись аутентифікації та повноважень, затверджених на сервері RADIUS. Ця функція особливо корисна для компаній, які зацікавлені в двухфакторній аутентифікації, що встановлює особу на основі того, що вона знає (пароль або ПІН-код) та тим, чим вона володіє (карта зі змінним паролем), надану деяким виробником карт зі змінним паролем. RADIUS (RFC # 2138) це поширена система, яка забезпечує віддалений доступ до мережевих служб і давно визнана як галузевий стандарт для віддаленого та контрольованого доступу до мережі. Повноваження користувача в RADIUS та інформація про доступ визначені на сервері RADIUS для того, щоб дати можливість цьому зовнішньому серверу виконати аутентифікацію, авторизацію та облік, коли це потрібно [3].

RADIUS використовується як протокол AAA [3]:

- англ. Authentication - процес, що дозволяє аутентифікувати (перевірити справжність) суб'єкта за його ідентифікаційними даними, наприклад, за логіном (ім'я користувача, номер телефону тощо) та паролю;
- англ. Authorization - процес, що визначає повноваження ідентифікованого суб'єкта на доступ до певних об'єктів чи сервісів;
- англ. Accounting - процес, що дозволяє вести збирання відомостей (облікових даних) про використані ресурси. Первинними даними (тобто традиційно передаються за протоколом RADIUS) є величини вхідного та вихідного трафіку: в байтах/октетах (з недавніх пір в гігабайтах). Проте протокол передбачає передачу даних будь-якого типу, що реалізується за допомогою VSA (Vendor Specific Attributes).

Робота Radius у середовищі Oracle показано на рисунку 1.3.

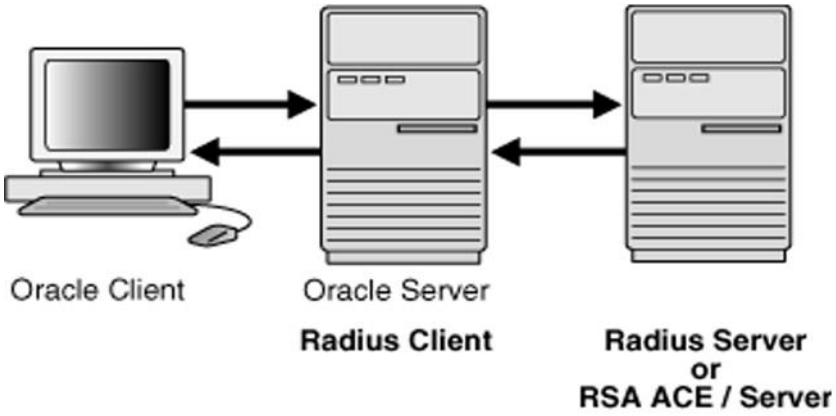

Рис. 1.3. Робота Radius в середовищі Oracle

Підтримка RADIUS в Oracle - це реалізація протоколів клієнта RADIUS (RADIUS Client), які дають можливість базі даних надавати користувачам RADIUS аутентифікацію, авторизацію та сервіси обліку. Вони надсилають запит аутентифікації на сервер RADIUS та діють відповідно до відповідей сервера. Аутентифікація може відбуватись в синхронному або асинхронному режимі аутентифікації та є частиною конфігурації Oracle для підтримки RADIUS [3].

Синхронний режим аутентифікації показаний на рис. 1.4 і проходить наступною послідовністю [3]:

1) користувач авторизується за допомогою коду (паролю) чи інших ідентифікаційних даних. Клієнтська система надсилає ці дані на сервер Oracle;

2) сервер Oracle, виступаючи в ролі RADIUS client, передає дані з клієнта Oracle на Сервер Radius;

3) сервер Radius, у свою чергу, передає дані на відповідний аутентифікаційний сервер, наприклад, Smart Card або SecurID ACE для підтвердження;

4) аутентифікаційний сервер надсилає або дозвіл, або забороняє доступ лист назад, на Сервер Radius; client;

5) сервер Radius перенаправляє цю відповідь на Сервер Oracle/RADIUS

6) сервер Oracle/RADIUS client перенаправляє відповідь назад, клієнту Oracle.

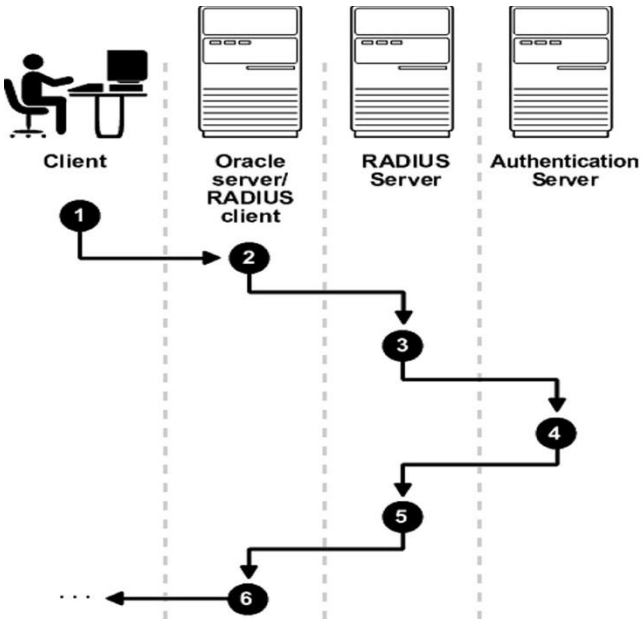

Рис. 1.4. Послідовність синхронної автентифікації

Асинхронний режим аутентифікації показаний на рис. 1.5 і проходить наступною послідовністю [3]:

1) користувач надсилає запит на вхід. Клієнтська система надсилає ці дані на сервер Oracle;

2) сервер Oracle, виступаючи в ролі RADIUS client, передає дані з клієнта Oracle на Сервер Radius;

3) сервер Radius, у свою чергу, передає дані на відповідний автентифікаційний сервер, наприклад, Smart Card або SecurID ACE;

4) автентифікаційний сервер надсилає запит на сервер Radius, який є випадковим набором чисел;

5) сервер Radius надсилає цей запит на Сервер Oracle/RADIUS client;

6) сервер Oracle/RADIUS client, своєю чергою, відправляє його клієнту Oracle;

7) користувач повинен ввести цей запит у запропоноване поле, далі клієнт Oracle надішле запит на сервер Oracle/RADIUS client;

8) сервер Oracle/RADIUS client надсилає відповідь користувача на сервер Radius;

9) сервер RADIUS надсилає відповідь користувача на автентифікаційний сервер для підтвердження;

10) автентифікаційний сервер надсилає або дозвіл, або забороняє доступ лист назад, на Сервер Radius;

11) сервер Radius перенаправляє цю відповідь на Сервер Oracle/RADIUS client;

12) сервер Oracle/RADIUS client перенаправляє відповідь назад, клієнту Oracle.

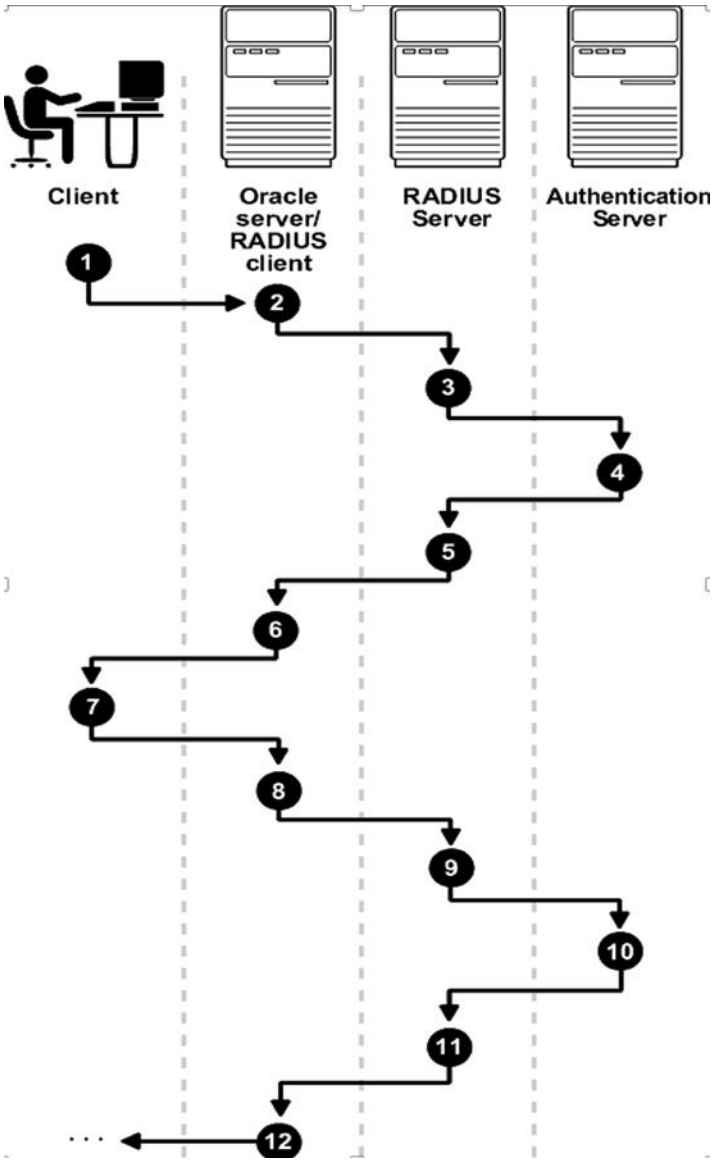

Рис. 1.5. Послідовність асинхронної автентифікації

Oracle Advanced Security надає користувачам RADIUS автентифікацію, дозволи авторизації, збережені в RADIUS та основні служби обліку при зверненні до бази даних Oracle.

РОЗДІЛ 2
ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ СУБД ORACLE

2.1. Прозоре шифрування даних

Найбільший кошмар будь-якої організації - коли хтось вкрав стрічки з резервною копією вашої бази даних. Безумовно, була побудована захищена система, зашифровані найконфіденційніші ресурси, розміщені сервери баз даних за мережевими екранами. Але злодій вибрав доступний йому спосіб: він взяв стрічки з резервною копією, щоб, ймовірно, скопіювати базу даних на інший сервер, запустити інстанс сервера цієї бази даних, а потім не поспішаючи переглянути всі ваші дані. Захист вмісту бази даних від такого крадіжки є не тільки хорошою практикою; це - вимога багатьох нормативно-правових та нормативно-технічних документів [3-4].

Один з варіантів полягає в тому, щоб зашифрувати в базі даних конфіденційні дані та зберігати ключі шифрування в іншому місці; без цих ключів будь-які викрадені дані не матимуть жодного значення. Проте потрібно знайти баланс між двома протилежними поняттями: зручність доступу додатків до ключів шифрування та захист, необхідний для запобігання викраденню цих ключів. При цьому дотримання корпоративних та державних нормативних вимог передбачає негайне прийняття рішення, без використання якого-небудь складного кодування [4].

Нова функціональна можливість сервера Oracle Advanced Security дозволяє зробити наступне: можна оголосити стовпець шифрованим, не написавши при цьому жодного рядка коду додатка. Коли користувачі вставляють дані, сервер бази даних прозоро шифрує ці дані та зберігає їх у стовпці. Точно так само, коли користувачі вибирають цей стовпець, сервер бази даних автоматично розшифровує його. Оскільки все це робиться прозоро без будь-якої зміни коду додатка, ця функціональна можливість має відповідну назву: прозоре шифрування даних (TDE, Transparent Data Encryption).

Все, що необхідно зробити для шифрування - визначити стовпець, який буде шифруватись, і сервер Oracle Database 11g створить криптографічно стійкий ключ

шифрування для таблиці, що містить цей стовпець, та зашифрує дані звичайного тексту у цьому стовпці, використовуючи вказаний вами алгоритм шифрування. Захист цього ключа таблиці має дуже важливе значення; сервер Oracle Database 11g шифрує його, використовуючи головний ключ, який зберігається в безпечному місці, що називається гаманцем (wallet), який може бути файлом сервера бази даних. Зашифровані ключі таблиць розміщуються в словнику даних [4]. Коли користувач вставляє дані у стовпець, визначений як зашифрований, сервер Oracle Database 11g витягує з гаманця головний ключ, розшифровує ключ шифрування для цієї таблиці, що знаходиться в словнику даних, використовує цей ключ для шифрування вхідного значення та зберігає зашифровані дані в базі даних, як показано на рисунку 2.1.

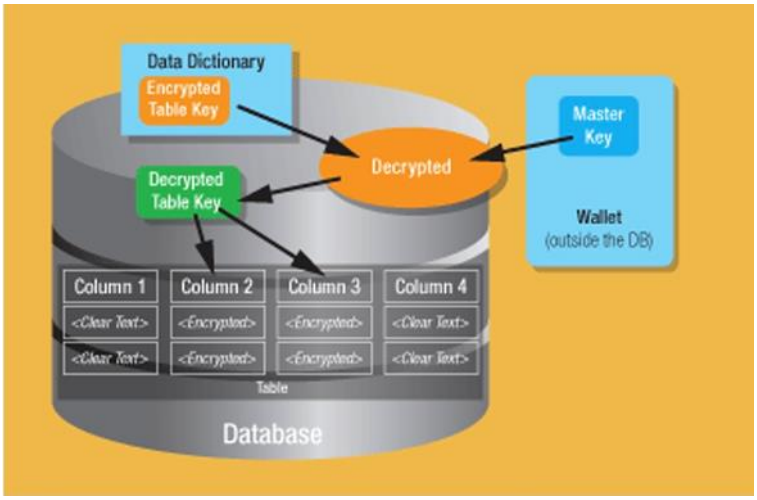

Рис. 2.1. Робота механізму прозорого шифрування даних

Написи на рисунку [4]:

- Data Dictionary – словник даних.
- Encrypted Table Key – зашифрований ключ таблиці.
- Master Key - головний ключ.
- Decrypted - розшифрований.
- Wallet (outside the DB) - гаманець (за межами бази даних).
- Decrypted Table Key -розшифрований ключ таблиці.
- Column - стовбець.
- Clear Text – звичайний текст.

- Encrypted - зашифрований.
- Table - таблиця.
- Database – база даних.

Коли користувач вибирає зашифровані стовпці, сервер Oracle Database 11g прозоро витягує з словника даних зашифрований ключ таблиці, а головний ключ - з гаманця і розшифровує ключ таблиці. Потім сервер бази даних розшифровує зашифровані на диску дані і повертає користувачеві звичайний текст.

Завдяки такому шифруванню, якщо дані будуть вкрадені з диска, вони не можуть бути витягнуті без головного ключа, який знаходиться в гаманці, що не входить до вкрадених даних. Навіть якщо вкрадений і гаманець, головний ключ не може бути витягнутий з нього без знання пароля гаманця. Отже, злодій не зможе розшифрувати дані, навіть якщо він вкрав диски або копії файлів даних. Це відповідає вимогам відповідності багатьом нормативним і керівним документам. І все це було зроблено без зміни додатка або написання складної системи шифрування та управління ключами [4].

Перед тим, як почати використовувати можливості TDE, ви повинні визначити місцезнаходження гаманця, встановити його пароль та відкрити гаманець.

2.1.1. Визначення розташування гаманця

Перед увімкненням можливостей TDE необхідно створити гаманець, в якому зберігатиметься головний ключ. За замовчуванням гаманець створюється у каталозі `$ORACLE_BASE/admin/$ORACLE_SID/wallet.` Так, якщо `$ORACLE_BASE – /u01/app/oracle, а $ORACLE_SID – SWBT4`, то гаманець зберігатиметься в каталозі `/u01/app/oracle/admin/swbt4/wallet`. Також можна вибрати інший каталог, вказуючи його у файлі sqlnet.ora, який знаходиться в каталозі `$ORACLE_HOME/network/admin`.

2.1.2. Створення гаманця

Тепер необхідно створити гаманець та встановити пароль для доступу до нього. Для цього як користувач із привілеєм ALTER SYSTEM виконайте наступний оператор:

```
alter system set encryption key authenticated by “wrl”;
```

Цей оператор:

- створює гаманець у каталозі, визначений у пункті 2.1.1;
- встановлює пароль гаманця - `"wrl"`;
- відкриває гаманець для зберігання та вилучення головного ключа засобами TDE.

Пароль залежить від регістру, і його необхідно укладати у подвійні лапки. Зауважимо, пароль "wrl" не відображається у вигляді звичайного тексту в жодних динамічних уявленнях продуктивності або журнальних файлах.

Гаманець створюється лише один раз, тому більше не потрібно повторювати два попередні кроки. Проте після запуску екземпляра сервера бази даних гаманець необхідно відкривати явно. Коли створюється гаманець (як вище у пункті 2.1.2), відкривається гаманець для роботи з ним. Після створення гаманця та встановлення пароля кожного разу, коли відкривається база даних, необхідно відкривати і гаманець, використовуючи його пароль:

```
alter system set encryption wallet open authenticated by
“wrl”;
```

Закривати гаманець необхідно, використовуючи оператор:

```
alter system set encryption wallet close;
```

Щоб засоби TDE працювали, гаманець має бути відкритим. В іншому випадку доступ буде отриманий тільки до всіх незашифрованих стовпців, але не до зашифрованих.

Щоб шифрувати стовпці, використовуючи засоби TDE, все, що необхідно зробити, це додати до визначення стовпців просту пропозицію - ENCRYPT. А до цього необхідно, який буде використовуватися тип шифрування та довжина ключа.

```
SQL> CREATE TABLE Judge(Judge_ID number Primary Key,
Judge_NAME varchar2(20) not null, Judge_SURNAME varchar2(30),
Judge_GENDER number not null, Judge_DATE_OF_BIRTH date not
null);
```

В даний час всі дані таблиці зберігаються у вигляді звичайного тексту. Необхідно перетворити стовпці `judge_name, judge_surname, judge_gender,` щоб вони зберігалися у зашифрованому вигляді. Для цього необхідно виконати оператори:

```
alter table judge modify (judge_name encrypt); alter
table judge modify (judge_surname encrypt);
alter table judge modify (judge_date_of_birth encrypt);
```

Цей оператор робить дві речі:

- створює ключ шифрування для таблиці;
- перетворює всі значення стовпців у зашифрований формат.

Цей оператор не змінює тип даних або розмір стовпця, і він також не створює жодних тригерів чи уявлень.

За замовчуванням для шифрування використовується алгоритм AES (Advanced Encryption Standard, удосконалений стандарт шифрування) зі 192-бітовим ключем. Можна вибрати інший алгоритм, вказуючи в операторі відповідну додаткову пропозицію [4].

Наприклад, щоб використовувати 128-бітове шифрування за алгоритмом AES, ви можете виконати оператор:

```
alter table judge modify (judge_name encrypt using
'AES128');
```

Також можна використовувати пропозиції AES128, AES192, AES256 або 3DES168 (168-bit Triple DES, триразове застосування алгоритму DES (Data Encryption Standard, стандарт шифрування даних) зі 168-бітовим ключем).

Після шифрування стовпця в описі таблиці з'явиться таке, як показано на рисунку 2.2:

```
SQL> desc judge
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 JUDGE_ID                                  NOT NULL NUMBER
 JUDGE_NAME                                NOT NULL VARCHAR2(20) ENCRYPT
 JUDGE_SURNAME                                      VARCHAR2(30) ENCRYPT
 JUDGE_GENDER                              NOT NULL NUMBER
 JUDGE DATE OF BIRTH                       NOT NULL DATE ENCRYPT
```

Рис. 2.2. Опис таблиці Judge після шифрування

Зверніть увагу на ключове слово `ENCRYPT`, вказане після типу даних. Для пошуку бази даних зашифрованих стовпців можна скористатися уявленням словника даних `DBA_ENCRYPTED_COLUMNS`. (Засоби TDE не можна застосовувати до таблиць схеми SYS.)

2.1.3. Питання продуктивності

Операції шифрування та дешифрування споживають час центрального процесора, тому необхідно розглянути їхній вплив на продуктивність. Коли відбувається звернення до незашифрованих стовпців таблиці, продуктивність не відрізняється від продуктивності при роботі з таблицями, для яких не використовуються засоби TDE. Коли ви звертаєтесь до зашифрованих стовпців, виникають невеликі накладні витрати на дешифрування під час вибірки даних та шифрування під час вставки, так що ви

можете захотіти шифрувати стовпці вибірково. Якщо вам більше не потрібно шифрувати стовпець, ви можете вимкнути шифрування таким чином [5]:

```
alter table judge modify (judge_name decrypt);
```

2.2. Налаштування Radius

Для встановлення та налаштування Radius виконується наступна послідовність дій [5]:

Крок 1. Установка Radius.

Крок 2.А. Конфігурація Radius на Oracle Client

1) Запуск Oracle Net Manager.

Пуск, Програми, Oracle - HOME_NAME, Configuration and Migration Tools, потім Net Manager.

2) Відкрити вкладку Oracle Net Configuration і з вкладки Local вибрати Profile.

3) З меню Naming вибрати Oracle Advanced Security. З'явиться вікно налаштувань Oracle Advanced Security, представлене рисунку 2.3.

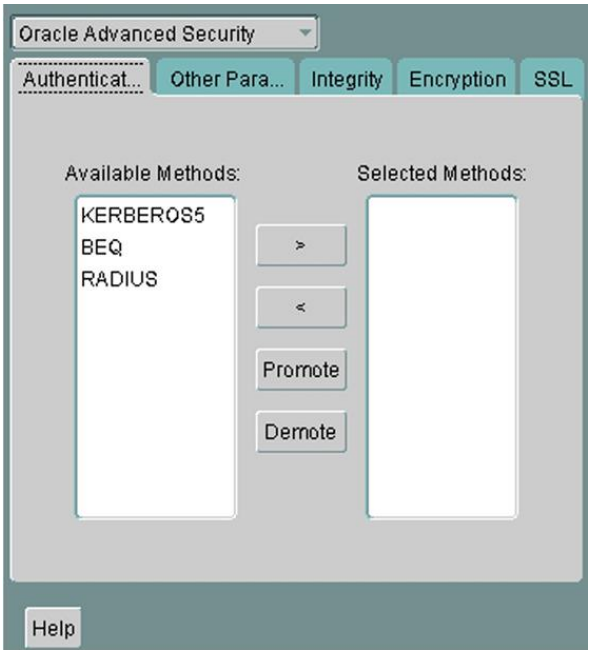

Рис. 2.3. Вікно налаштувань Oracle Advanced Security

Вкладка Authentication

4) Вибрати вкладку Authentication (вона має бути обрана за замовчуванням).

5) З доступних методів вибрати Radius.

6) Зберегти зміни.

Файл `sqlnet.ora` повинен оновитися, додати наступний рядок коду:

```
SQLNET.AUTHENTICATION_SERVICES=(RADIUS)
```

Крок 2.Б. Конфігурація Radius на Oracle Сервер.

Виконується та ж послідовність дій (1-6), що й у кроці 2.А, тільки на сервері Oracle.

7) Далі, у тому ж вікні вибирається вкладка Other Params (рисунок 2.4);

8) З падаючого меню Authentication Service вибрати Radius;

Файл `sqlnet.ora` повинен оновитися, додати наступні рядки коду:

```
SQLNET.AUTHENTICATION_SERVICES=RADIUS
SQLNET.RADIUS_AUTHENTICATION=RADIUS_server_{hostname|IP_address}
```

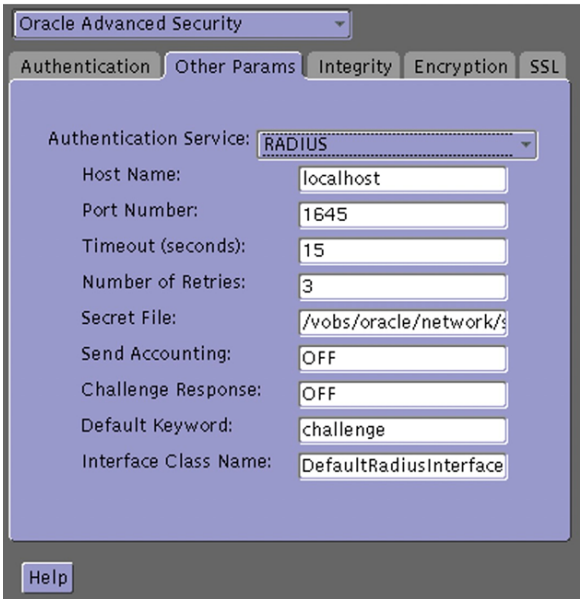

Рис. 2.4. Вікно налаштувань Oracle Advanced Security

Для налаштування параметрів ініціалізації Oracle Database Server необхідно додати наступні рядки до файлу `init.ora`:

```
OS_AUTHENT_PREFIX=""
```

За промовчанням файл init.ora знаходиться в папці `ORACLE_HOME/dbs` в ОС Linux і в папці `ORACLE_HOME\database` в Windows.

Після цього потрібно перезавантажити базу даних, використовуючи, наприклад, наступний код:

```
SQL> SHUTDOWN SQL> STARTUP
```

Крок 2. Конфігурація додаткових опцій Radius

Для зміни параметрів за замовчуванням необхідно виконати пункти 1-3 кроку 2. А. Далі:

4) Вибрати вкладку Other Params, як показано на рисунку 2.5.

5) З падаючого меню Authentication Service вибрати Radius.

6) Змінити параметри за замовчуванням по одному з полів, наприклад:

а) Port Number: задає порт прослуховування основного сервера RADIUS. Значення за замовчуванням 1645.

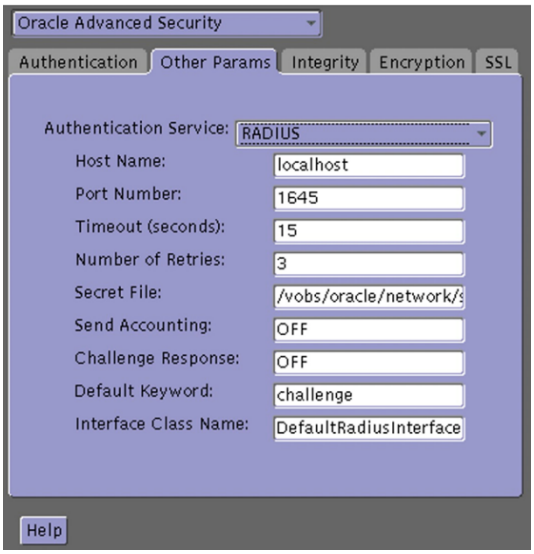

Рис. 2.5. Вікно налаштувань Oracle Advanced Security.

Вкладка Other Params

б) Timeout (seconds): Визначає час, протягом якого сервер бази даних Oracle очікує на відповідь від основного сервера RADIUS. Значення за замовчуванням 15 секунд.

в) Number of Retries: Визначає кількість разів, які сервер бази даних Oracle намагається доставити повідомлення на основний сервер RADIUS. Значення за замовчуванням 3 спроби.

7) Зберегти усі зміни.

Файл `sqlnet.ora` повинен оновитися, додати наступні рядки коду:

```
SQLNET.RADIUS_AUTHENTICATION_PORT=(PORT)
SQLNET.RADIUS_AUTHENTICATION_TIMEOUT= (NUMBER OF SECONDS TO
WAIT FOR response) SQLNET.RADIUS_AUTHENTICATION_RETRIES=
(NUMBER OF TIMES TO RE-SEND TO RADIUS server) SQLNET.
RADIUS_SECRET=(path/radius.key)
```

Крок 3. Створення користувача та надання доступу.

Для створення користувача та надання йому прав доступу необхідно:

1) Запустити SQL*Plus на сервері Radius та виконати наступну послідовність команд:

```
SQL> CONNECT system@database_name; SQL> Enter password:
SQL> CREATE USER username IDENTIFIED EXTERNALLY; SQL>
GRANT CREATE SESSION TO USER username;
SQL> EXIT
```

Цей користувач автоматично створюється на сервері бази даних Oracle.

Крок 4. Конфігурація авторизації Radius.

Для того, щоб користувач міг підключитися до сервера бази даних Oracle, необхідно додати наступне:

1) Додати параметр `OS_ROLE` у файл `init.ora` і переключити його на стан `TRUE`:

```
OS_ROLE=TRUE
```

2) Перезапустити базу даних, щоб система могла застосувати зміни у файлі `init.ora`. Команди для перезапуску можуть мати такий вигляд:

```
SQL> SHUTDOWN SQL> STARTUP
```

2.3. Аудит бази даних

Аудит Oracle може допомогти у визначенні неавторизованого доступу або внутрішнього зловживання по відношенню до інформації, що міститься в базі даних.

Простий набір основних дій аудиту повинен бути завжди активним. Мінімальний набір включає в себе відстеження доступу користувачів, використання системних привілеїв та зміну структури бази даних. Цей основний набір не покаже невдалі спроби доступу до конкретних даних, які не повинні бути доступними; проте, він забезпечить досить простий огляд "некоректного" доступу або використання привілеїв. Якщо співробітника підозрюють у недозволених діях або очікується атака, то може бути застосований більш деталізований аудит для конкретних таблиць. З точки зору управління БД, аудит зміни даних для всіх таблиць не є практичним і може вплинути на продуктивність системи в цілому. Аудит доступу для зміни даних повинен використовуватись для таблиць лише з особливо важливою інформацією (наприклад, заробітна плата співробітників в базі даних HR). Стандартні команди аудиту дозволяють контролювати всі системні та об'єктні привілеї доступу до будь-яких таблиць або представлень бази даних на select, delete, insert or update. Аудит може бути запущений

як для успішних, так і для неуспішних спроб або для тих і інших одночасно. Як для кожного користувача окремо, так і для всіх користувачів одночасно, він може виконуватись на рівні сесії або на рівні дії (доступу). На рівні дії - один запис створюється для однієї дії, а на сесійному - один запис для всіх контрольованих операцій однієї сесії [5].

Часто аудит сприймається як складний і повільний процес. Причина цьому - звичайна невігласність. Якщо включені більшість усіх опцій, то результатом є журнал аудиту, який може бути великим і складним для інтерпретації та управління. Крім того, якщо аудит задіяний на всіх таблицях та представленнях бази, то це може вплинути на продуктивність. Кожного разу, коли дія, яку контролює аудит, виконується, до журналу додається запис; очевидно, що чим більше використовується аудит, тим більше записів буде додано до системного табличного простору, виключно для аудиту. У деяких випадках це може призвести до подвоєння кількості записів у базі даних: оригінальна запис і запис, яка створена для неї аудитом [5].

Основне правило налаштування аудиту - це простота та розуміння того, що аудит та детальний моніторинг потрібні тільки для тих операцій та об'єктів, інформація про які дійсно потрібна. Важливо, що за допомогою простих звітів можна виявити порушення серед дій, зафіксованих в журналі аудиту. Також варто зазначити, що при установці Oracle за замовчуванням аудит вимкнений, і Oracle не постачається з якими-небудь стандартними налаштуваннями аудиту за замовчуванням або звітами для аналізу створеного журналу аудиту.

Завдання аудиту бази даних Oracle не повинно обмежуватися використанням команд аудиту; також успішно можуть бути використані інші технології.

Аудит Oracle. Усі привілеї, які можуть бути надані користувачеві або ролі бази даних, можуть бути проконтрольовані. Сюди включено доступ на читання, запис та видалення об’єктів на табличному рівні. Для більш детального аудиту можна використовувати тригери [6].

Системні тригери. Ця можливість була представлена, починаючи з Oracle 8, і дозволяє виконання операцій тригера, коли має місце системна подія. Сюди включені

запуск та зупинення бази даних, спроби входу та виходу, створення, зміна та видалення об'єктів схеми. За допомогою автономних транзакцій можна записувати в журнал згадані системні події [6].

Update, delete та insert тригери. Це "друга лінія оборони", яка дозволяє зрозуміти дії користувачів на більш детальному рівні. Для того, щоб відстежувати зміни в базі на рівні стовпця та рядка, можна написати тригери, які дозволять повністю зберігати дані до або після виконаної дії. Використання цього контролю дуже ресурсоємно, оскільки створюється і зберігається багато додаткових записів. Крім того, що існує ще один недолік, пов'язаний з цим методом – доступ на читання не можна відстежити за допомогою звичайних тригерів бази даних [5-6].

Детальний (Fine-grained) аудит. Детальний аудит вирішує проблему відстеження доступу на читання. Ця можливість заснована на внутрішніх тригерах, які спрацьовують, при розбиранні якоїсь частини SQL-пропозиції. Це дуже ефективно, оскільки SQL-пропозиція розбирається один раз для аудиту та виконання. Ця можливість використовує предикати, які визначені та перевіряються щоразу, коли відбувається доступ до відповідних об'єктів. Fine-grained аудит управляється PL/SQL пакетом який називається DBMS_FGA. Створена PL/SQL процедура виконується щоразу, коли виконується, що відповідає їй, дію з предикатом. Цей метод дозволяє контролювати як DML-операції лише на рівні рядків і стовпців, а й пропозиції читання. Слід застерегти читачів у тому, що для використання цієї можливості потрібний певний досвід програмування [7].

Системні журнали. СУБД Oracle генерує багато журнальних файлів, і багато хто з них може містити корисну інформацію для проведення аудиту. Наприклад, alert log використовується для запису інформації про запуск і зупинення бази, а також про структурні зміни, що вносяться, таких як додавання файлу даних до бази.

У виробничій базі даних нікому з користувачів не слід змінювати структуру схеми. Адміністраторам баз даних слід вносити зміни до спеціально відведеного для цього часу. Будь-які інші зміни слід розглядати як підозрілі. Спостереження за структурними змінами може увімкнути індикатори некоректного використання бази даних [7].

Аудит у Oracle розділений на три частини:

- аудит таких виразів як CREATE TABLE або CREATE SESSION;
- аудит привілеїв ALTER USER;
- аудит на об'єкт на об'єктному рівні SELECT TABLE.

Записи аудиту можуть бути розміщені або в аудиторську таблицю бази даних, або в аудиторський журнал операційної системи. Запис аудиту до журналу операційної системи в деяких випадках більш захищений, але це можливість доступна не всім платформ і її специфіка залежить від платформи. У цій статті як місце зберігання для журналу аудиту ми будемо використовувати базу даних.

Аудит включається для запису до бази даних додаванням наступного рядка у файлі init.ora. Символьний зв'язок до нього зазвичай може бути знайдений у `$ORACLE_HOME/dbs`.

```
audit_trail = db
```

Після цього базу даних потрібно перезапустити. Проста перевірка покаже, що аудит справді включений.

```
select name, value from v$parameter where name like
'audit%';
```

Результат роботи даного коду подано на рисунку 2.6.

```
SQL> select name,value from v$parameter where name like 'audit%';

NAME
--------------------------------------------------------------------------------
VALUE
--------------------------------------------------------------------------------
audit_sys_operations
FALSE

audit_file_dest
/u01/app/oracle/admin/orcl/adump

audit_syslog_level


NAME
--------------------------------------------------------------------------------
VALUE
--------------------------------------------------------------------------------
audit_trail
DB
```

Рис. 2.6. Перевірка роботи аудиту

Але контрольовані дії не відстежуються до тих пір, поки ці дії не задані явно; це вірно, крім випадків привілейованого доступу до бази даних, запуску та зупинки бази даних, структурних змін, таких як додавання файлу даних. Ці дії відстежуються у файлі операційної системи в `$ORACLE_HOME/rdbms/audit`, доки `audit_file_dest` не перевизначено у файлі `init.ora`. У Windows ці події з'являються в Event Viewer.

Для того, щоб перевірити наявність того, що якісь привілеї або вирази вже використовуються для аудиту, необхідно зробити наступне [7]:

```
select * from dba_stmt_audit_opts union
select * from dba_priv_audit_opts;
```

Щоб знайти якісь об'єкти вже контролюються аудитом, необхідно запросити подання `dba_obj_audit_opts`.

Щоб користувач міг задати команду аудиту, необхідною умовою для нього є наявність привілею AUDIT SYSTEM". Знайти користувачів, які мають цей привілей, можна, виконавши наступне:

```
select * from dba_sys_privs
where privilege like ' %AUDIT%';
```

Цей скрипт виведе набір команд аудиту в спул файл, який потім запуститься для виконання команд аудиту.

```
    set head off set feed off set pages 0 spool aud.lis
    select 'audit '||name||';' from system_privilege_map
    where (name like 'CREATE%TABLE%' or name like
'CREATE%INDEX%'
    or name like 'CREATE%CLUSTER%' or name like
'CREATE%SEQUENCE%'
    or name like 'CREATE%PROCEDURE%' or name like
'CREATE%TRIGGER%'
    or name like 'CREATE%LIBRARY%') union
    select 'audit '||name||';' from system_privilege_map
    where (name like 'ALTER%TABLE%' or name like 'ALTER%INDEX%'
    or name like 'ALTER%CLUSTER%' or name like
'ALTER%SEQUENCE%' or name like 'ALTER%PROCEDURE%' or name like
'ALTER%TRIGGER%'
    or name like 'ALTER%LIBRARY%') union
    select 'audit '||name||';' from system_privilege_map
    where (name like 'DROP%TABLE%' or name like 'DROP%INDEX%'
    or name like 'DROP%CLUSTER%' or name like 'DROP%SEQUENCE%'
    or name like 'DROP%PROCEDURE%' or name like 'DROP%TRIGGER%'
    or name like 'DROP%LIBRARY%') union
    select 'audit '||name||';' from system_privilege_map
    where (name like 'EXECUTE%INDEX%' or name like
'EXECUTE%PROCEDURE%' or name like 'EXECUTE%LIBRARY%')
    /spool off @@aud.lis
```

Для створення команд аудиту можна було б використовувати інший спосіб через уявлення бази даних dba_sys_privs, що використовує дійсні дозволи користувачів.

Цей спосіб може здатися кращим рішенням і включає менше команд, але потенційно це б не спрацювало для випадків, коли нові дозволи надані користувачам. У цьому випадку довелося б виконувати нові команди аудиту після надання нових привілеїв.

Зараз установки можуть бути переглянуті за допомогою цього SQL:

```
select audit_option,success,failure from
dba_stmt_audit_opts
union
select privilege,success,failure from dba_priv_audit_opts
```

Щоразу, коли користувач намагається щось торкнутися в базі даних, на що включений аудит, ядро Oracle перевіряє дію та створює або оновлює (у разі одного запису для сесії) запис у таблиці `AUD$`, власником якої є користувач SYS. За умовчанням ця таблиця знаходиться в табличному просторі SYSTEM.

`AUD$` - особлива таблиця [словника даних], оскільки з неї користувач SYS має право видаляти з неї записи. Якщо журнал аудиту включений і пишеться в базу даних, то число записів у цій таблиці необхідно уважно контролювати, щоб переконатися, що вона не росте занадто швидко, і не заповнила весь системний табличний простір. Стратегія очищення потребує адаптації, щоб зберегти розмір таблиці і якщо необхідно архівувати записи журналу аудиту для майбутнього використання. Одна з тактик може полягати в тому, щоб копіювати записи в підсумкову таблицю, створену для виконання спецперевірки з метою виявлення зловживань. Ці підсумкові таблиці можуть розташовуватися в окремій базі даних збільшення захищеності. Після копіювання таблиця `sys.aud$` може бути зрізана [8].

Таблицю `SYS.AUD$` можна пересунути в табличний простір, відмінний від SYSTEM. Тільки користувачі, яким надано спеціальний доступ до таблиці `SYS.AUD$`, можуть читати, змінювати та видаляти дані з неї. За промовчанням ці права мають SYS, але ці дії може виконувати будь-який інший користувач, наділений необхідними правами. Існують дві спеціальні ролі, яким дозволено доступ до таблиці

SYS.AUD$ на select і delete, це ролі DELETE_CATALOG_ROLE та SELECT_CATALOG_ROLE. Ці ролі не слід надавати звичайним користувачам.

Записи аудиту можуть бути переглянуті різними способами:

- шляхом вибірки записів із SYS.AUD$ - Це вихідний журнал аудиту;
- шляхом вибірки записів з dba_audit_trail - Це подання DBA, що показує вихідний журнал аудиту;
- шляхом вибірки записів з dba_audit_session - Це уявлення показує лише події входу і виходу;

Невдалі спроби входу. Вони можуть означати спроби атакуючого отримати неавторизований доступ до бази даних. Нижченаведений SQL яскраво демонструє це [8]:

```
select count(*), username, terminal, to_char(timestamp,
'DD-MON-YYYY') from dba_audit_session where returncode<>0
group by username, terminal, to_char(timestamp, 'DD-MON-
YYYY');
```

Спроби доступу неіснуючих користувачів до бази даних

Один цікавий додаток до наведеного вище SQL дозволяє знайти спроби входу в систему під неіснуючим користувачем. У цьому випадку також буде створено записи аудиту. Наступний SQL ілюструє це:

```
Select username, terminal, to_char (timestamp, 'DD-MON-
YYYY HH24:MI:SS') from dba_audit_session where returncode<>0
and not exists (select 'x' from dba_users where
dba_users.username= dba_audit_session.username)
```

Усі спроби увійти в систему під неіснуючим користувачем слід перевіряти та розслідувати щодня.

Спроби доступу до бази даних у незвичайний час. Слід перевіряти спроби доступу до бази даних у позаробочий годинник. Їм може виявитись звичайна понаднормова робота, але також легко - неавторизований доступ. Його можна перевірити наступним виразом:

```
Select username, terminal, action_name,
returncode,to_char (timestamp, 'DD-MON-YYYY HH24:MI:SS'),
to_char(logoff_time, 'DD-MON-YYYY HH24:MI:SS')
from dba_audit_session where to_date (to_char
(timestamp,'HH24:MI:SS'),'HH24:MI:SS') < to_date ('08:00:00',
'HH24:MI:SS')
or to_date (to_char
(timestamp,'HH24:MI:SS'),'HH24:MI:SS') > to_date ('19:30:00',
'HH24:MI:SS');
```

Наведені вище SQL показує будь-які з'єднання до 8:00 ранку та після 7:30 вечора. Будь-які з'єднання, особливо ті, які виконані привілейованими користувачами, такими як SYS або SYSTEM, мають бути досліджені. Особливу увагу слід звернути на те, звідки було зроблено доступ. Наприклад, якщо привілейований доступ виконано з машини, яка не знаходиться у відділі адміністратора, адміністратор повинен з'ясувати, навіщо він проводився.

Перевірка користувачів, які використовують загальний обліковий запис у базі даних.

Наступний вираз SQL шукає користувачів, які потенційно можуть використовувати загальний обліковий запис у базі даних:

```
select count(distinct(terminal)),username from
dba_audit_session
having count(distinct(terminal))>1 group by username
```

Множинні спроби доступу під різними обліковими записами з одного терміналу.

Даний вираз SQL досить простий і до нього може бути додано угруповання по дню, а також виведені користувачі для кожного терміналу [9].

```
select count(distinct(username)),terminal from
dba_audit_session
having count(distinct(username))>1 group by terminal
```

Цей звіт показує будь-кого, хто намагається отримати доступ перебором облікових записів і паролів, але сюди можуть потрапити законослухняні користувачі, які використовують різні облікові записи для різних аспектів своєї роботи. У будь-якому разі адміністратору слід з'ясувати це надалі.

Налаштування аудиту є одним з перших кроків для забезпечення безпеки бази даних. Використання аудиту має бути частиною загального плану безпеки організації, до якого входить і Oracle. Слід регулярно контролювати базу даних на неправильність конфігурації або наявність новоутворених уразливостей, які можуть стати брехнею в інформаційній безпеці системи.

Через свою складну природу і велику кількість різних параметрів, сервер Oracle може бути по-різному налаштований, однак, щоб найкращим чином забезпечити безпеку необхідно завжди дотримуватися принципу найменших привілеїв. Як тільки база даних стане частиною загального плану безпеки і буде коректно налаштована та регулярно перевіряється, тоді аудит слід розглядати як важливу частину цієї спільної стратегії [9].

В основному, не варто представляти будь-які привілеї звичайним користувачам у виробничій базі даних, варто видалити більшість привілеїв PUBLIC, видалити, заблокувати або змінити паролі всіх облікових записів за умовчанням. Варто переконатися в тому, що користувачі дотримуються політики безпеки під час роботи з паролями та увімкнено функцію управління паролями.

Важливо, щоб налаштування аудиту планувалося з точки зору продуктивності та зручності використання. Журнал аудиту також повинен був керувати.

Не менш важливим є те, що дані журналу аудиту можна описувати в категоріях захисту інформації.

РОЗДІЛ 3
РОЗРОБКА БАЗИ ДАНИХ

3.1. Постановка задачі

У цій кваліфікаційній роботі необхідно забезпечити безпеку бази даних автоматизації проведення змагань з фігурного катання. База повинна передбачати відображення категорій фігуристів, тренерів, суддів, що є в базі даних, кожне місце (призове та не призове) фігуриста в рамках цього змагання, а також враховувати інформацію про те, яким суддею, якому фігуристу, яка оцінка була поставлена за ту чи іншу програму. Інформація про оцінки в даній базі даних призначена для перегляду фігуристом, тренером або суддею, але вона також виключає можливість модифікації, оновлення або видалення даних ними. Таким чином, вводиться необхідність розмежування прав доступу.

Для досягнення поставленої мети база даних «Автоматизація проведення змагань з фігурного катання» має надавати такі можливості:

- зберігання інформації про фігуристів;
- зберігання інформації про тренерів;
- зберігання інформації про хореографів;
- зберігання інформації про суддів;
- зберігання інформації про технічну частину проведення змагань;
- облік та автоматизація оцінок фігуристів.

При роботі з базою технічний фахівець повинен мати змогу вирішувати такі завдання:

1) приймати заявки на участь та реєструвати їх у системі;
2) розміщувати інформацію про програми фігуристів;
3) фіксувати оцінки фігуристів;

4) проводити видалення фігуристів із бази. Дане явище може мати місце під час розпаду пар, закінчення кар'єри в аматорському вигляді фігурного катання, відмові від участі у змаганні тощо.

Фігурист, тренер, хореограф, суддя мають змогу вирішувати такі завдання:

1) переглядати список учасників змагання;

2) переглядати поточні та підсумкові оцінки, без можливості редагування;

3) для обраного фігуриста (судді, тренера, хореографа) отримати цікаві відомості.

3.2. Аналіз предметної області

Потрібно розробити базу даних для автоматизації проведення змагань із фігурного катання. База містить відомості про фігуристів, які характеризуються такими параметрами (реалізовано у таблиці Фігурист (Figure_Skater):

- унікальний код фігуриста;
- ім’я;
- прізвище;
- країна;
- стать;
- дата народження;
- місто проживання;
- зріст;
- рік початку кар’єри;
- тренер;
- хореограф;
- категорія фігурного катання.

Категорія суддів, що оцінюють програми фігуристів, характеризується такими параметрами (реалізовано у таблиці «Суддя (Judge)»):

- унікальний код судді;

- ім’я;
- прізвище;
- стать;
- дата народження.

Кожен фігурист (пара) має свого тренера та хореографа, які у свою чергу можуть тренувати кількох фігуристів. Кожен тренер і хореограф мають такі характеристики (реалізовано у таблицях «Тренер (Coach)» та «Хореограф (Choreographer)» відповідно):

- унікальний код тренера (хореографа);
- прізвище;
- ім’я.

У системі є інформація про змагання, яка представлена такими характеристиками (реалізовано в таблиці «Про Змагання (About_Competition)»):

- унікальний код змагання;
- назва змагання;
- країна проведення;
- місто проведення;
- адреса льодової арени;
- дата початку змагання;
- дата кінця змагання.

У межах змагання фігуристи та судді ідентифікуються за заявками. Інформація про заявки представлена наступними характеристиками (реалізовано у таблиці "Заявка Фігуристи (Request_Figure_Skating)"):

- унікальний код заявки;
- унікальний код 1-го фігуриста;
- унікальний код 2-го фігуриста (може бути представлений порожнім полем у разі одиночної категорії фігурного катання);
- унікальний код змагання.

У ході проведення змагання фігуристи представляють 2 види програм: коротку та довільну, які оцінюють судді. Система суддівства наступна: кожен суддя ставить оцінку кожному фігуристу (парі) за кожен елемент, виконаний у програмі (від -3 до 3).

Далі виводиться середній бал, який згодом підсумовується з базовою вартістю елемента. Перелік можливих елементів є у системі (реалізовано в таблиці «Заявка Суддя (Request_Judge)»).

У системі також є інформація про програми фігуристів, яка має такі параметри (реалізовано в таблиці «Про програму (About_Program)»):

- унікальний код виду програми;
- унікальний код заявки;
- унікальний код змагання;
- назва програми;
- музика.

Оцінки за технічні елементи за результатами програм зберігаються у системі та мають такі характеристики (реалізовано у таблиці «Оцінка за техніку (Request_Figure_Skating)»):

- унікальний код судді;
- унікальний код заявки фігуристів;
- унікальний код виду програми;
- унікальний код елемента;
- оцінка за елемент.

Після закінчення кожного етапу змагання підсумкові оцінки заносяться до системи, і визначається поточне місце фігуриста (пари). Після закінчення змагання підраховуються підсумкові оцінки, шляхом підсумовування результатів двох програм, підбиваються результати, присуджуються місця та медалі (реалізовано в таблиці «Результат (Result)»).

Під час початкової стадії розробки БД та концептуального проєктування ми визначаємо початковий набір сутностей. Ці сутності є найбільш важливою інформацією

про об'єкти системи з погляду представлення кінцевих користувачів і проєктувальника. Деякі сутності є об'єктами реального світу, наприклад фігуристи, тренери, судді. Інші являють собою інформацію про сутності, наприклад, елементи, категорії фігурного катання. У цій роботі будуть використовуватися сутності, подані в таблиці 3.

Таблиця 3.1

Опис призначення таблиць

Ім'я сутності	Опис сутності	Тип сутностей
Segment	Дані про види програм	Сильна
Gender	Дані про поле	Сильна
Medal	Дані про медаль	Сильна
Category	Дані про категорію фігурного катання	Сильна
Element	Дані про елементи, які фігуристи виконують у своїх програмах	Сильна
Figure_Skater	Дані про фігуриста	Слабка
Judge	Дані про суддю	Слабка
Coach	Дані про тренера	Сильна
Choreographer	Дані про хореографа	Сильна
About_Competition	Дані про змагання	Сильна
Request_Figure_Skater	Дані про заявку фігуриста (-ів) на участь у змаганні	Слабка
Request_Judge	Дані про заявку судді на суддівство в змаганні	Слабка
About_Programm	Дані про програму, що надається фігуристом(-ами)	Слабка
Technical_Element_Score	Дані про оцінки фігуриста(-ів) за технічні елементи	Слабка
Total_Segment_Score	Дані про оцінки фігуриста(-ів) за види програми	Слабка
Result	Дані про підсумкову оцінку фігуриста(-ів), місцях та медалях.	Слабка

3.3. Концептуальне проєктування

Враховуючи особливості даної системи, розглянутої в аналізі предметної області, виділимо таблиці та зв'язки між ними, які представлені на попередній ER-діаграмі. Загальна ER-діаграма представлена рисунку 3.1.

На рисунку 13 представлено остаточний варіант концептуальної моделі бази даних, з урахуванням усіх бізнес правил, призначеної для проведення змагань з фігурного катання.

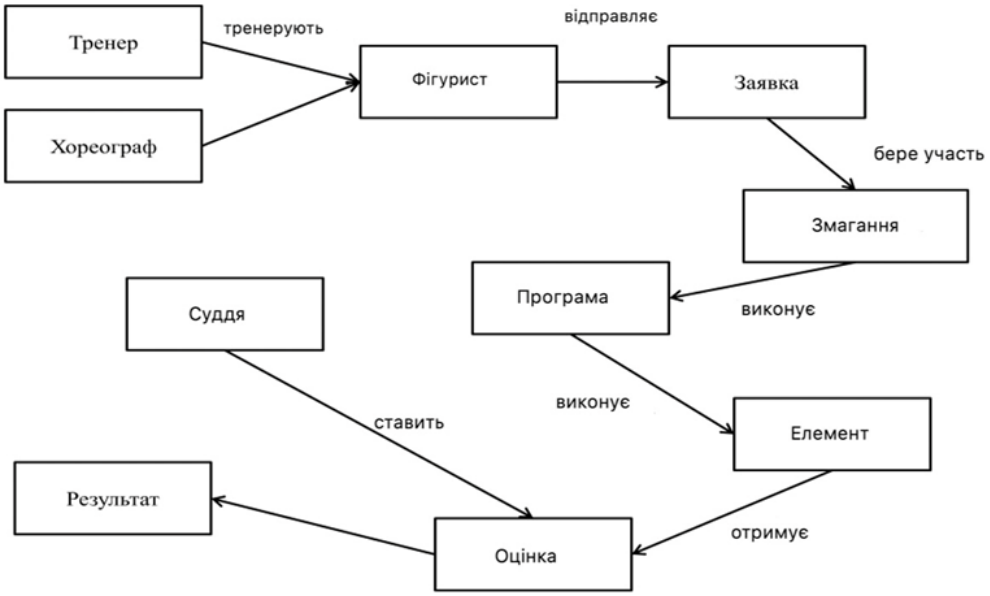

Рис. 3.1. Загальна ER-діаграма

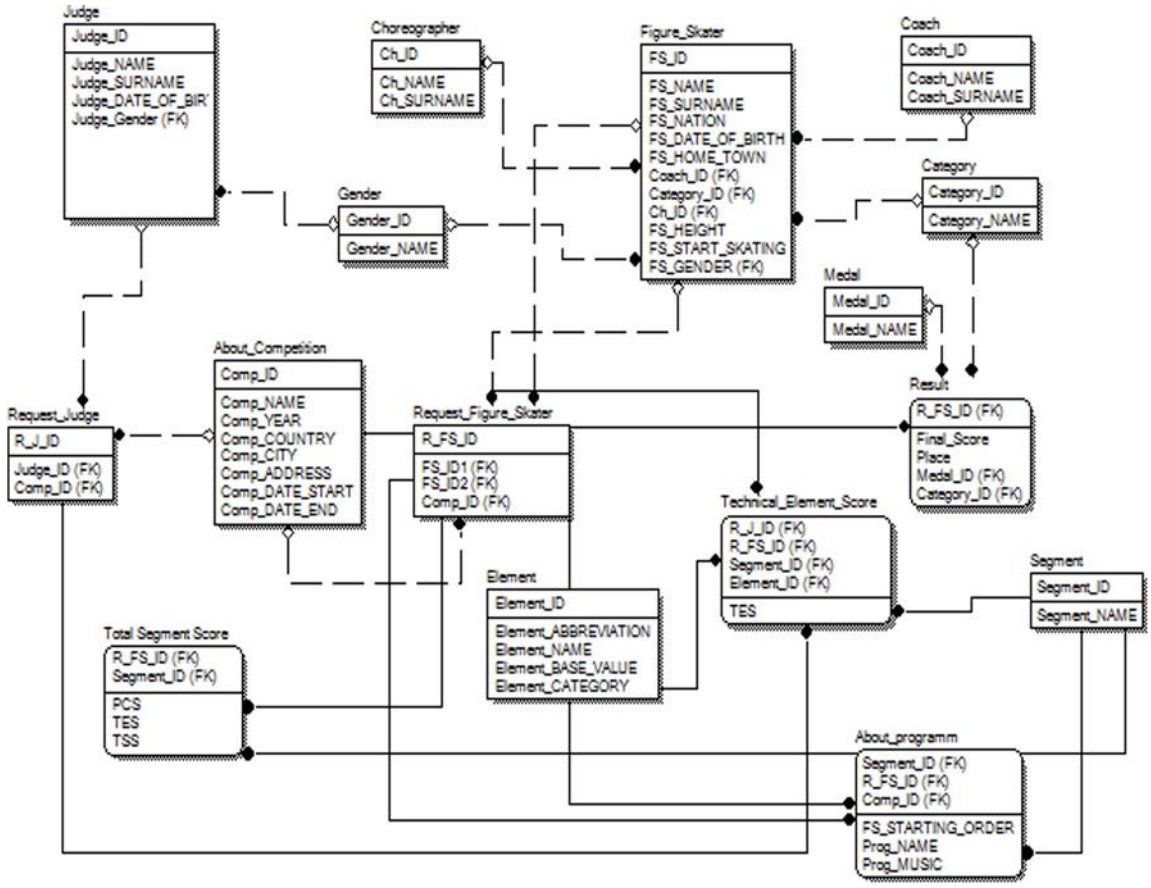

Рис. 3.2. Остаточний варіант концептуальної моделі

3.4. Логічне проєктування

Завдання логічного проєктування полягає у створенні реляційних табличних структур мовою DDL.

Однією з найважливіших елементів у БД є розробка прав доступу до неї, так як потрібна захист від несанкціонованого доступу та захист від доступу. Для захисту від збоїв розробляється стратегія резервного копіювання. Для захисту від несанкціонованого доступу кожному користувачеві доступ до даних надається лише відповідно до його прав доступу.

3.4.1. Створення користувачів, логінів та завдання паролів

Однією з найважливіших складових проєкту бази даних є розробка засобів захисту БД. Захист даних має два аспекти: захист від збоїв і захист від несанкціонованого доступу. Для захисту від збоїв розробляється стратегія резервного копіювання. Для захисту від несанкціонованого доступу кожному користувачеві доступ до даних надається тільки відповідно до його прав доступу. Для нашої бази даних на основі аналізу предметної області створюються 3 облікові записи: адміністратора, суддів та користувача (фігуриста, тренера, хореографа). Адміністратору надається повний контроль над БД, технічні оператори можуть додавати, видаляти та оновлювати дані, а в права користувача вносяться обмеження на зміну і оновлення таблиць.

```
    CREATE          TABLESPACE          fss          DATAFILE
'/u01/app/oracle/oradata/asel/fss.DAT'SIZE     150M     REUSE
AUTOEXTEND ON NEXT 2M MAXSIZE 500M;
    create user fss identified by fss DEFAULT TABLESPACE fss
TEMPORARY TABLESPACE TEMP; alter user fss QUOTA unlimited ON
fss; grant create session to fss; grant create any table to
fss; grant create any procedure to fss; grant create any trigger
to fss; grant create any sequence to fss; grant create any view
to fss; grant update any table
```

Рис. 3.2. Створення користувачів в базі даних (фрагмент коду)

3.4.2. Схеми відносин, складені мовою визначення даних (ddl, data definition language)

Створення таблиць

```
    CREATE TABLE Gender(
    Gender_ID number PRIMARY KEY, Gender_NAME VARCHAR2(10) NOT
NULL);
    CREATE TABLE Categor ( Category_ID number PRIMARY KEY,
    Category_NAME VARCHAR2(40) NOT NULL);
    CREATE TABLE Segment( Segment_ID number Primary key,
    Segment_NAME varchar2(30) not null);
    CREATE TABLE Element(
    Element_ID    number    not    null    PRIMARY    KEY,
Element_ABBREVIATION  varchar2(10)  not  null  ,  Element_NAME
VARCHAR2(80)  not  null,  Element_BASE_VALUE  float  not  null,
Element_Category varchar2(15) not null);
    CREATE TABLE Coach(
    Coach_ID number Primary Key, Coach_NAME varchar2(20) not
null,
    Coach_SURNAME varchar2(30));
    CREATE  TABLE  Choreographer(  Ch_ID  number  Primary  Key,
Ch_NAME varchar2(20) not null, Ch_SURNAME varchar2(30));
```

Рис. 3.3. Створення таблиць (фрагмент коду)

3.5. Фізичне проєктування

Oracle була обрана як СУБД для розробки бази даних. Вона є всеосяжною, інтегрованою системою, яка забезпечує користувачів організації безпечною, надійною та продуктивною платформою для обробки індустріальної інформації та додатків, що

стосуються інтелектуальних ресурсів підприємства. Oracle надає потужні, знайомі інструменти для IT-фахівців та працівників інформаційної сфери, зменшуючи складність створення, розгортання, управління та використання даних підприємства та аналітичних додатків на платформах від мобільних пристроїв до інформаційних систем підприємства. Завдяки повному набору функцій, взаємодії з існуючими системами та автоматизації типових задач, Oracle 11 g надає повне рішення у галузі зберігання даних для підприємств будь-якого масштабу.

Діаграма фізичної моделі даної бази даних представлена рисунку 3.4.

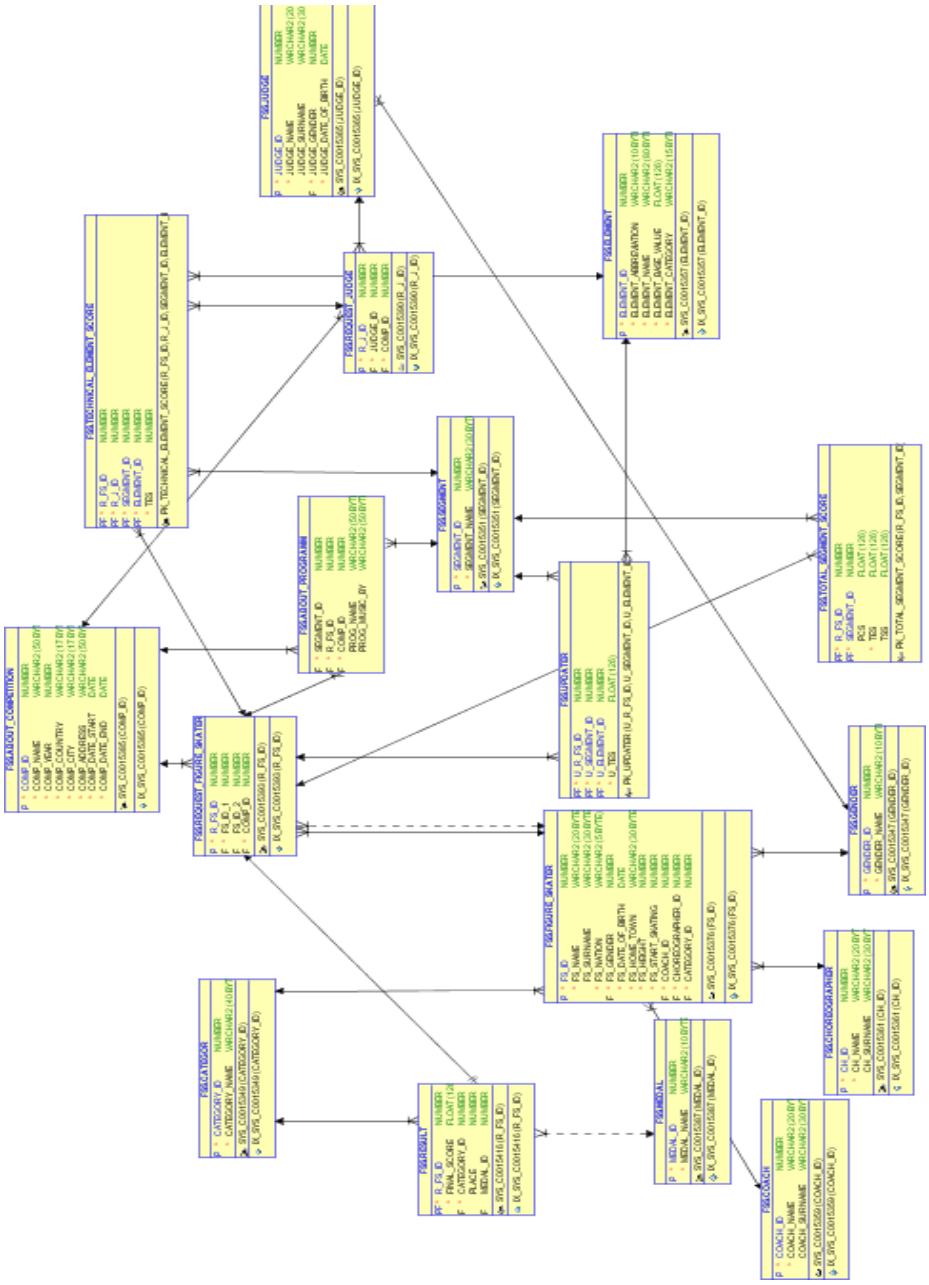

Рис. 3.4. Діаграма фізичної моделі цієї бази даних

3.6. Створення клієнтської програми

Для створення клієнтської програми було вибрано середовище розробки JDeveloper. JDeveloper - інтегроване середовище розробки програмного забезпечення, розроблене корпорацією Oracle. Надає можливість для розробки мовами програмування Java, JavaScript, BPEL, PHP, SQL, PL/SQL та мовами розмітки HTML, XML.

JDeveloper покриває весь життєвий цикл розробки програмного забезпечення від проєктування, кодування, налагодження, оптимізації та профілювання до його розгортання. На рисунку 3.5 представлений повний життєвий цикл розробки додатків.

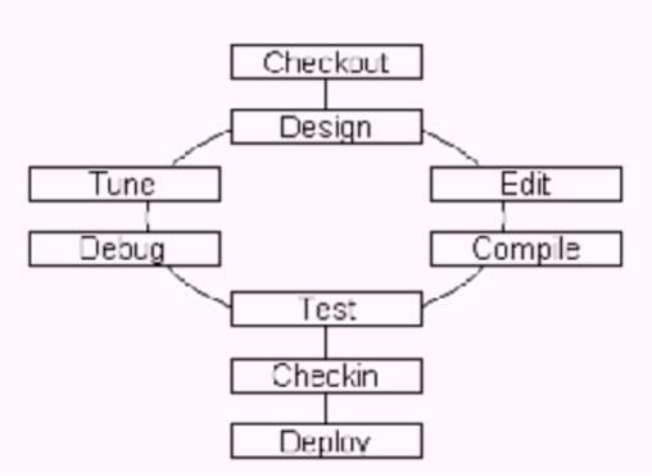

Рис. 3.5. Повний життєвий цикл для розробки програм у середовищі JDeveloper

У типовому сценарії розробки розробник запускає JDeveloper, перевіряє додаток за допомогою системи управління вихідним кодом та починає цикл розробки. Розробники UML-моделей допомагають розробнику при конструюванні додатка та, можливо, при генерації вихідного коду. JDeveloper надає майстри (wizards) та редактори - як візуальні, так і зорієнтовані на код (тобто, працюють з вихідним текстом програми) - щоб додати функціональні можливості та різні інструментальні засоби для компіляції, перевірки, налагодження та налаштування додатка. Коли розробник буде задоволений, він може повернути додаток до системи управління вихідним кодом для ще однієї перевірки, а після достатнього тестування розгорнути його (додаток) в призначеному місці.

Виробник зазначає як основне завдання середовища - максимальне використання можливостей візуального та декларативного підходу до розробки програмного забезпечення на додаток до зручного середовища кодування. Oracle JDeveloper інтегрована з Oracle ADF - Java EE-каркасом для створення комерційних програм на Java.

До версії 11g JDeveloper поставлявся в трьох редакціях (від молодшої до старшої): Java Edition, J2EE Edition та Studio Edition. Кожна старша редакція включає можливості молодшої, всі редакції постачалися безкоштовно. JDeveloper 11g має лише дві редакції: Java Edition та Studio Edition (можливості J2EE Edition внесені до Studio Edition).

JDeveloper написана повністю на Java, тому працює на всіх операційних системах, що мають JDK.

Інтерфейс цієї програми дуже зручний для використання. Із додатком можуть працювати два види користувачів: користувач та технічний оператор. Технічному оператору надано всі права, а користувачі можуть лише переглядати інформацію.

Після запуску програми відкривається перша форма - форма входу, представлена рисунку 3.6. На цьому етапі проводиться авторизація користувача (рисунок 3.7). При введенні неправильного пароля системи видасть повідомлення про помилку (наведено на рисунку 3.8).

Рис. 3.6. Форма входу (Welcome page)

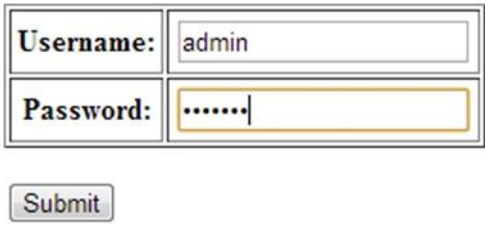

Рис. 3.7. Авторизація користувача. Введення пароля

Invalid username or password

Рис. 3.8. Повідомлення про помилку під час введення неправильного пароля

При здійсненні входу під логіном "Технічний оператор (Techoper)" пароль є комбінацією - "techoper123". Технічний оператор має права на всю базу даних. Він може змінювати, додавати та видаляти будь-які дані з будь-якої таблиці, контролювати систему обліку оцінок. При успішній авторизації під "Технічний оператор (Techoper)" відкриється форма, представлена на рисунку 3.9.

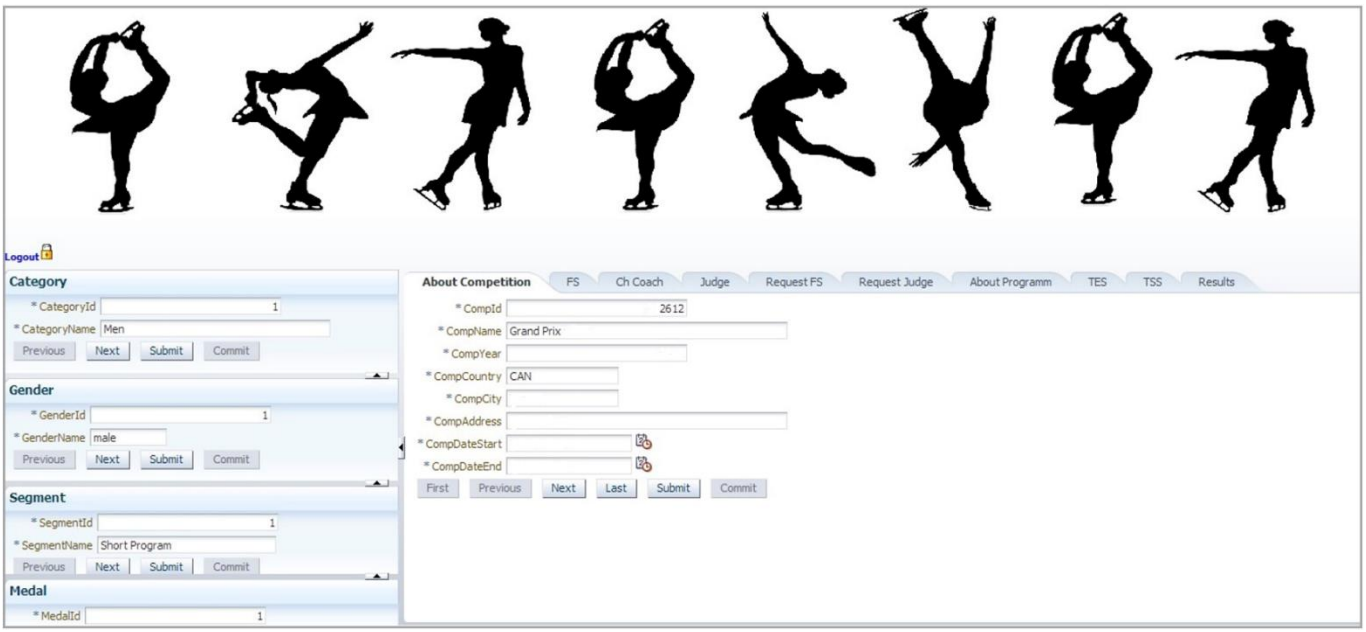

Рис. 3.9. Форма адміністратора

У формі, представленій на рисунку 19, технічний оператор має право додавати, змінювати або видаляти дані про Поле (Gender), Медалі (Medal), Вид програми

(Segment), Категорії фігуриста (Category) та ін. Кнопки для виконання вищеописаних операцій представлені на рисунку 3.10.

Рис. 3.10. Кнопки для додавання, видалення та збереження даних

При здійсненні входу під логіном "Користувач (User)" пароль є комбінацією - "user1234". Користувач може переглядати таблиці, дозволені системою. При успішній авторизації під логіном "Користувач(User)" відкриється форма користувача, представлена на рисунку 3.11.

Рис. 3.11. Форма користувача

ВИСНОВКИ

У ході виконання даної кваліфікаційної роботи було розроблено базу даних «Автоматизація проведення змагань з фігурного катання» та забезпечено її безпеку.

При проєктуванні роботи були враховані всі основні функції даної бази даних. Ця база даних проєктувалася на Oracle 11g, так це середовище проєктування, що найбільше задовольняє за функціональними можливостями. У базі даних враховані права користувачів, і доступом до інформації обмежений. Так, наприклад, змінювати, додавати та видаляти інформацію з таблиць може лише технічний оператор. В той час, як користувач може тільки переглядати інформацію.

База даних містить безліч тригерів і процедур, що зберігаються, які описані вище, це дозволяє прискорити процес обробки інформації, а так само спростити використання програми на великій кількості комп'ютерів. Система передбачає супровід програми на тривалий період, оскільки створена з урахуванням бажаних змін замовника, що вносяться.

Для захисту був використаний продукт Oracle Advanced Security, а саме, Прозоре шифрування (TDE) та протокол аутентифікації Radius.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. S. Pendse et al., "Oracle Database In-Memory on Active Data Guard: Real-time Analytics on a Standby Database," 2020 IEEE 36th International Conference on Data Engineering (ICDE), Dallas, TX, USA, 2020, pp. 1570-1578.

2. P. Jing-wei, Z. Min, C. Ping and X. Wei-guang, "A Lightweight Vulnerability Scanning and Security Enhanced System For Oracle Database," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 2019, pp. 1699-1702.

3. R. Čerešňák, K. Matiaško and A. Dudáš, "Improvement of Data Searching in MongoDB with the Use of Oracle Database," 2021 18th International Multi-Conference on Systems, Signals & Devices (SSD), Monastir, Tunisia, 2021, pp. 1388-1393.

4. S. Wang, Y. Yang and S. Liu, "Research on Audit Model of Dameng Database based on Security Configuration Baseline," 2020 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS), Shenyang, China, 2020, pp. 833-836.

5. K. Natarajan and V. Shaik, "Transparent Data Encryption: Comparative Analysis and Performance Evaluation of Oracle Databases," 2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), Bangalore, India, 2020, pp. 137-142.

6. M. Fotache, A. Munteanu, C. Strîmbei and I. Hrubaru, "Framework for the Assessment of Data Masking Performance Penalties in SQL Database Servers. Case Study: Oracle," in IEEE Access, vol. 11, pp. 18520-18541, 2023.

7. S. Alornyo, E. Aidoo, K. Kissi Mireku, B. Kwofie, X. Hu and M. Asante, "ID-Based Outsourced Plaintext Checkable Encryption in Healthcare Database," 2019 International Conference on Cyber Security and Internet of Things (ICSIoT), Accra, Ghana, 2019, pp. 48-53.

8. J. Chen, H. Du, Z. Wang, N. Xue, J. Peng and W. Li, "Method for Mining Security Vulnerabilities of Data Storage of Electric Power Internet of Things Based On

Spark Framework and RASP Technology," 2022 International Conference on Knowledge Engineering and Communication Systems (ICKES), Chickballapur, India, 2022, pp. 1-5.

9. M. Salahat et al., "Analysis of Query Processing on Different Databases," 2023 International Conference on Business Analytics for Technology and Security (ICBATS), Dubai, United Arab Emirates, 2023, pp. 1-7.

10. J. Guo and J. Sun, "Secure and Efficient Nearest Neighbor Query for an Outsourced Database," in IEEE Access, vol. 8, pp. 83754-83764, 2020.

11. M. Kvet and J. Papan, "The Complexity of the Data Retrieval Process Using the Proposed Index Extension," in IEEE Access, vol. 10, pp. 46187-46213, 2022.

12. L. Liu, "Facial Authentication System Design of Online Interactive Platform for Innovation and Entrepreneurship Courses for Mobile Platform Terminals," 2022 3rd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2022, pp. 1649-1652.

13. R. G. L. D'Oliveira, S. E. Rouayheb, D. Heinlein and D. Karpuk, "Notes on Communication and Computation in Secure Distributed Matrix Multiplication," 2020 IEEE Conference on Communications and Network Security (CNS), Avignon, France, 2020, pp. 1-6.

14. M. Müller, S. Rodriguez Garzon and A. Küpper, "COST: A Consensus-Based Oracle Protocol for the Secure Trade of Digital Goods," 2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS), Oxford, UK, 2020, pp. 72-81.

15. H. Al Breiki, L. Al Qassem, K. Salah, M. Habib Ur Rehman and D. Sevtinovic, "Decentralized Access Control for IoT Data Using Blockchain and Trusted Oracles," 2019 IEEE International Conference on Industrial Internet (ICII), Orlando, FL, USA, 2019, pp. 248-257.

16. C. Shou, İ. B. Kadron, Q. Su and T. Bultan, "CorbFuzz: Checking Browser Security Policies with Fuzzing," 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), Melbourne, Australia, 2021, pp. 215-226.

17. Z. -Y. Zhao and P. Zeng, "Efficient All-or-Nothing Public Key Encryption With Authenticated Equality Test," in IEEE Access, vol. 9, pp. 94099-94108, 2021.

18. K. Park et al., "LAKS-NVT: Provably Secure and Lightweight Authentication and Key Agreement Scheme Without Verification Table in Medical Internet of Things," in IEEE Access, vol. 8, pp. 119387-119404, 2020.

19. B. Huang et al., "BoR: Toward High-Performance Permissioned Blockchain in RDMA-Enabled Network," in IEEE Transactions on Services Computing, vol. 13, no. 2, pp. 301-313, 1 March-April 2020.

20. D. E. Simos, J. Zivanovic and M. Leithner, "Automated Combinatorial Testing for Detecting SQL Vulnerabilities in Web Applications," 2019 IEEE/ACM 14th International Workshop on Automation of Software Test (AST), Montreal, QC, Canada, 2019, pp. 55-61.