

**МІНІСТЕРСТВО ВНУТРІШНІХ СПРАВ УКРАЇНИ**

**Харківський національний університет внутрішніх справ**

**Факультет №4**

**Кафедра інформаційних технологій**

**ТЕКСТ ЛЕКЦІЇ**

**з дисципліни «ОРГАНІЗАЦІЯ БАЗ ДАНИХ ТА ЗНАНЬ»**

**за темою: Клієнт-серверні технології СУБД Oracle**

**Галузь знань: 1701 «Інформаційна безпека»**

**Напрямок підготовки: 6.170102 «Системи технічного захисту інформації»**

**Освітньо-кваліфікаційний рівень: бакалавр**

**м. Харків  
2017 рік**

**Текст лекції** з дисципліни «Організація баз даних та знань» за темою: «Клієнт-серверні технології СУБД Oracle » для курсантів за напрямом підготовки «Системи технічного захисту інформації» (6.0170102).

**СХВАЛЕНО**

Науково-методичною радою  
Харківського національного  
університету внутрішніх справ

\_\_\_\_\_  
(дата, місяць, рік )      Протокол № \_\_\_\_\_

**ЗАТВЕРДЖЕНО**

Вченою радою факультету №4

\_\_\_\_\_  
(дата, місяць, рік )      Протокол № \_\_\_\_\_

\_\_\_\_\_  
(підпис)      **Марков В.В.**

**ПОГОДЖЕНО**

Секцією науково-методичної ради  
ХНУВС з технічних дисциплін

\_\_\_\_\_  
(дата, місяць, рік )      Протокол № \_\_\_\_\_

\_\_\_\_\_  
(підпис)      **Струков В.М.**

**ЗАТВЕРДЖЕНО**

На засіданні кафедри інформаційних  
технологій

\_\_\_\_\_  
(дата, місяць, рік )      Протокол № \_\_\_\_\_

\_\_\_\_\_  
(підпис)      **Струков В.М.**

**Рецензенти:**

Єрохін А.Л., декан факультету комп'ютерних наук ХНУРЕ, д.т.н., професор

Гнусов Ю.В., завідувач кафедри кібербезпеки ХНУВС, к.т.н., доцент

Розробники: Колісник Т.П.– Харків: Харківський національний університет  
внутрішніх справ, 2017.

## План лекції

1. Функції СУБД
2. Архітектура багатокористувачевих СУБД
3. Системний каталог.

## Література:

### Основна:

1. Дейт К.Дж. **Введение в системы баз данных**. 6-е издание. - Киев – Москва: Діалектика, 1998. - 784 с.; Стор. 45-54, 564-590.
2. Коннолли Т., Бегг К., Страчан А. **Базы данных: проектирование, реализация и сопровождение**. Теория и практика. 2-е издание. Москва – Санкт – Петербург - Киев: Вильямс, 2000. - 1111 с.; Стор. 88-102.
3. Д. Кренке. **Теория и практика построения баз данных**. 8-е изд. – СПб.: Питер, 2003. – 800 с.: ил. – (Серия «Классика Computer Science»); Стор. 379-413, 52-58.

### Додаткова:

1. Кириллов В.В. Основы проектирования реляционных баз данных. Учебное пособие. Санкт-Петербургский Государственный институт точной механики и оптики (технический университет). Кафедра вычислительной техники. - <http://www.cs.ifmo.ru>
2. Кузнецов С.Д. Основы современных баз данных. Информационно-аналитические материалы. - <http://www.citmgu.ru/>

## Текст лекції

### 1. Функції СУБД

Розглянемо типи функцій і служб (**сервісів**), які повинна забезпечувати типова СУБД. У свій час **Кодд** запропонував перелік з восьми сервісів, що повинні бути реалізовані в будь-який повномасштабній СУБД. Нижче наведено короткий опис кожного з них.

#### 1.1. Збереження, витяг і відновлення даних

СУБД повинна надавати користувачам можливість зберігати, витягати й оновлювати дані в базі даних.

Це **фундаментальна функція СУБД**. Спосіб реалізації цієї функції в СУБД повинний дозволити ховати від кінцевого користувача внутрішні деталі фізичної реалізації системи (наприклад, файлову організацію чи структури збереження, які використовуються).

#### 1.2. Каталог, який доступний кінцевим користувачам

СУБД повинна мати доступний кінцевим користувачам каталог, у якому зберігається опис елементів даних.

Ключовою особливістю такої архітектури є наявність інтегрованого системного каталогу з даними про схеми, користувачів, програми та ін. Передбачається, що каталог доступний як користувачам, так і функціям СУБД. **Системний каталог**, чи **словник даних**, є сховищем інформації, що описує дані в базі даних (по суті, це "**дані про дані**", чи **мета-дані**). У залежності від типу СУБД кількість інформації і спосіб її застосування можуть варіюватися. Як правило в системному каталозі зберігаються наступні зведення:

- імена, типи і розміри елементів даних;
- імена зв'язків;
- обмеження підтримки цілісності, що накладаються на дані;
- імена санкціонованих користувачів, яким надане право доступу до даних;
- зовнішня, концептуальна і внутрішня схеми і відображення між ними;

— статистичні дані, наприклад частота трансакцій і лічильники звертань до об'єктів бази даних.

Системний каталог дозволяє досягти наступних **переваг**:

— інформація про дані може бути централізовано зібрана і збережена, що дозволить контролювати доступ до цих даних, як і до будь-якого іншого ресурсу;

— можна визначити зміст даних, що допоможе іншим користувачам зрозуміти їх призначення;

— спрощується повідомлення, тому що зберігаються точні визначення змісту даних. У системному каталозі також можуть бути зазначені один чи декілька користувачів, що є власниками даних або мають право доступу до них;

— завдяки централізованому збереженню **надмірність і суперечливість** опису окремих елементів даних можуть бути легко виявлені;

— внесені в базу даних зміни можуть бути заархівовані;

— наслідки будь-яких змін можуть бути визначені ще до їх внесення, оскільки в системному каталозі зафіксовані всі існуючі елементи даних, встановлені між ними зв'язки, а також усі їх користувачі;

— міри забезпечення безпеки можуть бути додатково посилені;

— з'являються нові можливості організації підтримки цілісності даних;

— може виконуватися аудит інформації, що зберігається.

### 1.3. Підтримка трансакцій

СУБД повинна мати механізм, що гарантує виконання або всіх операцій відновлення даної трансакції, або жодної з них.

Але спочатку пояснимо, що таке **трансакція**.

**ТРАНСАКЦІЯ** ПРЕДСТАВЛЯЄ СОБОЮ НАБІР ДІЙ, ЯКІ ВИКОНУЮТЬСЯ ОКРЕМИМ КОРИСТУВАЧЕМ ЧИ ПРИКЛАДНОЮ ПРОГРАМОЮ З МЕТОЮ ДОСТУПУ ЧИ ЗМІНИ ВМІСТУ БАЗИ ДАНИХ.

Прикладами простих трансакцій у нашому тестовому проєкті може служити додавання в базу даних зведень про нового співробітника, відновлення зведень про зарплату деякого співробітника чи видалення зведень про деякий об'єкт нерухомості. Прикладом більш складної трансакції може бути видалення зведень про співробітника з бази даних і передача відповідальності за всі об'єкти нерухомості, які їм керуються, іншому співробітнику. У цьому випадку в базу даних буде потрібно внести відразу декілька змін. Якщо під час виконання трансакції відбудеться збій, наприклад через вихід з ладу комп'ютера, **база даних попадає в суперечливий стан, оскільки деякі зміни вже будуть внесені, а інші - ще ні**. Тому всі часткові зміни повинні бути скасовані для повернення бази даних у колишній, несуперечливий стан.

### 1.4. Сервіс керування паралельністю

СУБД повинна мати механізм, що гарантує коректне відновлення бази даних при паралельному (одночасному) виконанні операцій відновлення багатьма користувачами.

Одна з основних цілей створення і використання СУБД полягає у тому, щоб декілька користувачів мало можливість здійснювати одночасний доступ до даних, які спільно обробляються. Одночасний доступ порівняно просто організувати, якщо всі користувачі виконують тільки читання даних, оскільки в цьому випадку вони не можуть перешкодити один одному. Однак, коли два чи більше користувачів одночасно одержують доступ до бази даних, конфлікт із небажаними наслідками легко може виникнути, наприклад, якщо хоча б один з них спробує оновити дані. Розглянемо трансакції T1 і T2, які виконуються паралельно, послідовність виконання яких представлена в **табл.1**.

У трансакції T1 50 гривень знімається з рахунку, баланс якого зберігається в стовпці **Баланс**, а у трансакції T2 100 гривень зараховується на цей же рахунок. Якби трансакції виконувалися послідовно, тобто одна за іншою, без чергування окремих операцій, то в остаточному підсумку баланс дорівнював би 150 гривням, незалежно від

порядку виконання транзакцій. Інакше може виникнути наступна ситуація. Припустимо, що транзакції T1 і T2 стартували практично одночасно і прочитали вихідне значення балансу, рівним 100 гривень. Потім транзакція T2 збільшує це значення на 100 гривень (до 200 гривень) і зберігає отримане значення в базі даних. Негайно після цього транзакція T1 зменшує свою копію ліченого вихідного значення на 50 гривень (до 50 гривень) і зберігає отримане значення в базі даних замість тільки що записаного результату попереднього відновлення, унаслідок чого виникає ефект "утрати" 100 гривень.

Таблиця 1

Час	Транзакція T <sub>1</sub>	Транзакція T <sub>2</sub>	Баланс
t <sub>1</sub>		Читання Баланс	100
t <sub>2</sub>	Читання Баланс	Баланс = Баланс + 100	100
t <sub>3</sub>	Баланс = Баланс – 50	Запис Баланс	200
t <sub>4</sub>	Запис Баланс		50
t <sub>5</sub>			50

СУБД повинна гарантувати, що при одночасному доступі до бази даних багатьох користувачів подібних конфліктів не відбудеться.

#### 1.5. Сервіс відновлення

СУБД повинна надавати засоби відновлення бази даних на випадок будь-якого її ушкодження або руйнування.

Під час обговорення підтримки транзакцій згадувалося, що при збої транзакції база даних повинна бути повернута в несуперечливий стан. Подібний збій може відбутися в результаті виходу з ладу системи запам'ятовуючого пристрою, помилки апаратного чи програмного забезпечення, які можуть привести до зупинки СУБД. Крім того, користувач може знайти помилку під час виконання транзакції і зажадати її скасування. В усіх цих випадках СУБД повинна надати механізм відновлення бази даних і повернення її до несуперечливого стану.

#### 1.6. Сервіс контролю доступу до даних

СУБД повинна мати механізм, що гарантує можливість доступу до бази даних тільки санкціонованих користувачів.

Неважко привести приклади, коли потрібно сховати деякі збережені в базі дані зведення від інших користувачів. Наприклад, менеджерам відділень компаній можна було б надати всю інформацію, зв'язану з зарплатою співробітників, але бажано сховати її від інших користувачів. Крім того, базу даних варто було б захистити від будь-якого несанкціонованого доступу. Термін «**безпека**» відноситься до захисту бази даних від навмисного чи випадкового несанкціонованого доступу. Передбачається, що СУБД забезпечує механізми подібного захисту даних.

#### 1.7. Підтримка обміну даними

СУБД повинна мати здатність до інтеграції з комунікаційним програмним забезпеченням.

Більшість користувачів здійснюють доступ до бази даних за допомогою терміналів. Іноді ці термінали приєднані безпосередньо до комп'ютера із СУБД. В інших випадках термінали можуть знаходитися на значному віддаленні й обмінюватися даними з комп'ютером, на якому розташовується СУБД, через мережу. У будь-якому випадку СУБД одержує запити у вигляді повідомлень обміну даними і аналогічним чином відповідає на них. Усі спроби передачі даних керуються менеджером обміну даними. Будь-яка СУБД повинна мати здатність інтеграції з різноманітними існуючими менеджерами обміну даними. Навіть СУБД для персональних комп'ютерів повинні підтримувати роботу в локальній мережі, щоб замість декількох розрізнених баз даних для кожного окремого користувача можна було б установити одну централізовану базу даних і

використовувати її як загальний ресурс для всіх існуючих користувачів. При цьому передбачається, що не база даних повинна бути розподілена в мережі, а вилучені **користувачі повинні мати можливість доступу до централізованої бази даних**. Така топологія називається **розподіленою обробкою**.

#### **1.8. Служби підтримки цілісності даних**

СУБД повинна мати інструменти контролю за тим, щоб дані і їх зміни відповідали заданим правилам.

Цілісність бази даних означає коректність і несуперечність збережених даних. Вона може розглядатися як ще один тип захисту бази даних. Крім того, дане питання зв'язане з забезпеченням безпеки, тому воно має більш широкий зміст, оскільки цілісність зв'язана з якістю самих даних. Цілісність за звичай виражається у вигляді обмежень чи правил збереження несуперечності даних, що не повинні порушуватися в базі. Наприклад, можна вказати, що співробітник не може відповідати одночасно більш ніж за десять об'єктів нерухомості. У цьому випадку при спробі закріпити черговий об'єкт нерухомості за деяким співробітником, СУБД повинна перевірити, чи не перевищений встановлений ліміт, і у випадку виявлення подібного перевищення заборонити закріплення нового об'єкта за даним співробітником з відповідним повідомленням.

#### **1.9. Служби підтримки незалежності від даних**

СУБД повинна мати інструменти підтримки незалежності програм від фактичної структури бази даних.

Поняття незалежності від даних уже розглядалося в раніше. Звичайно вона досягається за рахунок реалізації механізму підтримки чи представлень підсхем. Фізична незалежність від даних досягається досить просто, тому що маємо кілька типів припустимих змін фізичних характеристик бази даних, що ніяк не впливають на відображення. Однак домогтися повної логічної незалежності від даних складніше. Як правило, система легко адаптується до додавання нового об'єкта, чи атрибута зв'язку, але не до їх видалення. У деяких системах узагалі забороняється вносити будь-як зміни у вже існуючі компоненти логічної схеми.

#### **1.10. Допоміжні служби**

СУБД повинна надавати деякий набір різних допоміжних служб. Допоміжні утиліти, як правило, призначені для надання допомоги АБД в ефективному адмініструванні бази даних. Нижче приводяться деякі приклади подібних утиліт:

**Утиліти імпортування**, призначені для завантаження бази даних із плоских файлів або в з іншої СУБД, а також утиліти експортування, що служать для вивантаження бази даних у плоскі файли або іншу СУБД.

**Засоби моніторингу**, призначені для відстеження характеристик функціонування і використання бази даних.

**Програми статистичного аналізу**, що дозволяють оцінити продуктивність чи ступінь використання бази даних.

**Інструменти реорганізації індексів**, призначені для перебудови індексів і обробки випадків їх переповнення.

**Інструменти збору «сміття» і перерозподілу пам'яті** для фізичного усунення вилучених записів із запам'ятовуючих пристроїв, об'єднання звільненого простору і перерозподілу пам'яті в разі потреби.

## **2. Архітектура багатокористувачевих СУБД**

У цьому розділі познайомимося з різними типовими архітектурними рішеннями, які використовуються при реалізації багатокористувачевих СУБД, а саме зі схемами **телеобробки, файловим сервером і технологією "клієнт/сервер"**.

### **2.1. Телеобробка**

Традиційною архітектурою багатокористувачевих систем раніш вважалася схема, що одержала назву "телеобробки", при якій один комп'ютер з єдиним процесором був з'єднаний з декількома терміналами. При цьому **вся обробка виконувалася в рамках**

*єдиного комп'ютера*, а приєднані до нього користувальницькі термінали були типовими "не інтелектуальними" пристроями, *не здатними функціонувати самостійно*. З центральним процесором термінали були зв'язані за допомогою кабелів, по яких вони посилали повідомлення користувальницьким додаткам (через підсистему керування обміном даними операційної системи).

У свою чергу, користувальницькі додатки зверталися до необхідних служб СУБД. У такий же спосіб повідомлення поверталися назад на користувальницький термінал. На жаль, при такій архітектурі основне і *надзвичайно велике навантаження покладалося на центральний комп'ютер, що повинний був виконувати не тільки дії прикладних програм і СУБД, але і значну роботу з обслуговування терміналів (наприклад, форматування даних, виведених на екрани терміналів)*.

В останні роки був досягнутий істотний прогрес у розробці високопродуктивних персональних комп'ютерів і мереж, які з них склалися.. Ця тенденція привела до *появи наступних двох типів архітектури СУБД*:

1. Технології **файлового сервера**.
2. Технології **"клієнт/сервер"**.

## **2.2. Файловий сервер**

У середовищі файлового сервера обробка даних розподілена в мережі (ЛОС). Файловий сервер містить файли, необхідні для роботи додатків і самої СУБД, Однак *користувальницькі програми і сама СУБД розміщені і функціонують на окремих робочих станціях, і звертаються до файлового сервера тільки в міру необхідності одержання доступу до потрібного їм файлу*. Таким чином, файловий сервер функціонує просто як *спільно використовуваний жорсткий диск*. СУБД на кожній робочій станції посилає запити файловому серверу по всім необхідним їй даним, що зберігаються на диску файл-сервера. Такий підхід характеризується значним мережним трафіком, що може привести до зниження продуктивності всієї системи в цілому. Розглянемо, наприклад, ситуацію, коли користувач надсилає запит на вибірку даних про всіх співробітників відділення компанії, що знаходиться за певною адресою. Цю задачу можна сформулювати за допомогою відповідного SQL-оператора.

Оскільки файловий сервер не розуміє команд мовою SQL, СУБД повинна запросити у файлового сервера файли, що відповідають відношенням **Відділення і Працівник**, а не шукані імена співробітників. Таким чином, архітектура з використанням файлового сервера володіє *наступними основними недоліками*.

1. Великий обсяг мережного трафіку.
2. На кожній робочій станції повинна знаходитися повна копія СУБД.
3. Керування паралельністю, відновленням і цілісністю ускладнюється, оскільки доступ до тим самим файлів можуть здійснювати відразу кілька екземплярів СУБД.

## **2.3. Технологія „клієнт/сервер"**

Технологія "клієнт/сервер" була розроблена з метою усунення недоліків, що мають у перших двох підходах. "Клієнт/сервер" означає такий спосіб взаємодії програмних компонентів, при якому вони утворюють єдину, систему. Як видно з "самої назви, існує деякий **клієнтський** процес, що вимагає визначених ресурсів, а також **серверний** процес, який ці ресурси надає. При цьому зовсім необов'язково, щоб вони знаходилися на тому самому комп'ютері. *На практиці прийнято розміщати сервер на одному вузлі локальної мережі, а клієнти — на інших вузлах*.

У контексті бази даних клієнт керує користувальницьким інтерфейсом і логікою додатка, діючи як складна робоча станція, на якій виконуються додатки баз даних. Опишемо послідовність кроків для виконання звичайного запиту у такій системі:

1. Клієнт приймає від користувача запит, перевіряє синтаксис і генерує запит до бази даних мовою SQL або іншою мовою бази даних, що відповідає логіці додатка.
2. Потім він передає повідомлення серверу, очікує надходження відповіді і форматує отримані дані для представлення їх користувачу.

3. Сервер приймає й обробляє запити до бази даних, а потім передає отримані результати назад клієнту. Така обробка включає перевірку повноважень клієнта, забезпечення вимог цілісності, підтримку системного каталогу, а також виконання запиту і відновлення даних. Крім цього, підтримується керування паралельністю і відновленням.

У таблиці 2 наведено функції, які виконують сервер та клієнт у даній системі.

**Таблиця 2**

<b>Функції учасників в середовищі „клієнт/сервер”</b>	
<b>Клієнт</b>	<b>Сервер</b>
Керує користувальницьким інтерфейсом	Приймає й обробляє запити до бази даних з боку клієнтів
Приймає і перевіряє синтаксис уведеного користувачем запиту	Перевіряє повноваження користувачів
Виконує програму	Гарантує дотримання обмежень цілісності
Генерує запит до бази даних і передає серверу	Виконує запити/відновлення і повертає результати клієнту
Відображає отримані дані користувачу	Підтримує системний каталог
Забезпечує рівнобіжний доступ до бази даних	
Забезпечує керування відновленням	

Даний тип архітектури має приведені нижче переваги:

1. Забезпечується більш широкий доступ до існуючої бази даних.
2. Підвищується загальна продуктивність системи. Оскільки клієнти і сервер знаходяться на різних комп'ютерах, їхні процесори здатні виконувати додатки паралельно. При цьому настроювання продуктивності комп'ютера із сервером спрощується, якщо на ньому виконується тільки робота з базою даних.
3. Вартість апаратного забезпечення знижується. Досить потужний комп'ютер з великим пристроєм збереження потрібний тільки серверу — для збереження і керування базою даних.
4. Скорочуються комунікаційні витрати. Додатки виконують частину операцій на клієнтських комп'ютерах і посилають через мережу тільки запити до бази даних, що дозволяє істотно скоротити обсяг даних, які пересилаються по мережі.
5. Підвищується рівень несуперечності даних. Сервер може самостійно керувати перевіркою цілісності даних, оскільки всі обмеження визначаються і перевіряються тільки в одному місці. При цьому кожному додатку не придется виконувати власну перевірку.
6. Ця архітектура дуже природно відображається на архітектуру відкритих систем.

У *трьохрівневій* архітектурі "клієнт/сервер" *тонкий* (не інтелектуальний) клієнт на робочій станції керує тільки користувальницьким інтерфейсом, тоді як середній (*другий*) рівень обробки даних керує всією іншою логікою додатка. *Третім* рівнем тут є сервер бази даних. Ця трьохрівнева архітектура виявилася більш придатною для деяких середовищ — наприклад, для мереж **Internet** і **Intranet**, де як клієнта може використовуватися звичайний **Web-броузер**.

### **3. Системний каталог**

СУБД повинна мати доступний кінцевому користувачу системний каталог чи словник даних. У цьому розділі ми познайомимося із системними каталогами більш докладно.

СИСТЕМНИЙ КАТАЛОГ - СХОВИЩЕ ДАНИХ, ЩО ОПИСУЮТЬ ІНФОРМАЦІЮ, ЯКА ЗБЕРІГАЄТЬСЯ В БАЗІ ДАНИХ, ТОБТО МЕТА-ДАНІ, ЧИ "ДАНІ ПРО ДАНІ".

Системний каталог СУБД є одним з фундаментальних компонентів системи. Багато програмних компонентів будуються на використанні даних, що зберігаються в системному каталозі. Наприклад, модуль контролю прав доступу використовує системний



каталог для перевірки наявності в користувача повноважень, необхідних для виконання операцій, які запитані ним. Для проведення подібної перевірки системний каталог повинний включати наступні компоненти:

- імена користувачів, яким дозволений доступ до бази даних;
- імена елементів даних у базі даних;
- елементи даних, до яких кожен користувач має право доступу, і дозволені типи доступу до них - для вставки, відновлення, чи видалення.

Іншим прикладом можуть служити засоби перевірки цілісності даних, що використовують системний каталог для перевірки того, чи задовольняє запитана операція усім встановленим обмеженням підтримки цілісності даних. Для виконання цієї перевірки в системному каталозі повинні зберігатися такі зведення:

- імена елементів даних з бази даних;
- типи і розміри елементів даних;
- обмеження, установлені для кожного з елементів даних.

Як уже згадувалося раніше, термін "словник даних" часто використовується для програмного забезпечення більш загального типу, чим просто каталог СУБД. Система словника даних може бути або пасивною, або активною.

**Активна** система завжди погодиться зі структурою бази даних, оскільки вона автоматично підтримується цією системою.

**Пасивна** система може суперечити стану бази даних через ініційованих користувачами зміни.

### 3.1. Служба IRDS

СЛУЖБА СЛОВНИКА ІНФОРМАЦІЙНИХ РЕСУРСІВ IRDS (INFORMATION RESOURCE DICTIONARY SYSTEM - IRDS) являє собою програмний інструмент, призначений для КЕРУВАННЯ ІНФОРМАЦІЙНИМИ РЕСУРСАМИ ОРГАНІЗАЦІЇ, А ТАКОЖ ДЛЯ ЇХ ДОКУМЕНТУВАННЯ.

Вона включає визначення таблиць, що містять словник даних, і операції, що можуть бути використані для доступу до цих таблиць. Згадані операції забезпечують несуперечливий метод доступу до словника даних і спосіб перетворення визначень даних зі словника одного типу у визначення іншого типу. Іншими словами, саме вона бере на себе функції роботи з словником даних.

Визначення IRDS були прийняті як стандарт Міжнародною організацією стандартизації **ISO (International Organization for Standardization)** (ISO, 1990; 1993).

Стандарти IRDS визначають набір правил збереження інформації в словнику даних і доступу до неї, переслідуючи при цьому три наступні цілі:

- розширюваність даних;
- цілісність даних;
- контрольований доступ до даних.

Служба IRDS побудована на використанні інтерфейсу сервісів, що складається з набору функцій, доступного для виклику з метою одержання доступу до словника даних. Інтерфейс сервісів може викликатися з боку таких типів користувацьких інтерфейсів, як:

- панельний інтерфейс;
- командна мова;
- файли експорту/імпорту;
- прикладні програми.

**Панельний інтерфейс** складається з набору панелей чи екранів, кожний з яких надає доступ до заздалегідь описаного набору сервісів. Він трохи нагадує мову QBE і дозволяє користувачам переглядати і змінювати словник даних.

**Інтерфейс командної мови** складається з набору команд чи операторів, що дозволяють виконувати маніпуляції зі словником даних. Інтерфейс командної мови може

викликатися інтерактивно (за допомогою термінала) чи за допомогою впровадження операторів у програмі на мовах програмування високого рівня.

**Інтерфейс експорту/імпорту** генерує файл, якому можна передавати між різними IRDS-сумісними системами. Стандарт IRDS визначає й універсальний формат обміну інформацією. Причому він не вимагає, щоб база даних, що лежить в основі словника даних, відповідала якійсь однієї моделі даних.

#### **Питання для контролю**

1. Перелічіть основні функції СУБД. Яка функція СУБД вважається головною?
2. Як Ви розумієте каталог, який доступний кінцевим користувачам? Яка інформація зберігається в ньому?
3. Що таке транзакція і в чому полягає функція керування нею?
4. Опишіть, як працює сервіс керування паралельністю.
5. Що таке відновлення даних у СУБД?
6. Для чого потрібен сервіс контролю доступу до даних?
7. У чому сутність підтримки обміну даних?
8. Яку функцію виконує служба підтримки цілісності даних?
9. Що таке незалежність від даних і як працює відповідна служба?
10. Перелічіть основні допоміжні служби та дайте їх стислий опис.
11. Які типи архітектури багатокористувачевих СУБД існують і у чому їх відмінності?
12. Поясніть, як Ви розумієте поняття «Телеобробка»?
13. Опишіть у стислому вигляді принципи роботи файлового серверу. У чому недоліки даної архітектури?
14. Яке переваги надає використання топології «клієнт/сервер»?
15. Перелічіть, які основні операції виконуються з боку серверної частини топології «клієнт/сервер»?
16. Що таке системний каталог и у чому полягають його основні функції?
17. Навіщо потрібна служба словника інформаційних ресурсів та з яких сервісів вона складається?