УДК 681.3

М. М. ІВАНЮК
Хмельницький національний університет

ТЕХНОЛОГІЯ OPEN DATABASE CONNECTIVITY (ODBC)

В статті висвітлюється єдиний інтерфейс для доступу до змішаних баз даних SQL.
The only interface for the admittance to the farraginous base data SQL is explained in this article.

Стандарт ODBC

До цього часу мову SQL використовували інтерактивно, тобто способом введення з терміналу окремих операторів. Окрім цього способу існує програмний спосіб використання мови SQL, способом впровадження SQL-операторів в програми на процедурних мовах програмування. Альтернативним підходом, який використовується деякими розробниками СУБД, є надання в розпорядження програміста бібліотеки стандартних функцій, які можуть викликатися із створених ними прикладних програм. Для багатьох програмістів використання різних бібліотек стандартних функцій є звичайною практикою, тому вони оцінюють API як більш зручний спосіб роботи з SQL.

Прикладний API включає набір бібліотечних функцій, що забезпечують програміста різноманітними типами доступу до бази даних, такими як підключення, виконання різних SQL-операторів, вибірка окремих рядків даних з результуючих таблиць запитів і т.д. Єдиним недоліком подібного підходу є відсутність універсальності, тобто програма обов'язково повинна бути оброблена передкомпілятором і зв'язана з бібліотекою API, що поставляється в складі конкретної цільової СУБД. При необхідності використання цієї ж програми в середовищі іншої СУБД буде потрібно, як мінімум, виконати її обробку новим передкомпілятором і зв'язати бібліотеку API з новою СУБД. З тими ж проблемами зіштовхуються і незалежні розроблювачі програмного забезпечення, які зазвичай змушені писати окремі версії свого додатку для кожної з цільових СУБД, з якими даний додаток планується використовувати. Як правило, це пов'язано з додатковою витратою чималих ресурсів, що витрачаються на розробку і супровід програмного забезпечення, специфічного для окремих типів цільових СУБД, а не власне самого створюваного додатка.

Щоб впорядкувати даний підхід, фірма Microsoft розробила стандарт, що одержав назву Open Database Connectivity – ODBC. Технологія ODBC передбачає використання єдиного інтерфейсу для доступу до змішаних баз даних SQL, причому мова SQL розглядається як базовий стандартний засіб доступу до даних. Даний інтерфейс (який вбудовується безпосередньо в мову С) забезпечує високий ступінь універсальності, у результаті чого один і той самий додаток може одержувати доступ до даних, що зберігаються в базах даних різних цільових СУБД, без необхідності внесення змін у його програмний текст. Таким чином, розроблювачі одержали інструмент, що дозволяє створювати і поширювати додатки архітектури "клієнт/сервер", здатні працювати із широким спектром різних цільових СУБД. Для зв'язку додатка з будь-якою обраною користувачем цільовою СУБД досить лише мати відповідний ODBC-драйвер.

В даний час технологія ODBC фактично придбала значення галузевого стандарту. Основною причиною популярності цієї технології є її гнучкість, що забезпечує розроблювачів наступними перевагами:

1) додатки більше не зв'язані з прикладним API якоїсь однієї конкретної СУБД;
2) SQL-оператори можуть включатися у вихідний текст додатка або динамічно створюватися безпосередньо під час виконання програми;
3) додаток може ігнорувати особливості використовуваних протоколів передачі даних;
4) дані можуть посилатися і доставлятися у тому форматі, що є найбільш зручним для даного додатку;
5) засоби підтримки ODBC розроблені з урахуванням вимог стандартів X/Open і CLI (Call-Level Interface);
6) драйвери ODBC існують для більш ніж п'ятдесяти різних типів найпоширеніших СУБД.

В інтерфейс ODBC включені наведені нижче елементи: 1) бібліотека функцій, виклик яких дозволяє додатку підключатися до бази даних, виконувати SQL-оператори і витягати інформацію з результуючих наборів даних; 2) стандартний метод підключення і реєстрації в СУБД; 3) стандартне представлення для даних різних типів; 4) стандартний набір кодів помилок; 5) типовий синтаксис SQL-операторів, побудований на використанні специфікацій X/Open і ISO CLI.

Загальна архітектура ODBC включає чотири елементи.

Додаток виконує обробку даних, виклик функцій бібліотеки ODBC для відправлення SQL-операторів у СУБД і вибірку інформації.

Менеджер драйверів виконує завантаження драйверів за вимогою додатка. Менеджер драйверів був розроблений компанією Microsoft і являє собою бібліотеку DLL.

Драйвери й агенти баз даних обробляють виклики функцій ODBC і направляють SQL-запити в конкретні джерела даних, а також повертають отримані результати додатку. За необхідності драйвери виконують модифікацію вихідного запиту додатка з метою приведення його у відповідність синтаксичним вимогам цільового СУБД. Драйвери можуть надавати тільки ті можливості, що забезпечуються цільовою СУБД. Від них не потрібно власної реалізації тих можливостей, що дана СУБД не підтримує. Наприклад, якщо

цільова СУБД не підтримує операції відкритого з'єднання, то ця функція не буде підтримуватися ODBC-драйвером. Єдиним важливим виключенням з цього правила є драйвери для СУБД, що не мають власних ядер, наприклад, таких як Xbase. У цьому випадку ядро СУБД, що забезпечує хоча б мінімальну підтримку мови SQL, повинне бути реалізоване в самому драйвері.

В архітектурному рішенні з використанням декількох ODBC-драйверів (рис. 1) усі згадані вище задачі повинні вирішуватися самим ODBC-драйвером і використовувати агенти баз даних не потрібно.

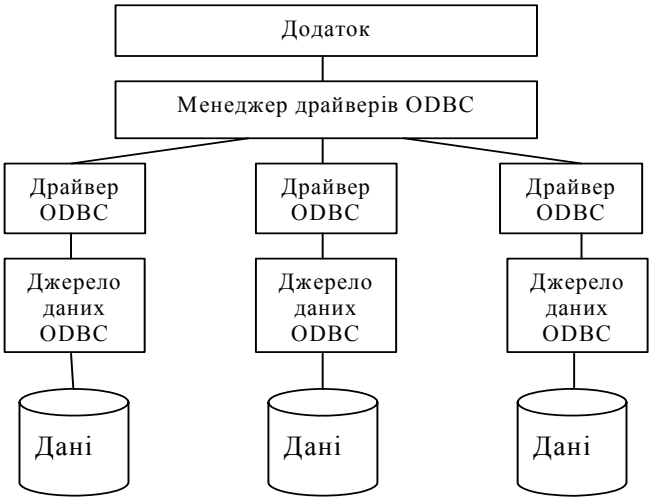

Рис. 1. Архітектура ODBC при використанні декількох драйверів

У випадку використання єдиного ODBC-драйвера (рис. 2) для кожного з типів СУБД буде потрібне застосування агентів бази даних, розташованих на стороні сервера. При обробці запитів на доступ до бази даних ці агенти тісно співпрацюють з ODBC-драйвером, розташованим на стороні клієнта. У середовищі Windows єдиний ODBC-драйвер реалізований у виді бібліотеки DLL. Агенти баз даних реалізуються як процеси-демони, виконувані на сервері з цільовою СУБД.

Джерела даних. Цей компонент містить ті дані, доступ до яких необхідний користувачу додатка. Дані зберігаються в базі даних, контрольовані цільовою СУБД, операційною системою, а також мережною операційною системою, якщо така використовується.

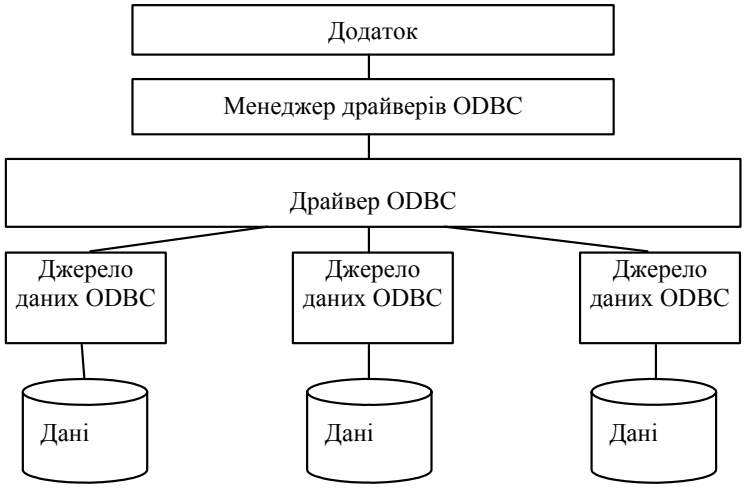

Рис. 2. Архітектура ODBC при використанні єдиного драйвера

Архітектура ODBC визначає для драйверів два різних рівні відповідності: рівень ODBC API і рівень граматики ODBC SQL. Обмежимося розглядом рівня граматики ODBC SQL. Для ознайомлення з рівнем ODBC API потрібно звернутись до документа "Microsoft ODBC Reference Guide". У стандарті ODBC визначається граматичне ядро, що відповідає специфікаціям X/Open CAE (1992) і ISO CLI (1995). Більш ранні версії ODBC були побудовані на попередніх версіях цих специфікацій і не включали їхньої повної реалізації. У стандарті ODBC 3.0 цілком реалізовані обидві ці специфікації і додатково додані функції, що зазвичай використовуються розроблювачами в інтерактивних додатках баз даних.

Стандарт ODBC також включає визначення мінімального рівня граматики, що відповідає базовому рівню відповідності вимогам ODBC, а також визначення розширеної граматики, що включає загальноприйняті розширення мови SQL, реалізовані в різних СУБД.

Мінімальний рівень підтримки мови SQL:

- Оператори мови визначення даних (Data Definition Language — DDL):
CREATE TABLE і DROP TABLE.
- Оператори мови маніпулювання даними (Data Manipulation Language — DML): SELECT, INSERT, UPDATE SEARCHED і DELETE SEARCHED.
- Вирази: найпростіші (наприклад, A > B + C).
- Типи даних: CHAR, VARCHAR або LONG VARCHAR.
Основний рівень підтримки мови SQL:
- Мінімальний рівень підтримки граматики мови SQL і типів даних.
- Мова DDL: оператори ALTER TABLE, CREATE INDEX, DROP INDEX, CREATE VIEW, DROP VIEW, GRANT і REVOKE.
- Мова DML: усі можливості оператора SELECT.
Вирази: підзапити, що узагальнюють функції, наприклад, SUM і MIN.
- Типи даних: DECIMAL, NUMERIC, SMALLINT, INTEGER, REAL, FLOAT, DOUBLE PRECISION.
Розширений рівень підтримки мови SQL:
- Мінімальний і основний рівні підтримки граматики мови SQL і типів даних.
- Мова DML: відкриті з'єднання, позиціоновані оператори UPDATE, позиціоновані оператори DELETE, оператор SELECT FOR UPDATE і підтримка об'єднань.
- Вираження: скалярні функції (наприклад, SUBSTRING і ABS), функції дати, часу і літерали тимчасових оцінок.
- Типи даних: BIT, TINYINT, BIGINT, BINARY, VARBINARY, LONG VARBINARY, DATE, TIME, TIMESTAMP.
- Пакети SQL-операторів.
- Виклики процедур.

Приклад організації доступу до БД сервера MySQL з СУБД MS Access

Для виконання роботи необхідно, щоб на комп'ютері були встановлені такі додатки: **Администратор источников данных ODBC** та **MYSQL ODBS Driver**.

Насамперед після встановлення драйвера потрібно встановити в джерелах даних користувача ODBC нове джерело доступу до бази даних. Для цього, виконуємо наступні дії:

1. Відкриваємо вікно **Источники данных** (ODBC), яке знаходиться за таким шляхом *Пуск / Настройка / Панель управления / Администрирование / Источники данных (ODBC)*. В даному вікні переходимо на вкладку **Пользовательский DSN** (рис. 3).

2. Додаємо нове джерело даних, натиснувши кнопку **Добавить**…, після чого з'являється вікно **Создание нового источника данных** (рис. 4).

3. У цьому вікні вибираємо ім'я драйвера MySQL ODBC 3.51 Driver (або просто MySQL) і натискаємо кнопку **Готово**, після чого з'являється вікно MySQL ODBC 3.51 Driver - DNS Configuration (рис. 5).

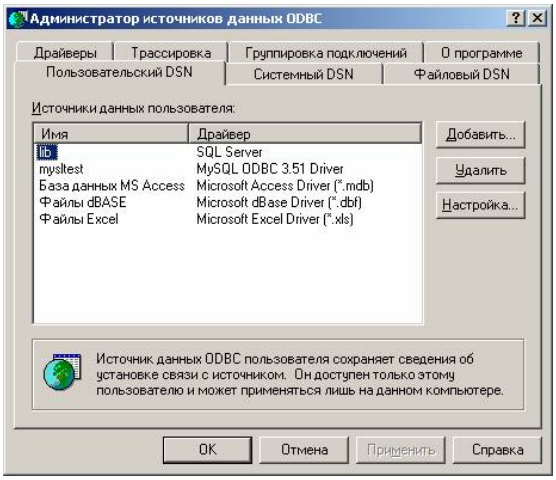

Рис. 3. Вікно "Администратор"

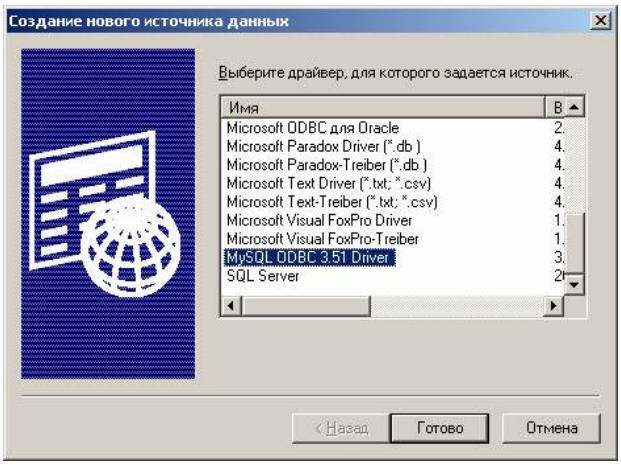

Рис. 4. Вікно "Создание нового источника данных"

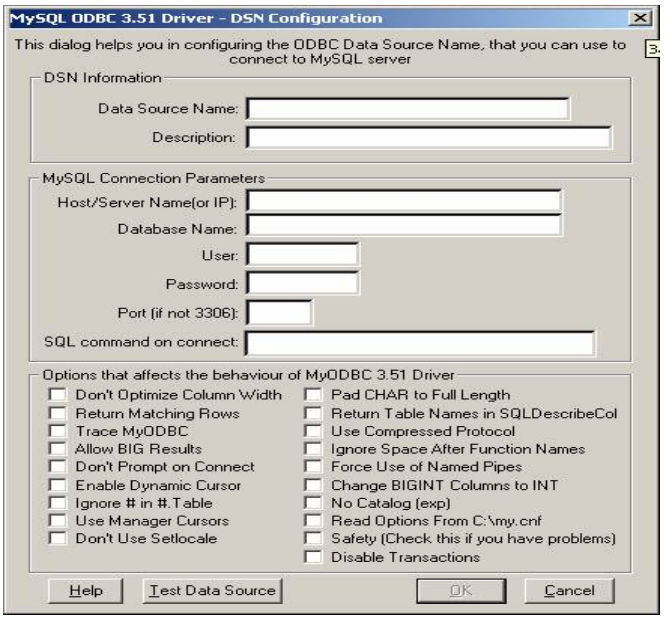

Рис. 5. Вікно "MySQL ODBC 3.51 Driver – DNS Configuration"

4. Для задання параметрів джерела даних в цьому вікні необхідно заповнити наступні поля:

1. Data Source Name. В цьому полі потрібно ввести ім'я джерела даних, з яким ви маєте працювати, наприклад, bdek000 (ім'я може бути довільним, але бажано латинськими літерами).

2. Host/Server Name(or IP). В цьому полі необхідно написати ім'я сервера (або його IP адресу), на якому працює сервер MySQL, де створена база даних. В нашому випадку встановимо IP адресу 172.20.1.83.

3. Database Name. В цьому полі необхідно ввести ім'я бази даних, з якою ви працюєте. В нашому випадку кожний користувач вводить ім'я своєї бази даних (користувач ek001 вводить ek001, користувач ek002 вводить ek002 і т.д.). В інші поля інформацію вводити поки що не потрібно.

В противному випадку з'являється вікно-повідомлення, представлене на рис. 6.

5. Після запису даних в відповідні поля натискуємо кнопку Test Data Source. Якщо немає ніякої помилки, то з'являється вікно повідомлення подане на рис. 7.

В цьому випадку потрібно натиснути кнопку ОК і перевірити правильність введення даних.

6. В разі успішного виконання попереднього пункту, після натиснення ОК з'являється новий запис у вікні **Администратор источников данных ODBC** (рис. 8).

На цьому етапі підключення нового джерела даних закінчується і можна приступати до створення бази даних в MS Access, яка б оперувала з даними, що зберігаються на сервері баз даних MySQL.

7. Запускаємо MS Access і створюємо базу даних з довільним ім'ям (в якості імені можемо обрати встановлений нами псевдонім – bdek000).

8. У вікні маніпулювання об'єктами бази даних натискаємо праву кнопку миші та обираємо пункт **Связь с таблицями**, як показано на рис. 9.

9. У вікні **Связь** (рис. 10), яке з'являється після виконання попереднього пункту, в полі **тип файлів** вибираємо тип ODBC Databases.

10. У вікні **Выбор источника данных**, на вкладці **Источник данных компьютера** вибираємо

встановлене джерело даних. В нашому випадку це джерело носить назву bdek000 (рис. 11).

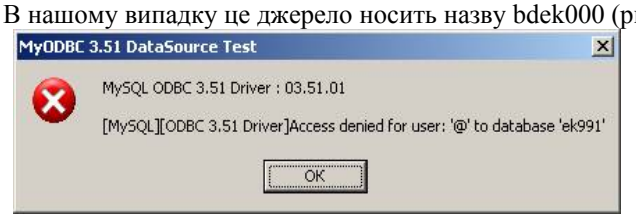

Рис. 6. Повідомлення про неуспішне підключення до БД

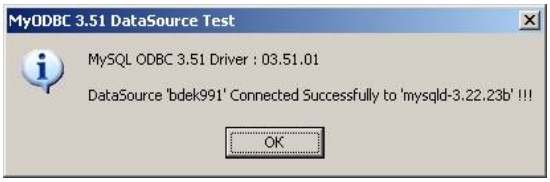

Рис. 7. Повідомлення про успішне підключення до БД

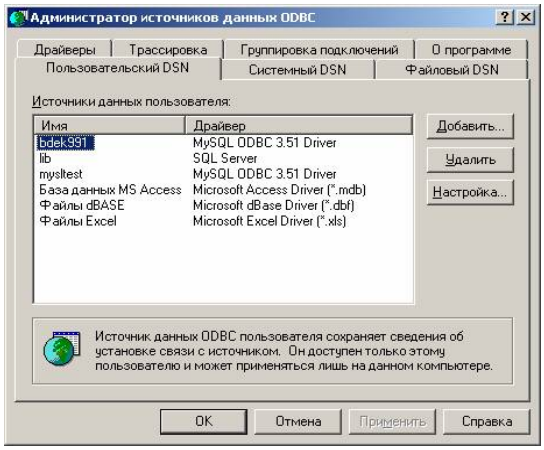

Рис. 8. Вікно "Администратор источников даных ODBC"

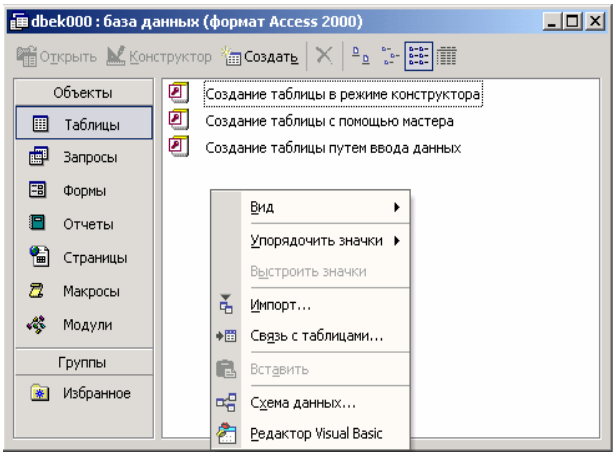

Рис. 9. Вікно маніпулювання об'єктами БД

11. Після вибору джерела даних з'являється вікно **Связь с таблицами** (рис. 12), в ньому представлені таблиці, серед яких потрібно вибрати таблицю для зв'язку. Вибираємо таблицю bd1 і натискаємо OK. Після цього таблиця bd1 з'являється у вікні маніпулювання об'єктами бази даних (рис. 13). Всі подальші дії з таблицею тепер можна проводити засобами середовища MS Access, дана таблиця буде сприйматися як таблиця, створена засобами MS Access, проте всі внесені змінні будуть записуватись не в файл bdek000.mdb, а в базу даних, записану на сервері MySQL.

12. Окрім маніпулювання даними, які знаходяться на сервері БД, засобами MS Access, можна також імпортувати дані із MySQL в MS Access. З імпортованими даними можна працювати як з локальними таблицями, дані при цьому на сервері не будуть поновлюватись, але будуть зберігатись в файлі бази даних MS Access. MS Access буде розуміти імпортовані таблиці як таблиці створені його ж засобами.

Процедура імпортування даних із MySQL в MS Access майже аналогічна, як і для зв'язку з таблицями. Відмінність полягає тільки в тому, що замість пункту **Связь с таблицами**, потрібно вибрати пункт **Импорт**

(див. рис. 9). Всі інші дії виконуються, як і для зв’язку з таблицями.

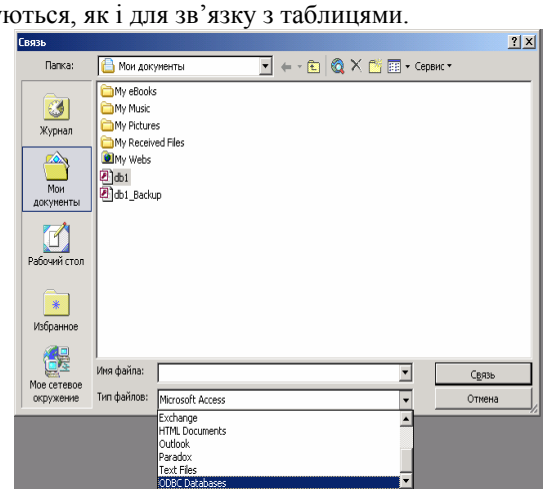

Рис. 10. Вікно “Связь”

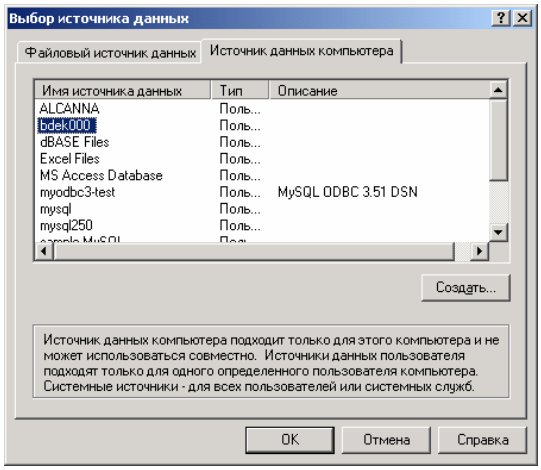

Рис. 11. Вікно “Выбор источника данных”

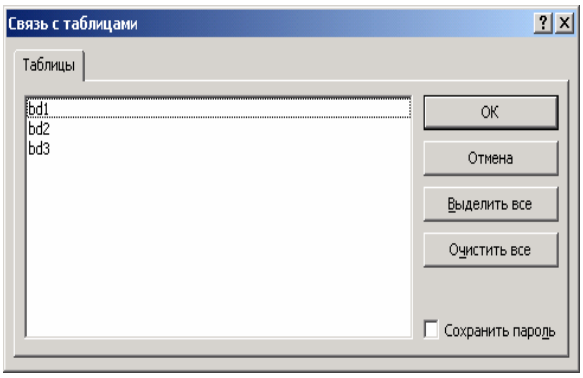

Рис. 12. Вікно “Связь с таблицами”

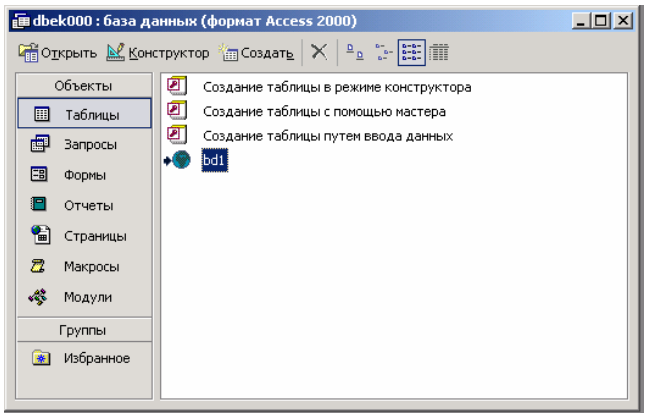

Рис. 13. Вікно маніпулювання об'єктами

Список використаних джерел

1. Автоматизированные информационные технологии в экономике : [учебник / под ред. Г. А. Титоренко]. – М. : Комоютер, ЮНИТИ, 1998. – 400 с.
2. Бойко В. В. Проектирование информационной автоматизированной системы на основе СУБД / В. В. Бойко, В. М. Савинков. – М. : Финансы и статистика, 1982.
3. Гетц К. Программирование в Microsoft Office: Полное руководство по VBA / К. Гетц, М. Джильберт. – К. : Издательская группа BHV, 2007.
4. Дейт К. Введение в системы баз данных / Дейт К. – М. : Наука, 1998. – 520 с.
5. Єрьоміна Н. В. Проектування баз даних / Єрьоміна Н. В. – К. : КНЕУ, 1998. – 208 с.