Министерство образования и науки Российской Федерации
Федеральное агентство по образованию
Государственное образовательное учреждение высшего профессионального образования
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. Н.Э. БАУМАНА
(МГТУ им. Н.Э. Баумана)
Факультет «Информатика и системы управления»
Кафедра «Компьютерные системы и сети»

Брешенков А.В., Волкова Е.А., Галямова Е.В.

Знакомство с СУБД DB2. Язык DDL

Методические указания
для выполнения лабораторной работы по курсу
«СЕТЕВЫЕ БАЗЫ ДАННЫХ»

2009 год,
Москва

Лабораторная работа «**Знакомство с СУБД DB2. Язык DDL»**

Аннотация

Система управления базами данных DB2 (СУБД DB2) компании IBM — это мощная высокопроизводительная многоплатформенная СУБД. Она поставляется с целым набором клиентских инструментов, которые могут быть использованы для выполнения операторов языка структурированных запросов SQL.

База данных DB2 для z/OS отличается высокой степенью готовности, масштабируемостью и производительностью при обработке транзакций, в неё включена поддержка очень больших баз данных благодаря поддержке 64-разрядных виртуальных систем хранения. Однако существуют некоторые специфические особенности работы с СУБД на больших ЭВМ класса IBM мэйнфрейм System z, которым и посвящено данное методическое указание.

В данном сборнике изложены основы реализации баз данных DB2 на мэйнфрейме. Дано краткое описание архитектуры СУБД DB2,компонентов языка SQL, способов доступа к DB2 для z/OS, а также процесса выполнения скриптов. В практической части приведены примеры sql скриптов и указаны настройки утилиты SPUFI для их запуска.

Цели лабораторной работы:

- Изучение интерфейса ISPF (Interactive System Productivity Facility) для работы с СУБД DB2;
- использование утилиты SPUFI (SQL Processor Using File Input) для выполнения скриптов;
- создание и заполнение объектов DB2;
- изучение влияния ссылочной целостности с использованием утилиты SPUFI.

1. Теоретическая часть

1.1 База данных DB2. Общие сведения.

Система управления реляционными базами данных DB2 (реляционная СУБД, Relational Database Management System - RDBMS) является составной частью семейства продуктов IBM Information Management. Считается, что DB2 была первой реляционной СУБД, в которой начали использовать SQL (язык, который обеспечивает интерфейс к реляционным СУБД). В ней поддерживаются все 3 категории языка SQL:

- Язык DML – Data Manipulation Language – используется для чтения и изменения данных. Используются 4 оператора: SELECT, UPDATE, INSERT, DELETE;

- Язык DDL – Data Definition Language – используется для создания, изменения и удаления объектов. Используются 3 оператора: CREATE, ALTER, DROP;

- Язык DCL – Data Control Language – используется для выдачи и отмены авторизации, имеет 2 оператора: GRANT и REVOKE.

Управление базой возможно из разных точек. Если она установлена на ОС Windows, Linux или Unix, то она имеет графический интерфейс, который в свою очередь, имеет удобный конструктор запросов и окно просмотра содержимого базы, также в графическом интерфейсе предусмотрен и ручной ввод запросов на языке SQL. На этих ОС предусмотрен и вариант работы через командную строку, который также позволяет вводить запросы, просматривать журналы базы и производить ее обслуживание – архивацию, резервное копирование и т.п.

Если база установлена на мейнфрейм, то в z/OS предусмотрена целая подсистема работы с DB2 – SPUFI (Главное меню интерфейса SPUFI показано на рисунке 1).

Логическая организация данных также стандартная: база данных, а в ней таблицы. В таблице – поля и ключевые поля для объединения таблиц. У каждой таблицы есть идентификатор – схема, позволяющая группировать таблицы по их владельцам. На физическом уровне все гораздо сложнее, поэтому описанию физического устройства DB2 посвящен раздел 1.2.

Сервер базы данных может быть установлен как на локальной машине, так и на удаленной. Доступ осуществляется через сетевые протоколы (если используется ОС Windows, то это протокол TCP/IP, порт 50000, а если используется ОС z/OS – то это протоколы SNA и TCP/IP, порт 446).

К системным ресурсам DB2 требовательна, хотя работа в целом зависит от поставленной задачи. Наилучшая производительность достигается путем установки на мейнфрейм.

1.2 Физическое устройство DB2

Элементы физического уровня базы данных DB2 можно разделить на 2 группы:

- Структуры данных: используются для организации данных пользователя и

- Структуры системы: управляются самой DB2.

Рассмотрим подробнее структуры данных и структуры системы, специфичные для DB2.

1.2.1 Структуры данных DB2

Структуры данных также можно разбить на 2 типа:

- Базовые структуры (виды, табличные и индексные пространства, группы жестких дисков);

- Структуры схем (типы данных, функции, триггеры, большие объекты и хранимые процедуры).

Структуры схем – это новый тип объектов, который ввели в мейнфрейм для совместимости с семейством DB2 для ОС Windows, Linux и Unix. Схема – это логическое объединение этих новых объектов.

Базовые структуры

Виды (Views)

Вид является альтернативным способом представления данных, содержащихся в одной или нескольких таблицах. Можно создать вид таблицы, в котором видна часть полей и разрешить работу и их изменение одной группе пользователей. Таким образом, эти пользователи не имеют доступа ко всей таблице, а могут видеть и работать только с конкретным видом таблицы. Таким образом, виды используются в целях безопасности. Также можно обезопасить работу с базой, запрещая полностью прямую работу с таблицами – для этого работа с базой и таблицами ведется только с использованием видов. Вид можно использовать для упрощения сложных запросов для начинающих пользователей – запрос к реальной базе будет создан автоматически – пользователь об этом даже не узнает.

Табличные пространства (Table spaces)

Таблица – это логическая структура. Она хранится в физическом наборе данных (data set) – в табличном пространстве. Табличное пространство – структура на физическом диске, может хранить одну или несколько таблиц. Имя табличного пространства состоит из имени базы данных и из непосредственно имени табличного пространства: PAYROLL.ACCNT_RECV (PAYROLL – имя базы данных, а ACCNT_RECV – имя табличного пространства). Табличные пространства бывают 3х типов: Простые (Simple), Сегментированные (Segmented) и Библиотечные (Partitioned).

База данных DB2 использует наборы данных с виртуальным методом

доступа VSAM, т.е. каждый сегмент – это набор данных VSAM.

Индексные пространства (Index Spaces)

Индексное пространство – другая структура данных, которая хранит один индекс. Стоит отметить, что при создании индекса DB2 сама создает для него индексное пространство.

Группы жестких дисков (Storage Groups)

Хранят набор томов на дисках DASD, которые, в свою очередь, непосредственно хранят таблицы и индексы.

На рисунке 2 приведена иерархия структур данных, иллюстрирующая все вышесказанное.

Структуры схем

Тип данных, определяемый пользователем (User-Defined Data Type – UDT)

С помощью UDT можно создавать пользовательские типы данных, если есть такая необходимость. Однако, UDT базируются на существующих типах. Пример – валюта, можно создать тип EURO, основанный на типе decimal – десятичное число и тип US_DOLLAR. В результате мы не можем складывать евро с долларами.

Функция, определяемая пользователем (User-Defined Function – UDF)

Функцию UDF можно легко задать на основе уже существующих функций базы DB2, таких как округление, определение среднего значения и т.п. или функцию можно написать с использованием SQL-запроса, это может быть даже небольшое приложение. Продолжая прошлый пример можно написать функцию конвертации валюты.

Триггер (Trigger)

Триггер определяет набор действий, которые должны быть выполнены по наступлению события – при вставке строки, при изменении

таблицы и т.п. Это фактически макрос (аналогичен макросу MS ACCESS)

Большой объект (Large Object – LOB)

Этот тип предназначен для обработки неструктурированных данных. Существуют 3 типа больших объектов:

- Двоичные – для мультимедиа данных;
- Символьные – для больших текстовых документов;
- Двухбайтовые символьные (Double Byte Character) – для больших текстовых документов, написанных на азиатских языках (2 байта на символ).

Большие объекты хранятся в специальных дополнительных таблицах и используются специальные пространства – LOB Table Spaces. При создании строки– большого объекта в таблице базы данных – происходит автоматическая ассоциация с таблицей и табличным пространством LOB.

Хранимые процедуры (Stored Procedures)

Хранимая процедура – пользовательское приложение для работы с базой данных, которое обычно хранится и выполняется на сервере (также может выполняться для локальных целей). Разработаны специально для клиент-серверной технологии. Программа хранится на сервере, а клиент ее только вызывает. Программа взаимодействует с базой и возвращает результат клиенту. Таким образом, снижается сетевой трафик – передается меньше запросов и т.п.

1.2.2 Системные структуры

Каталог и директория

База данных DB2 самостоятельно обслуживает набор таблиц, которые хранят служебные метаданные – данные обо всех объектах подсистемы DB2. Каталог хранит информацию обо всех объектах: таблицах, видах, индексах и т.п. Директория же хранит информацию о

приложениях. Каталогу можно отправить запрос на получение информации об объектах, а директории нет.

При создании пользовательской таблицы, база автоматически записывает все атрибуты – имя таблицы, владельца, табличное пространство и т.п. и сохраняет в таблицу каталога SYSIBM.SYSTABLES. Все поля таблицы автоматически записываются в таблицу SYSIBM.SYSCOLUMNS. Список авторизованных пользователей, способных просматривать и изменять таблицу, хранится в таблице SYSIBM.SYSTABAUTH. Все индексы пользовательской таблицы записываются в таблицу SYSIBM.SYSINDEXES.

Пулы буферов

Предназначены для временного хранения страниц табличных пространств и индексов. Выполняют роль кэш-памяти между базой DB2 и физическим диском. Ускоряют работу базы данных.

Активные и архивные журналы

База данных записывает любые изменения данных и другие значимые события в журнал событий. Эта информация используется для восстановления данных при сбоях, либо для осуществления запланированного отката. Каждая запись делается в активном журнале. Он имеет конечный объем. Когда он заполняется, то DB2 создает архивный журнал и копирует все значения активного журнала туда. Архивный журнал сбрасывается на диск или ленту.

Есть так называемый набор данных-связка, который следит за активными и архивными журналами, с его помощью осуществляется откат из записи а архивном журнале.

2. Практическая часть

В этой лабораторной работе Вам предстоит создать свою базу данных на мэйнфрейме и ознакомиться с изученными в теоретической части компонентами языка SQL и утилитами DB2 для z/OS.

2.1 Настройка DB2 для работы в ISPF

1. В главном меню ISPF перейдите в меню DB2I
   P->B->ADM->ADM->i
2. Выберите опцию D для настройки DB2.
3. Выберите язык, который будете использовать: IBMCOB или PL/I и нажмите ENTER.
4. Если выбран IBMCOB, то появится окно выбора настроек языка COBOL – убедитесь, что в качестве разделителя используется апостроф ‘.
5. Для Вас создана специальная база данных $DBYXX, где YXX – последние три цифры вашего логина. Во всех скриптах необходимо использовать **свою** базу данных.

2.2 Запуск скриптов в SPUFI. Создание и определение таблиц

1. Выберите в меню DB2 опцию 1 для запуска SPUFI.
2. В качестве входного набора данных введите 'TSOUsxx.CF82.SPUFIN(CF82DDE1)', где вместо TSOUsxx – вставьте ваш логин.

   Текст скрипта CF82DDE1:

```
CREATE TABLE DEPT
   ( DEPTNO   CHAR(3)      NOT NULL,
     DEPTNAME VARCHAR(36)  NOT NULL,
     MGRNO    CHAR(6)            ,
     ADMRDEPT CHAR(3)      NOT NULL,
     LOCATION VARCHAR(16)        ,
```

PRIMARY KEY (DEPTNO))
IN DATABASE $DBYXX;

CREATE UNIQUE INDEX DEPX ON DEPT(DEPTNO);

3. В качестве выходного набора данных введите 'TSOUsxx.CF82.SQLOUT'
4. Выберите YES для опций 5-9 и нажмите ENTER.
5. Исследуйте появившееся окно настроек SPUFI и нажмите ENTER.
6. Сделайте скрин-шот появившегося окна и отметьте в отчете результат выполнения пунктов 1-5.

   После выполнения пунктов 1-5 можно редактировать и запускать скрипты в SPUFI.

   • *Можно редактировать набор данных как обычно, используя редактор команд ISPF.*

   • *Можно написать комментарий, если вставить в первых двух столбцах 2 тире (--).*
7. Скрипт TSOUsxx.CF82.SPUFIN (CF82DDE1), который создает таблицу DEPT(см. рис.4).
8. Просмотрите содержимое скрипта, читая следующую информацию:

• Это DDL, который создаёт (CREATE) таблицу DEPT . Если в имени таблицы не определен владелец, ваш USERID станет владельцем.

• Перечень описаний столбцов следует за выражением CREATE. Каждый пункт в списке определяет имя столбца, его атрибуты, а также поддержку нулевой характеристики. Например, первый столбец в таблице, будет известен как DEPTNO. Это будет 3-байтовая колонка, которая будет содержать символьные данные. Этот столбец не позволяет NULL данные. Третий столбец будет называться MGRNO. Это 6-байтовая колонка, которая будет содержать символьные данные. В отличие от столбца DEPTN, MGRNO разрешает NULL данные,

поскольку нет четко указанного запрета. Другие типы данных: INTEGER, SMALLINT, DECIMAL, DATE, TIME, TIMESTAMP и VARCHAR. Для дальнейшего обсуждения атрибутов этих и других типов данных, обратитесь к SQL Справочному руководству. Другой спецификацией относительно NULL является 'NOT NULL WITH DEFAULT ". Если значение по умолчанию не указано в синтаксисе команды, оно зависит от выбранного типа столбца.

• После списка столбцов найдите определение первичного ключа (PRIMARY KEY). В этом примере, столбец DEPTNO выбран в качестве первичного ключа. Если в PRIMARY KEY входит более одного столбца, они составляют список в скобках, разделенный запятыми. Любой столбец в первичном ключе должен быть определен как NOT NULL или NOT NULL WITH DEFAULT. Как правило, NOT NULL указан, поскольку значение первичного ключа должно быть уникальным. Исключением может быть использование времени. Значение по умолчанию для времени, CURRENT TIMESTAMP, который измеряется в микросекундах.

• В целях обеспечения соблюдения уникальных значений в первичном ключе, CF82DDE1 также содержит DDL создающий уникальный индекс(UNIQUE INDEX) в столбцах первичного ключа.

Вы получили по умолчанию табличное пространство, поэтому нет необходимости определять его.

9. Нажмите F3, чтобы сохранить скрипт и вернуться в главное меню SPUFI.

   Замечание. Имейте в виду, что если опции 5 и 6 отмечены звёздочками, значит этот этап редактирования (подготовки скрипта к запуску) уже пройден.

10. Нажмите Enter, чтобы выполнить следующие шаги - запустить,автоматически, сохранить изменения базы данных в случае

успешного завершения (autocommit) и просмотреть вывод (в наборе данных, указанном в пункте 4 настроек SPUFI).

Убедитесь, что SQL выполнен правильно.

11. Следующий SQL скрипт CF82DDE2 создает таблицу EMP.

12. Просмотрите содержимое скрипта, читая следующую информацию:

• Первое выражение создает таблицу EMP. Обратите внимание на типы столбцов и определение NULL атрибутов. Свойства ссылочной целостности(Referential integrity characteristics) не являются частью определения новой таблицы.

• Выражение Создать УКАЗАТЕЛЬ (CREATE INDEX) создает уникальный индекс для того, что будет называться первичным ключом. Если UNIQUE INDEX существует, когда первичный ключ определен, менеджер баз данных будет использовать этот индекс в качестве индекса первичного ключа.

• Выражение ALTER добавляет необходимые свойства ссылочной целостности. PRIMARY KEY определяет EMPNO в качестве единственного столбца первичного ключа.

Обратите внимание на FOREIGN KEY. Он указывает имя внешнего ключа (RIEMPDEP), столбец (ы) в ключе (WORKDEPT), ссылка на родительскую таблицу (DEPT) и правило удаления DELETE RULE (SET NULL). Отметим, что это правило удаление возможно только потому, что внешний ключ столбца позволяет нулевые значения. Если внешний ключ не указан прямо, DB2 создает имя. Имя может быть от 1 до 8 символов в длину. Внешний ключ столбца(ов) должен соответствовать спецификации первичного ключа столбца(ов) в соотнесенной родительской таблице в отношении типа, размера и порядка. Не нужно никаких ссылок на столбцы, содержащиеся в исходной таблице. Так как таблица может иметь только один первичный ключ, названия родительской таблицы достаточно. Другие правила удаления включают

RESTRICT и CASCADE. Если правило не указано, предполагается RESTRICT.

Примечание: выражение ALTER было использовано в качестве иллюстрации. Таблицы с зависимостями от других таблиц можно создать с помощью CREATE с FOREIGN KEY.

• Второй индекс создан на внешний ключ в таблице EMP. Хотя этот индекс не является необходимым, он настоятельно рекомендуется, если зависимая таблица будет содержать какие-либо значительные объемы данных. В противном случае, проверка ограничения ссылочной целостности может показывать плохие результаты.

13. Запустите скрипт CF82DDE2. Убедитесь, что SQL выполнен правильно.
14. После CF82DDE2, исследуйте и запустите скрипт CF82DDE3 (создание вида VPHONE).

Этот вид является обьединением таблиц EMP и DEPT. Сейчас обе таблицы пустые, так что SELECT должен вернуть 0 строк.

2.3 Изменение данных, изучение ссылочной целостности

1. Измените и исследуйте CF82DDE4 SQL скрипт, который вставляет все данные из таблицы master EMP в вашу таблицу EMP, а затем вставляет все данные из таблицы master DEPT в вашу таблицу DEPT.

   Это имитирует выборку данных из производной таблицы в тестовую таблицу.
2. Выполните SQL скрипт CF82DDE4.
3. Отметьте в отчете результат выполнения скрипта. Объясните произошедшее.
4. Теперь измените и исследуйте CF82DDE5 SQL скрипт, который похож на CF82DDE4, но сначала вставляет строки из master таблицы в родительскую (DEPT), а затем в EMP.

5. Выполните SQL скрипт CF82DDE5.
6. Отметьте в отчете результат выполнения скрипта.
7. Теперь то же для UPDATE. Измените и исследуйте SQL скрипт CF82DDE6, который отражает необходимость переименования двух департаментов - D01 должен быть изменен на C01, D11 должен стать D31.
8. Выполните SQL скрипт CF82DDE6.
9. Отметьте в отчете результат выполнения скрипта и объясните его.
10. Собственно, там была ошибка в требованиях. D01 в действительности должен быть изменен на C10. Скрипт CF82DDE7 исправляет это. Измените и исследуйте SQL скрипт CF82DDE7.
11. Выполните SQL скрипт CF82DDE7.
12. Отметьте в отчете результат выполнения скрипта.
13. Как вы думаете, что DB2 проверяет при запуске этих скриптов?
14. Как называется отношение, которое вызвало провал второго обновления?
15. Компания хочет изменить D11 на D31, но обновление DEPT было предотвращено из-за ограничений. Таким образом, было решено сначала изменить EMP значения.

Измените и исследуйте SQL скрипт CF82DDE8.

16. Выполните SQL скрипт CF82DDE8.
17. Отметьте в отчете результат выполнения скрипта.
18. Как вы думаете, как D11 может быть изменен на D31?
19. Наконец, то же про DELETE. Компания хотела бы удалить D11 из таблицы DEPT.
20. Измените и исследуйте SQL скрипт CF82DDE9. Так же как удаление отдела он показывает, какие сотрудники находятся в этом отделе.
21. Выполните SQL скрипт CF82DDE9.
22. Отметьте в отчете результат выполнения скрипта.

23.Что произошло с тремя сотрудниками, которые работали в D11? Что-нибудь изменилось в их информации? Почему?

24.Что произошло бы, если правила удаления были CASCADE?

25.Что произошло бы, если правила удаления были RESTRICT?

26.Повлияли ли удаления из EMP на ссылочную целостность? Значат ли тут что-нибудь правила удаления?

2.4 Обновление данных

Чтобы закончить (и это может быть полезно, если данные повреждены позже) есть SQL оператор "Обновить" (refresh), который удаляет данные из EMP, удаляет данные из DEPT и вставляет новый набор данных из master таблиц.

1. Измените и исследуйте SQL скрипт CF82DDER.
2. Выполните SQL скрипт CF82DDER.

2.5 Самостоятельная работа

Разработать структуру базовых таблиц (не менее трех) базы данных (смотри варианты заданий к работе), удовлетворяющих требованиям целостности, непротиворечивости и неизбыточности.

Создать структуры базовых таблиц, и наполнить их содержимым состоящим более чем из 10 записей. При создании структуры таблиц целесообразно задавать ключевые (уникальные) поля. Это поможет в дальнейшем для организации связей между таблицами.

Примерные варианты предметной области базы данных

1 Студенческая библиотека

2 Прокат видеокассет

3 Риэлторская фирма

4 ГИБДД (Государственная инспекция безопасности дорожного движения) г. Х

5 Студенты ВУЗа

6 Кадры предприятия

7 Компьютерный фонд ВУЗа

8 Временные трудовые коллективы

9 ГТС - городская телефонная сеть

10 Автокаталог

11 Аудиторный фонд ВУЗа

12 Авиапассажирские перевозки г. Х

13 Склад предприятия

14 Кадры (преподаватели)

15 Экзаменационная сессия

16 Турагентство г. Х

17 Услуги от А до Я

18 Музыкальные торговые объекты региона Х

19 Авиалинии “Голубое небо” (пассажирские и транспортные перевозки)

20 Сборка и реализация компьютеров

21 Продуктовые магазины района Х

22 Оптовая база

23 Спортивные комплексы региона Х

24 Районный расчет квартплаты

25 Фирмы покупки и сбыта автомобилей

26 Налоговая инспекция. Налоги с физических и юридических лиц

Возможна другая тематика для создания баз данных по согласованию с преподавателем.

3. Требования к отчету

Отчет о выполнении лабораторной работы должен содержать ответы на вопросы, поставленные перед студентом в методических указаниях,

скрин-шоты результатов выполнения практических заданий, структуру разработанной базы данных, SQL-код для её создания и ответы на контрольные вопросы. Оформление отчета должно быть выполнено в соответствии с требованиями ГОСТов.

4. Контрольные вопросы

Какие существуют компоненты языка SQL?

Как запустить SQL скрипт?

Чем отличается вид от таблицы?

Как создать таблицу? Как создать вид?

Перечислите свойства ссылочной целостности?

Какие виды правил удаления существуют?

Какие есть способы восстановления утерянных данных в DB

5. Список литературы

5.1 Paolo Bruni, Mark Anders, «DB2 for z/OS: Data Sharing in a Nutshell», International Business Machines Corporation, 2006.

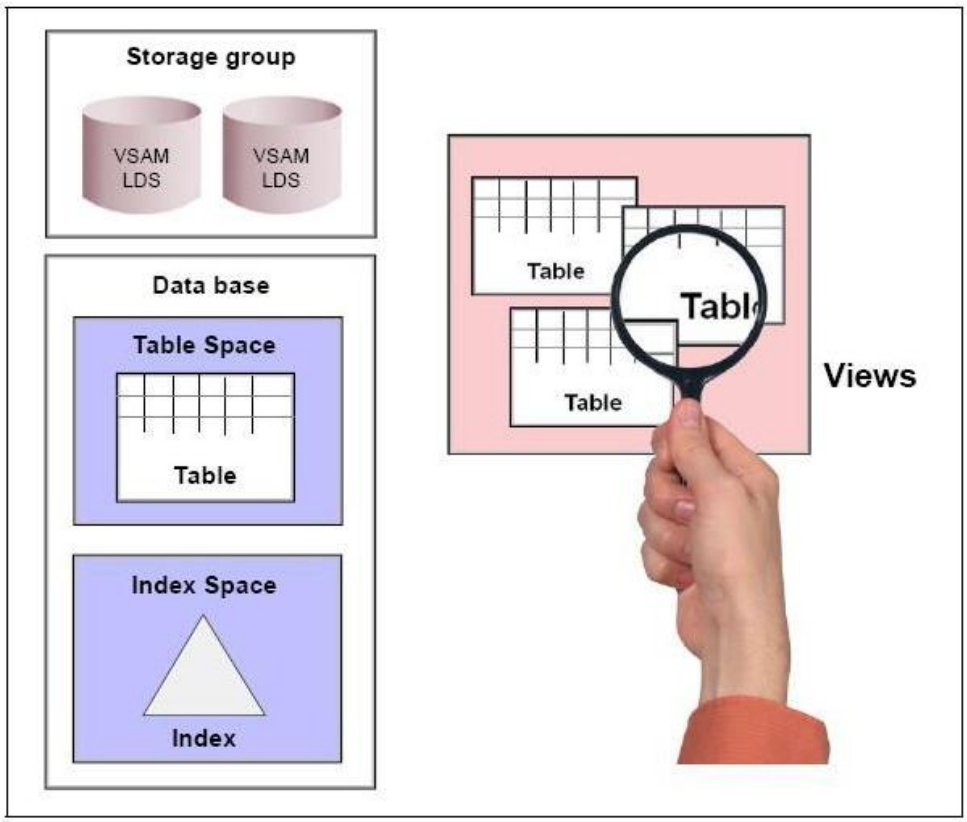
5.2 Справочник «DB2 SQL Reference», International Business Machines Corporation, 2003.

5.3 Mike Ebbers, Wayne O'Brien, Bill Ogden, «Introduction to the New Mainframe: z/OS Basics», IBM corp., 2005.

Содержание:

```
                              SPUFI                                  SSID:
===>

Enter the input data set name:         (Can be sequential or partitioned)
 1  DATA SET NAME ... ===> 'TSOUsxx.CF82.SPUFIN(CF82DDE1)'
                      +
 2  VOLUME SERIAL ... ===>             (Enter if not cataloged)
 3  DATA SET PASSWORD ===>             (Enter if password protected)

Enter the output data set name:        (Must be a sequential data set)
 4  DATA SET NAME ... ===> 'TSOUsxx.CF82.SQLOUT'

Specify processing options:
 5  CHANGE DEFAULTS    ===> YES        (Y/N - Display SPUFI defaults panel?)
 6  EDIT INPUT ......  ===> YES        (Y/N - Enter SQL statements?)
 7  EXECUTE .........  ===> YES        (Y/N - Execute SQL statements?)
 8  AUTOCOMMIT ......  ===> YES        (Y/N - Commit after successful run?)
 9  BROWSE OUTPUT ...  ===> YES        (Y/N - Browse output data set?)

For remote SQL processing:
10  CONNECT LOCATION   ===>


PRESS:  ENTER to process    END to exit              HELP for more information
```

Puc. 1



Puc.2

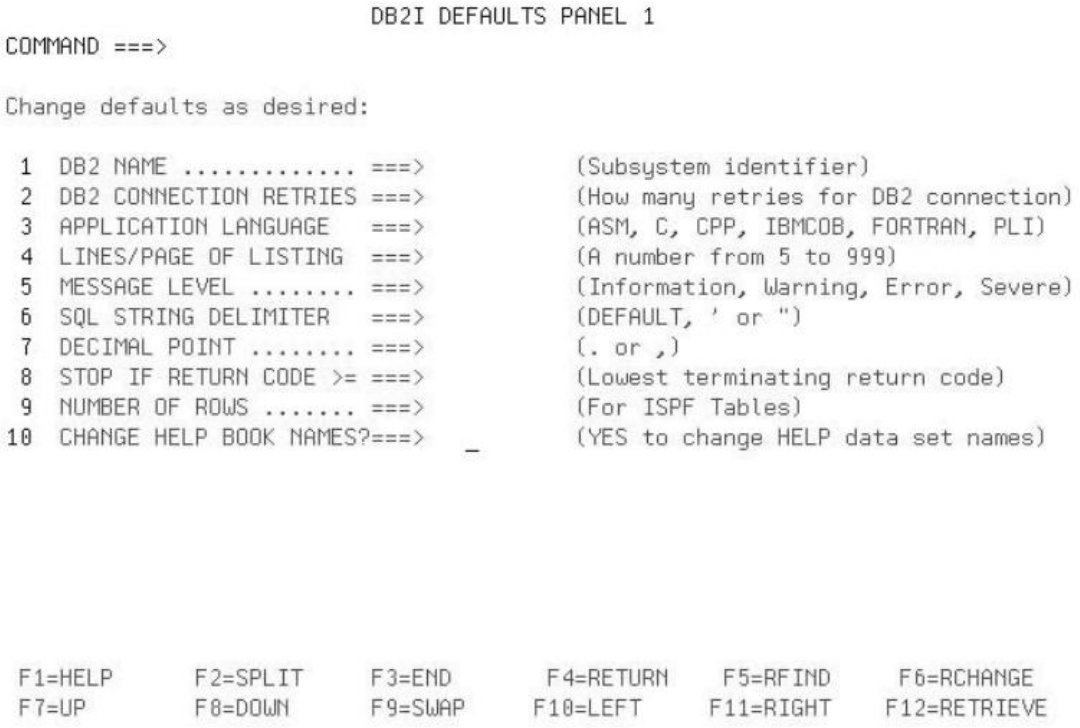


Рис. 3

ТАБЛИЦЫ

	СТОЛБЕЦ	ТИП	Нулевой	ОПИСАНИЕ
EMP	EMPNO	CHAR(6)	NO	Номер работника PK
	FIRSTNME	VARCHAR(12)	NO	Имя работника
	MIDINIT	CHAR(1)	NO	Отчество работника
	LASTNAME	VARCHAR(15)	NO	Фамилия работника
	WORKDEPT	CHAR(3)	YES	№ отдела работника FK
	PHONENO	CHAR(4)	YES	телефон работника
	HIREDATE	DATE	YES	дата найма
	JOB	CHAR(8)	YES	обязанности работника
	EDUCLVL	SMALLINT	YES	время обучения
	SEX	CHAR(1)	YES	пол: M или F
	BIRTHDATE	DATE	YES	дата рождения
	SALARY	DECIMAL(9,2)	YES	годовая зарплата
	BONUS	DECIMAL(9,2)	YES	бонус
	COMM	DECIMAL(9,2)	YES	комиссия
DEPT	DEPTNO	CHAR(3)	NO	номер отдела PK
	DEPTNAME	VARCHAR(36)	NO	название отдела
	MGRNO	CHAR(6)	YES	EMPNO руководителя отдела
	ADMRDEPT	CHAR(3)	NO	номер главного отдела

Рис. 4

Подрисуночные подписи

Рис. 1. Использование редактора подсистемы SPUFI

Рис. 2. Иерархия объектов в подсистеме DB2

Рис. 3.Окно выбора настроек меню DB2I

Рис. 4. Структура таблиц EMP и DEPT