Лабораторная работа 2

Генерация хранилища данных и заполнение его записями

Задание на лабораторную работу

В рамках данной лабораторной работы необходимо произвести генерацию структуры хранилища данных и заполнить его записями. В качестве исходных данных выступает база «North Wind».

Выполнение

3.1 Процесс создания хранилища данных начинается с проектирования его архитектуры. Существуют три типа технологии OLAP:[1]

• многомерная OLAP (Multidimensional OLAP — MOLAP);

• реляционная OLAP (Relational OLAP — ROLAP);

• гибридная OLAP (Hybrid OLAP — HOLAP).

MOLAP — это классическая форма OLAP, так что её часто называют просто OLAP. Она использует суммирующую БД, специальный вариант процессора пространственных БД и создаёт требуемую пространственную схему данных с сохранением как базовых данных, так и агрегатов.

ROLAP работает напрямую с реляционным хранилищем, факты и таблицы с измерениями хранятся в реляционных таблицах, и для хранения агрегатов создаются дополнительные реляционные таблицы.

HOLAP использует реляционные таблицы для хранения базовых данных и многомерные таблицы для агрегатов.

Для создаваемого хранилища используется тип ROLAP: исходные данные и данные хранилища расположены в реляционной БД.

3.2 Основой для дизайна хранилища данных является денормализованная модель (хранилище с измерениями), составляющими

которой являются таблица фактов (fact table) и таблицы измерений (dimension tables).

Таблица фактов является основной таблицей хранилища данных и , как правило, содержит уникальный составной ключ, объединяющий первичные ключи таблиц измерений. Как правило, она содержит сведения об объектах или событиях, совокупность которых будет в дальнейшем анализироваться. Обычно говорят о четырех наиболее часто встречающихся типах фактов. К ним относятся:

- факты, связанные с транзакциями (Transaction facts). Они основаны на отдельных событиях (типичными примерами которых являются телефонный звонок или снятие денег со счета с помощью банкомата);

- факты, связанные с «моментальными снимками» (Snapshot facts). Основаны на состоянии объекта (например, банковского счета) в определенные моменты времени, например на конец дня или месяца. Типичными примерами таких фактов являются объем продаж за день или дневная выручка;

- факты, связанные с элементами документа (Line-item facts). Основаны на том или ином документе (например, счете за товар или услуги) и содержат подробную информацию об элементах этого документа (например, количестве, цене, проценте скидки);

- факты, связанные с событиями или состоянием объекта (Event or state facts). Представляют возникновение события без подробностей о нем (например, просто факт продажи или факт отсутствия таковой без иных подробностей).

Таблицы измерений содержат неизменяемые либо редко изменяемые данные. В подавляющем большинстве случаев эти данные представляют собой по одной записи для каждого члена нижнего уровня иерархии в измерении. Таблицы измерений также содержат как минимум одно описательное поле (обычно с именем члена измерения) и, как правило, целочисленное ключевое поле (обычно это суррогатный ключ) для однозначной идентификации члена измерения.

Одно измерение куба может содержаться как в одной таблице (в том числе и при наличии нескольких уровней иерархии), так и в нескольких связанных таблицах, соответствующих различным уровням иерархии в измерении. Если каждое измерение содержится в одной таблице, такая схема

хранилища данных носит название «звезда» (star schema). Если же хотя бы одно измерение содержится в нескольких связанных таблицах, такая схема хранилища данных носит название «снежинка» (snowflake schema).

С целью повышения скорости выполнения запросов к хранилищу данных в рамках данной лабораторной работы предпочтение отдается схеме «звезда».

3.3 Запустите MySQL Workbench и выберите на стартовой странице пункт «Create New EER Model». Для более поздних версий программы, нажмите на плюсик в нижней секции стартовой страницы.

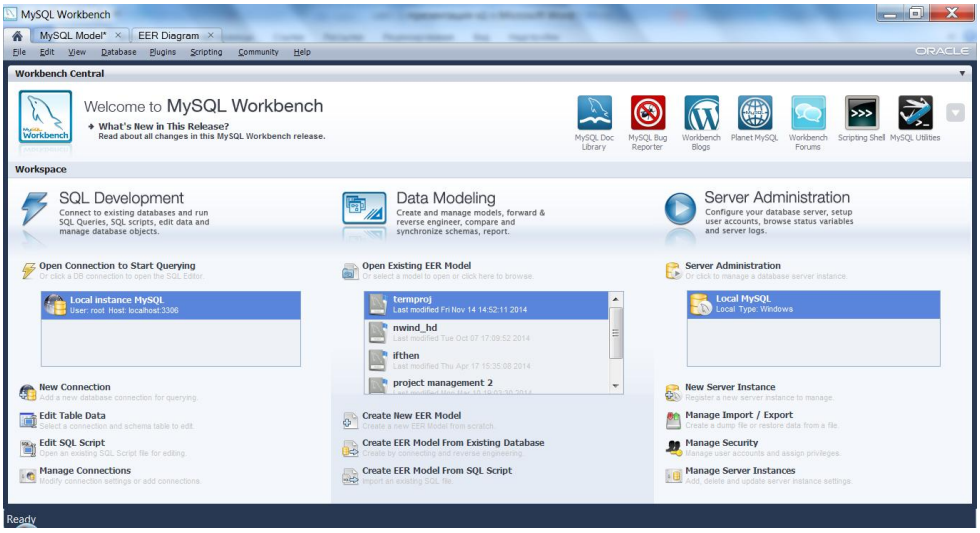

Рис.3.3.1 Начало работы с MySQL Workbench.

Перед Вами откроется новая вкладка. Сохраните модель, выбрав пункт «File»-«Save model» главного меню под именем nwind_hd. Измените также название схемы на nwind_hd, щёлкнув 2 раза по изображению цилиндра и введя новое название в появившемся поле.

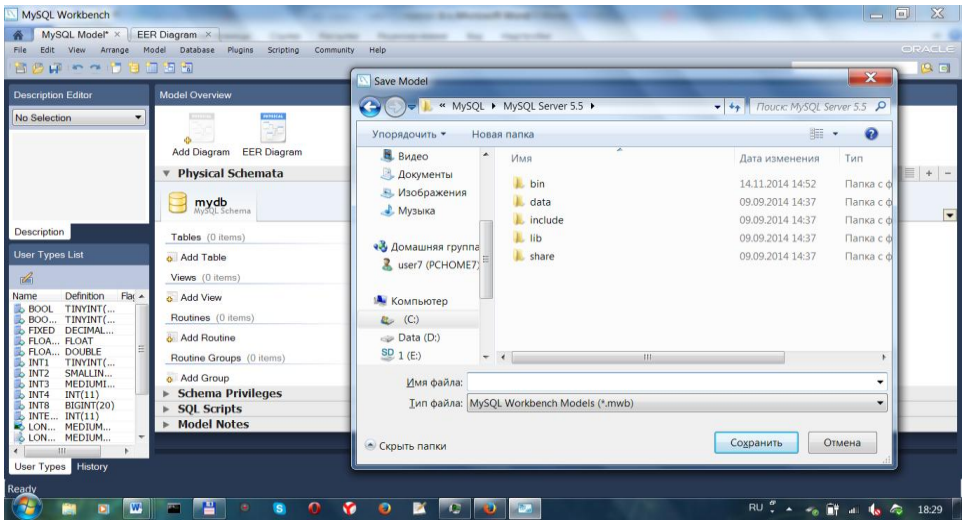

Рис.3.3.2 Окно сохранения модели.

Во вкладке выберите «Add Diagram» («Создать новую диаграмму»). Вы автоматически будете переведены в режим редактирования только что созданной диаграммы.

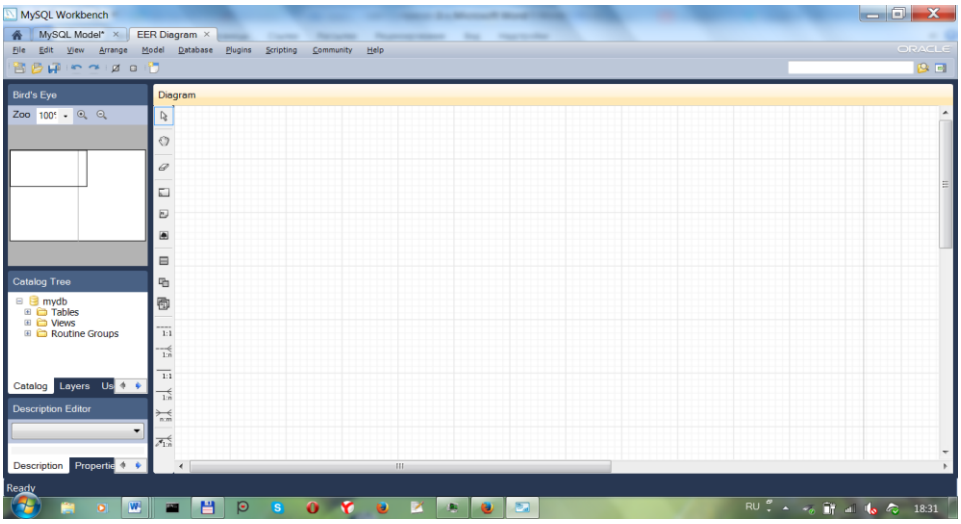

Рис.3.3.3. Добавление новой диаграммы.

С помощью инструментов добавления таблиц («Add Table» с вкладки «MySQL Model») и установления связей между ними (иконки в левой части рабочей области вкладки «Диаграмма») постройте основу хранилища – схему базы данных, как показано на рисунке 3.3.4:

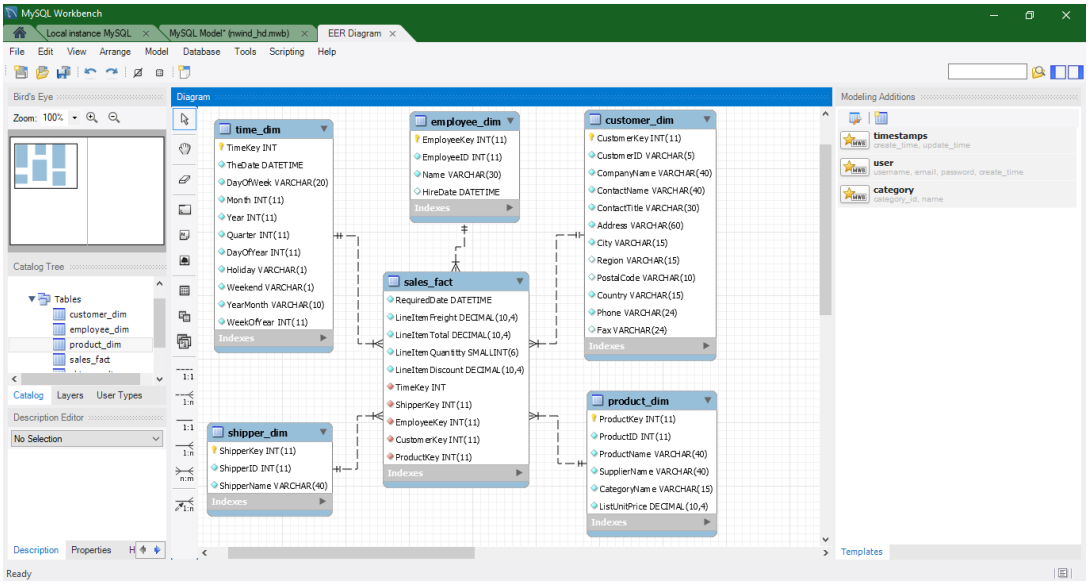

Рис.3.3.4. Схема хранилища данных.

Для этого нужно создать соответствующие колонки и правильно расставить флажки для каждой из них в окошке, появляющемся после выбора таблицы.

На диаграмме ииспользуются следующие обозначения:

– флажок NN (not null) установлен

– флажок NN снят

 – флажок PK (primary key) установлен. **Необходимо также установить флажок AI (auto increment) для каждого ключа, чтобы избежать ошибок на следующих этапах выполнения работы.**

– данные колонки создавать не надо, они появятся сами после установления связей. Связи устанавливаются при помощи кнопки 1:n следующим образом: сначала выбирается таблица фактов, а затем ключ из определённой таблицы измерений.

Рассмотрим подробнее структуру таблиц приведенной выше схемы.

Таблицы измерений:

- Перевозчик: ключ – id Перевозчика – имя Перевозчика;
- Продукт: ключ – id продукта – наименование его – имя поставщика – наименование категории – цена единицы продукта;
- Покупатель: ключ – id – имя компании – контактное лицо – должность – адрес – город – регион – почтовый индекс – страна – телефон – факс;
- Работник: ключ – id – имя работника – дата приема на работу;
- Время: ключ – несколько полей, содержащих значения даты доставки заказа в различном формате (подробное описание данного измерения приведено далее).

Таблица фактов:

- 5 ключей таблиц измерений;
- `northwind_hd`.`orders`.`requireddate` - предполагаемая дата заказа (из исходной таблицы Заказы);
- Плата за перевозку грузов морем – рассчитывается как `northwind_hd`.`orders`.`Freight` * `northwind_hd`.`order details`.`Quantity` / (SELECT SUM(Quantity) FROM `northwind_hd`.`order details` as od WHERE od.OrderID = `northwind_hd`.`orders`.`OrderID`);
- Суммарная стоимость заказа – рассчитывается как произведение цены единицы товара на его количество из order_details;
- Количество перевозимого товара – order_details quantity;

- Скидка – рассчитывается как произведение размера скидки на цену единицы товара на его количество из order_details.

Особое внимание здесь следует уделить измерению «Время». Члены измерений могут быть объединены одной или несколькими иерархиями. При построении OLAP-кубов по оси с датой заказа мы можем хотеть группировать точки (т.е. дни доставки заказов) по иерархии, например, *Год-Месяц-День*.

В рамках данной работы для измерения «Время» используется следующая иерархия (см.рисунок):

- Дата в длинном формате – ShippedDate (дата доставки заказа) из Orders (Заказы) исходной БД;
- Год – квартал – месяц - номер месяца - номер недели - день недели - день года – праздник - выходной – их значения рассчитываются на основе даты в длинном формате с помощью стандартных функций mysql date_format(date,specification) (типа printf()), quarter(date) и week(date) [2].

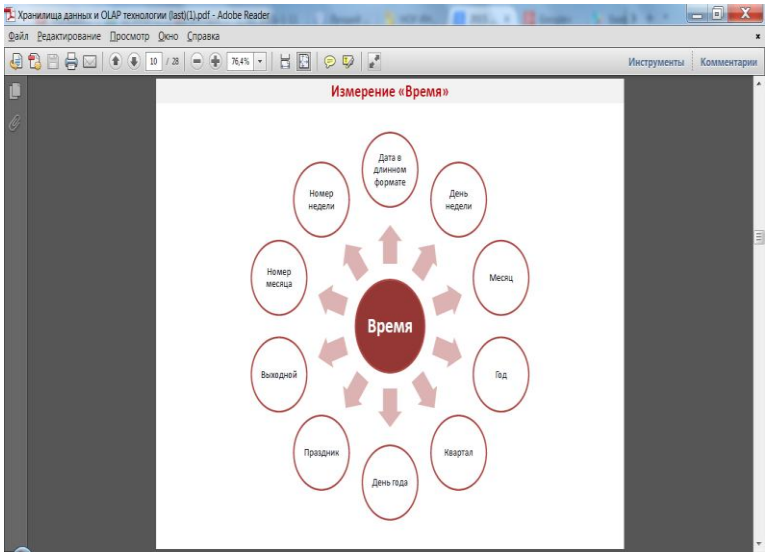

Рис.3.3.5. Иерархия значений измерения «Время».

3.4 Для создания БД хранилища необходимо сгенерировать SQL-скрипт и выполнить его. Для этого в главном меню выберите пункт «Database»-«Forward Engineering». Перед Вами откроется диалоговое окно (см.рисунки) генерации и выполнения скрипта. Сначала Вам будет предложено установить параметры генерации таблиц, затем – выбрать объекты схемы, которые будут созданы в базе данных (выберите все шесть таблиц).

Далее будет сгенерирован текст SQL-скрипта, текст которого приведен в Листинге 1. На следующем этапе выберите уже существующий экземпляр подключения к серверу MySQL или создайте новый. Чтобы выполнить запрос, формирующий структуру хранилища, нажмите «Execute». В результате Вы увидите отчет о состоянии произведенного к серверу запроса.

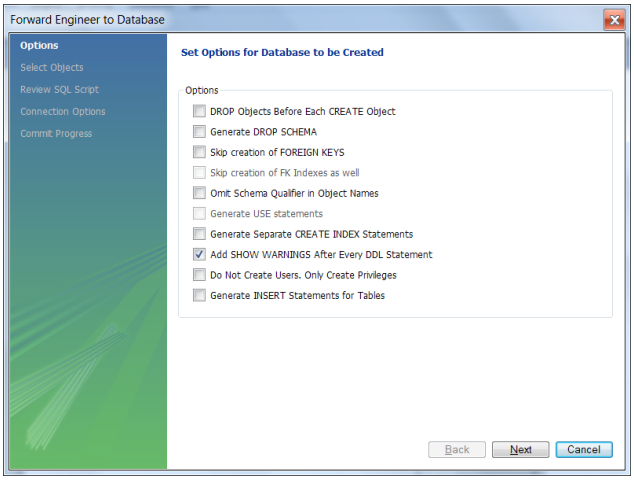

Рис.3.4.1. Настройка параметров выполнения скрипта.

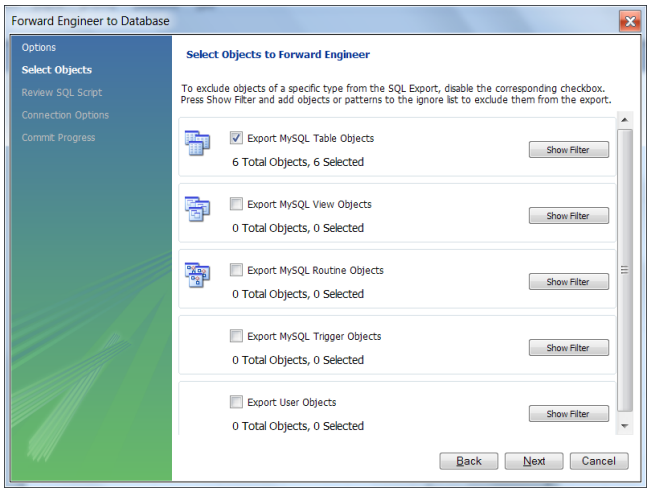

Рис.3.4.2. Выбор экспортируемых в базу объектов.

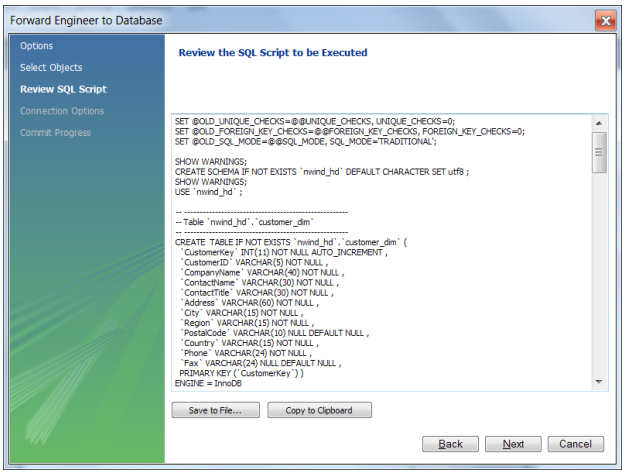

Рис.3.4.3. Генерация текста скрипта.

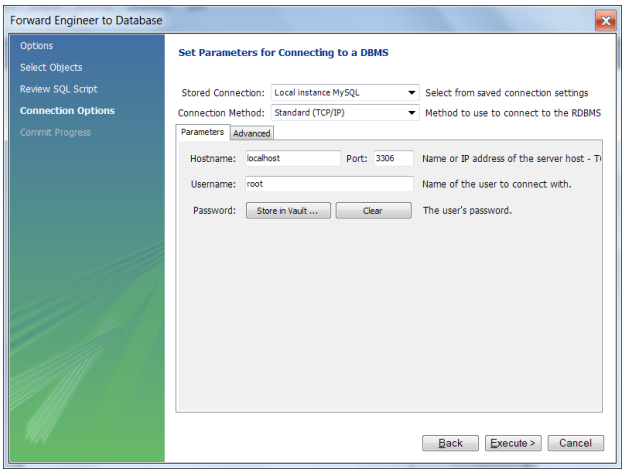

Рис.3.4.4. Установление подключения к серверу баз данных.

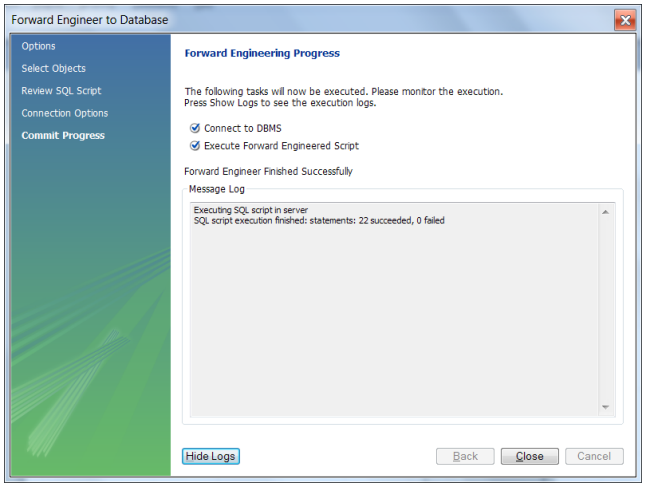

Рис.3.4.5. Результат выполнения скрипта.

Листинг 1. SQL-скрипт для генерации структуры хранилища данных

```
-- MySQL Workbench Forward Engineering


SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET                  @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;

SET                                        @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';


-- -----------------------------------------------------

-- Schema nwind_hd

-- -----------------------------------------------------


-- -----------------------------------------------------

-- Schema nwind_hd

-- -----------------------------------------------------

CREATE  SCHEMA  IF  NOT  EXISTS  `nwind_hd`  DEFAULT  CHARACTER  SET
utf8 ;

SHOW WARNINGS;

USE `nwind_hd` ;


-- -----------------------------------------------------

-- Table `nwind_hd`.`time_dim`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `nwind_hd`.`time_dim` (

  `TimeKey` INT NOT NULL AUTO_INCREMENT,

  `TheDate` DATETIME NOT NULL,

  `DayOfWeek` VARCHAR(20) NOT NULL,

  `Month` INT(11) NOT NULL,
```

```
  `Year` INT(11) NOT NULL,
  `Quarter` INT(11) NOT NULL,
  `DayOfYear` INT(11) NOT NULL,
  `Holiday` VARCHAR(1) NOT NULL,
  `Weekend` VARCHAR(1) NOT NULL,
  `YearMonth` VARCHAR(10) NOT NULL,
  `WeekOfYear` INT(11) NOT NULL,
  PRIMARY KEY (`TimeKey`))
ENGINE = InnoDB;

SHOW WARNINGS;

-- -----------------------------------------------------
-- Table `nwind_hd`.`shipper_dim`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `nwind_hd`.`shipper_dim` (
  `ShipperKey` INT(11) NOT NULL AUTO_INCREMENT,
  `ShipperID` INT(11) NOT NULL,
  `ShipperName` VARCHAR(40) NOT NULL,
  PRIMARY KEY (`ShipperKey`))
ENGINE = InnoDB;

SHOW WARNINGS;

-- -----------------------------------------------------
-- Table `nwind_hd`.`employee_dim`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `nwind_hd`.`employee_dim` (
```

```
  `EmployeeKey` INT(11) NOT NULL AUTO_INCREMENT,
  `EmployeeID` INT(11) NOT NULL,
  `Name` VARCHAR(30) NOT NULL,
  `HireDate` DATETIME NULL,
  PRIMARY KEY (`EmployeeKey`))
ENGINE = InnoDB;

SHOW WARNINGS;

-- -----------------------------------------------------
-- Table `nwind_hd`.`customer_dim`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `nwind_hd`.`customer_dim` (
  `CustomerKey` INT(11) NOT NULL AUTO_INCREMENT,
  `CustomerID` VARCHAR(5) NOT NULL,
  `CompanyName` VARCHAR(40) NOT NULL,
  `ContactName` VARCHAR(40) NOT NULL,
  `ContactTitle` VARCHAR(30) NOT NULL,
  `Address` VARCHAR(60) NOT NULL,
  `City` VARCHAR(15) NOT NULL,
  `Region` VARCHAR(15) NULL,
  `PostalCode` VARCHAR(10) NULL,
  `Country` VARCHAR(15) NOT NULL,
  `Phone` VARCHAR(24) NOT NULL,
  `Fax` VARCHAR(24) NULL,
  PRIMARY KEY (`CustomerKey`))
ENGINE = InnoDB;
```

```
SHOW WARNINGS;

-- -----------------------------------------------------
-- Table `nwind_hd`.`product_dim`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `nwind_hd`.`product_dim` (
  `ProductKey` INT(11) NOT NULL AUTO_INCREMENT,
  `ProductID` INT(11) NOT NULL,
  `ProductName` VARCHAR(40) NOT NULL,
  `SupplierName` VARCHAR(40) NOT NULL,
  `CategoryName` VARCHAR(15) NOT NULL,
  `ListUnitPrice` DECIMAL(10,4) NOT NULL,
  PRIMARY KEY (`ProductKey`))
ENGINE = InnoDB;

SHOW WARNINGS;

-- -----------------------------------------------------
-- Table `nwind_hd`.`sales_fact`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `nwind_hd`.`sales_fact` (
  `RequiredDate` DATETIME NOT NULL,
  `LineItemFreight` DECIMAL(10,4) NOT NULL,
  `LineItemTotal` DECIMAL(10,4) NOT NULL,
  `LineItemQuantitty` SMALLINT(6) NOT NULL,
  `LineItemDiscount` DECIMAL(10,4) NOT NULL,
  `TimeKey` INT NOT NULL,
  `ShipperKey` INT(11) NOT NULL,
```

```
`EmployeeKey` INT(11) NOT NULL,
`CustomerKey` INT(11) NOT NULL,
`ProductKey` INT(11) NOT NULL,
INDEX `fk_sales_fact_time_dim_idx` (`TimeKey` ASC),
INDEX `fk_sales_fact_shipper_dim1_idx` (`ShipperKey` ASC),
INDEX `fk_sales_fact_employee_dim1_idx` (`EmployeeKey` ASC),
INDEX `fk_sales_fact_customer_dim1_idx` (`CustomerKey` ASC),
CONSTRAINT `fk_sales_fact_time_dim`
  FOREIGN KEY (`TimeKey`)
  REFERENCES `nwind_hd`.`time_dim` (`TimeKey`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
CONSTRAINT `fk_sales_fact_shipper_dim1`
  FOREIGN KEY (`ShipperKey`)
  REFERENCES `nwind_hd`.`shipper_dim` (`ShipperKey`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
CONSTRAINT `fk_sales_fact_employee_dim1`
  FOREIGN KEY (`EmployeeKey`)
  REFERENCES `nwind_hd`.`employee_dim` (`EmployeeKey`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
CONSTRAINT `fk_sales_fact_customer_dim1`
  FOREIGN KEY (`CustomerKey`)
  REFERENCES `nwind_hd`.`customer_dim` (`CustomerKey`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
CONSTRAINT `fk_sales_fact_product_dim1`
```

```
    FOREIGN KEY (`ProductKey`)
    REFERENCES `nwind_hd`.`product_dim` (`ProductKey`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SHOW WARNINGS;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

3.5 Следующим шагом является заполнение хранилища данными. Принимая во внимание его архитектуру (в соответствии со схемой «звезда» таблицы измерений связаны с таблицей фактов связями типа «один-ко-многим»), следует сначала заполнить записями таблицы измерений, а затем – таблицу фактов. Для выполнения данной задачи можно использовать SQL-скрипт, текст которого приведен в Листинге 2. Для этого нужно перейти на вкладку «Local instance MySQL» и вставить в белое поле вкладки «Query 1» текст листинга.

Листинг 2. SQL-скрипт заполнения хранилища данных записями

```
use nwind_hd;
-- customer_dim --
insert into `nwind_hd`.`customer_dim`
(CustomerID,CompanyName,ContactName,ContactTitle,Address,City,Re
gion,PostalCode,Country,Phone,Fax)
select * from`northwind_hd`.`customers`;
-- shipper_dim --
insert into `nwind_hd`.`shipper_dim` (shipperid,shippername)
select shipperid,companyname from `northwind_hd`.`shippers`;
-- employee_dim; --
insert into `nwind_hd`.`employee_dim` (employeeid,name,hiredate)
select employeeid,concat_ws(" ",firstname,lastname) as
name,hiredate from`northwind_hd`.`employees`;
-- product_dim;--
insert into `nwind_hd`.`product_dim`
(productid,productname,suppliername,categoryname,listunitprice)
select p.productid,p.productname,s.companyname,
c.categoryname,p.unitprice
```

```
from `northwind_hd`.`products` as p,`northwind_hd`.`categories`
as c, `northwind_hd`.`suppliers` as s
where p.categoryid=c.categoryid and p.supplierid=s.supplierid
order by p.productid;
-- time_dim --
insert into `nwind_hd`.`time_dim` (
`TheDate`,`DayOfWeek` ,`Month`,`Year`,`Quarter`,`DayOfYear`
,`Holiday`,
    `Weekend` ,`YearMonth`,`WeekOfYear`)
SELECT DISTINCT
 S.ShippedDate AS TheDate,
 DATE_FORMAT(S.ShippedDate,'%W') as DW,
 DATE_FORMAT(S.ShippedDate,'%m') as M,
 DATE_FORMAT(S.ShippedDate,'%Y') as Y,
 QUARTER(S.ShippedDate) as Q,
 DATE_FORMAT(S.ShippedDate,'%j') as DY,
 'N' AS Holiday,
 CASE
 WHEN DATE_FORMAT(S.ShippedDate,'%w') = 0 THEN 'Y'
 WHEN DATE_FORMAT(S.ShippedDate,'%w') = 6 THEN 'Y'
 ELSE 'N'
 END as WEND,
 DATE_FORMAT(S.ShippedDate,'%M') as MY,
 WEEK(S.ShippedDate,3) as WY -- monday==1 --
FROM `northwind_hd`.`Orders` as  S
WHERE S.ShippedDate IS NOT NULL;
-- facts --
insert into `nwind_hd`.`sales_fact`
select timekey,customerkey,shipperkey,productkey,employeekey,
`northwind_hd`.`orders`.`requireddate`,
`northwind_hd`.`orders`.`Freight` * `northwind_hd`.`order
details`.`Quantity` /
(SELECT SUM(Quantity)
FROM `northwind_hd`.`order details` as od
WHERE od.OrderID = `northwind_hd`.`orders`.`OrderID`) AS
LineItemFreight,
`northwind_hd`.`order details`.`UnitPrice` *
`northwind_hd`.`order details`.`Quantity` AS LineItemTotal,
`northwind_hd`.`order details`.`quantity` as LineItemQuantity,
`northwind_hd`.`order details`.`discount`*`northwind_hd`.`order
details`.`unitprice`*
`northwind_hd`.`order details`.`quantity` as LineItemDiscount
from `northwind_hd`.`orders`
inner join `northwind_hd`.`order details`
on `northwind_hd`.`orders`.`orderid`=`northwind_hd`.`order
details`.`orderid`
inner join `nwind_hd`.`product_dim`
on `northwind_hd`.`order
details`.`productid`=`nwind_hd`.`product_dim`.`productid`
inner join `nwind_hd`.`customer_dim`
on
`northwind_hd`.`orders`.`customerid`=`nwind_hd`.`customer_dim`.`
customerid`
```

```
inner join `nwind_hd`.`employee_dim`
on
`northwind_hd`.`orders`.`employeeid`=`nwind_hd`.`employee_dim`.`
employeeid`
inner join `nwind_hd`.`shipper_dim`
on
`northwind_hd`.`orders`.`shipvia`=`nwind_hd`.`shipper_dim`.`ship
perid`
inner join `nwind_hd`.`time_dim`
on
`northwind_hd`.`orders`.`shippeddate`=`nwind_hd`.`time_dim`.`the
date`
where (`northwind_hd`.`orders`.`ShippedDate` IS NOT NULL);
```

Заключение

В ходе выполнения данной работы было сформировано хранилище данных для базы данных «North Wind». Перенос данных из исходной базы в хранилище был осуществлен с помощью SQL-скрипта с использованием стандартных функций MySQL преобразования данных. В отчете было приведено подробное описание всех выполненных в рамках работы процедур, тексты скриптов, изображения, иллюстрирующие шаги выполнения работы.

Список информационных источников

1. http://citforum.ru/consulting/BI/xolap_classification/
2. http://www.mysql.ru/docs/man/

Оглавление