**4. DESIGN**

**4.1 DATA FLOW DIAGRAM [ DFD ]**

The DFD was first developed by Larry Constiane as a way of expressing system in a graphical form. A DFD, also known as Bubble Chart, has a purpose of clarifying system requirement and identifying major transformation that will become the programs in the system design.

**DFD SYMBOLS**

* A **RECTANGLE** defines a source or destination of system data
* An **ARROW** identifies data flow or data in motion. It is a pipeline through which information flow.
* A **CIRCLE** or a **BUBBLE** (Some people use an over bubble) represents a process transforms in coming data flow into outgoing data flow.
* An **OPEN RECTANGLE** is a data store or data at rest or a temporary rest repository of data.

Note that a DFD describe what data flow (logical) rather than they are processed, so it does not depend on hardware, software and data structure or file organization.

**PROCESSING OF E-DISTRIBUTED CHANNEL**

**DFD for Processing of Project:-**

Error Msg

Manufacture

Error Msg

Dealer

Error Msg

Retailer

Error Msg

Customer

Admin

Error Msg

**DFD FOR BILL RECEIPT MODULE**

Bill Receipt

User

**DFD FOR PRODUCT MODULE PROCESSING**

Order details

User

**DFD FOR APPROVING TRANSACTION OF E-DISTRIBUTED SYSTEM BY ADMINISTRATOR**

Login

administrator

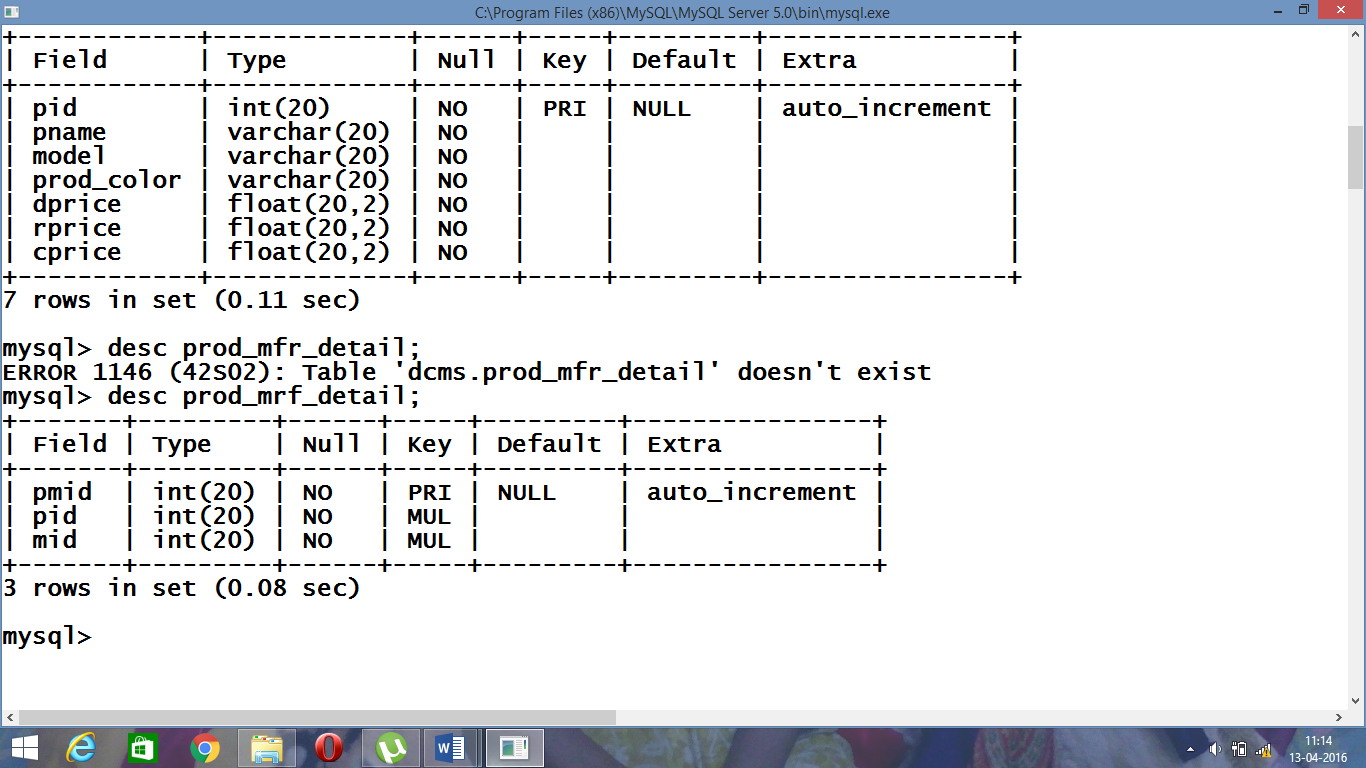
Registration Table

Orders

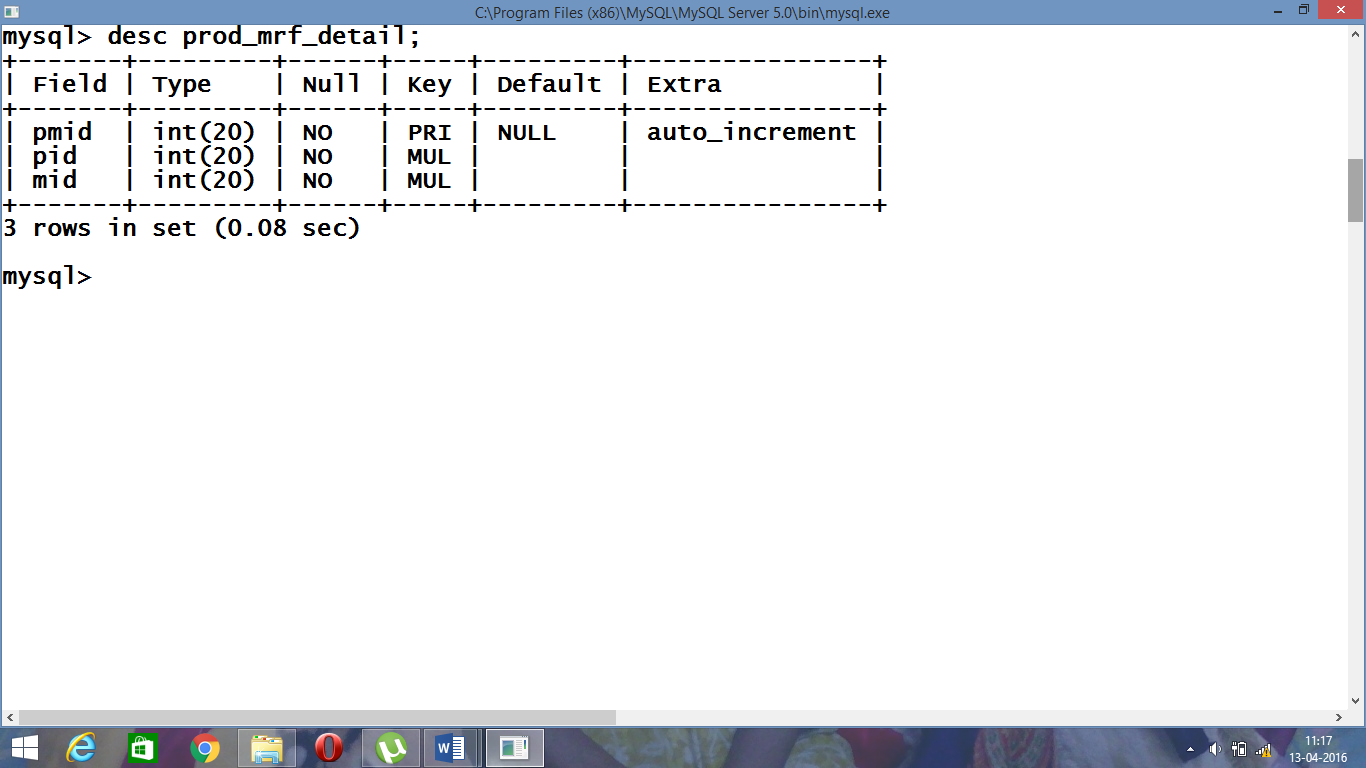
transacts Ads

**4.2Backend Design**

# TABLE: PRODUCT



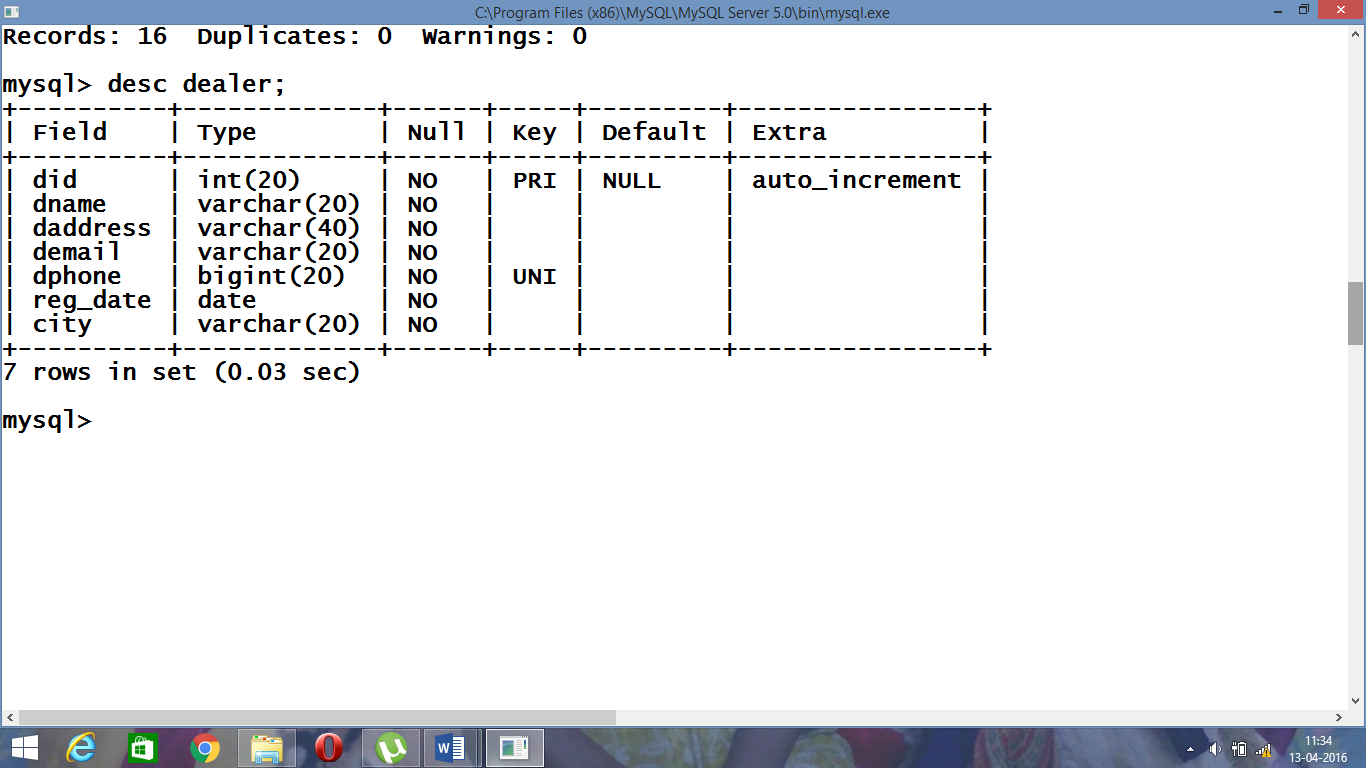
# TABLE: PROD\_MFR\_DETAIL



# TABLE: MANUFACTURE

# 

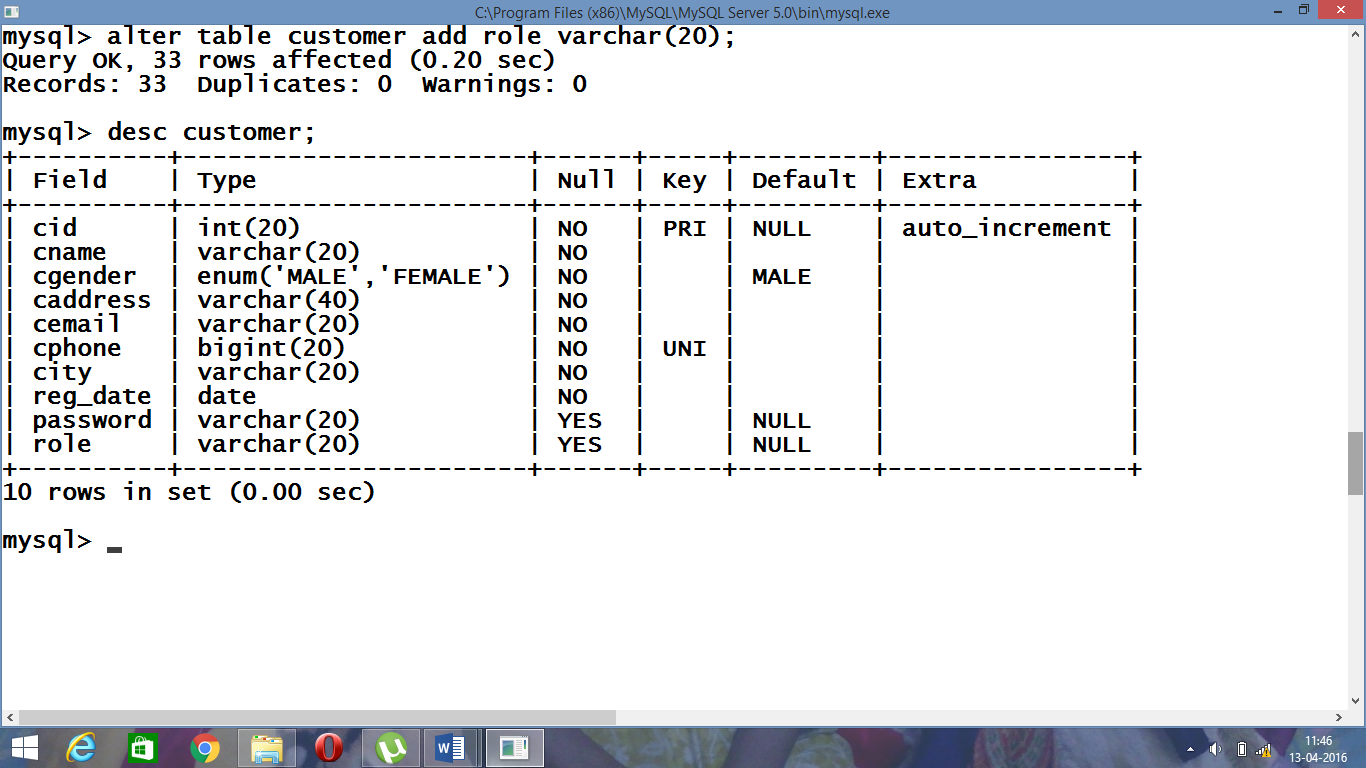
# TABLE: DEALER



# TABLE: RETAILER

# 

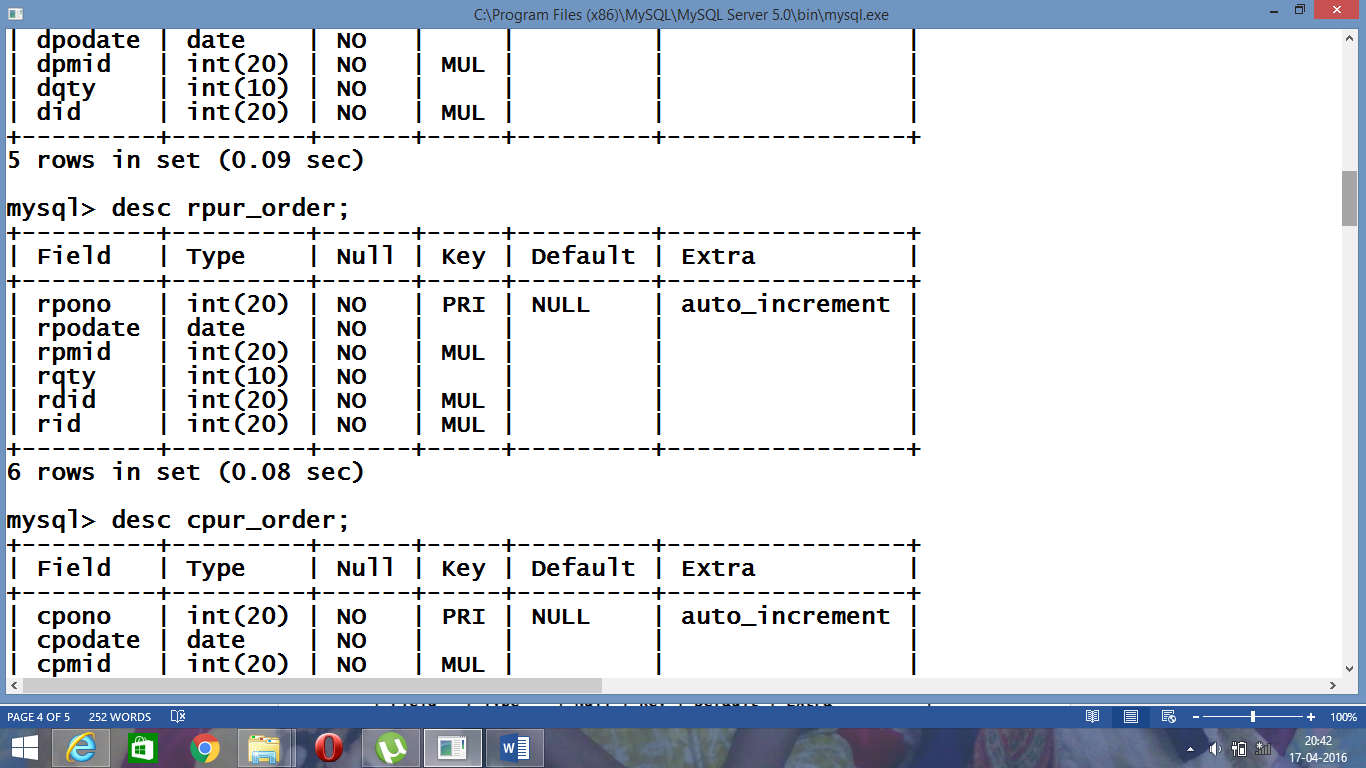
# TABLE: CUSTOMER



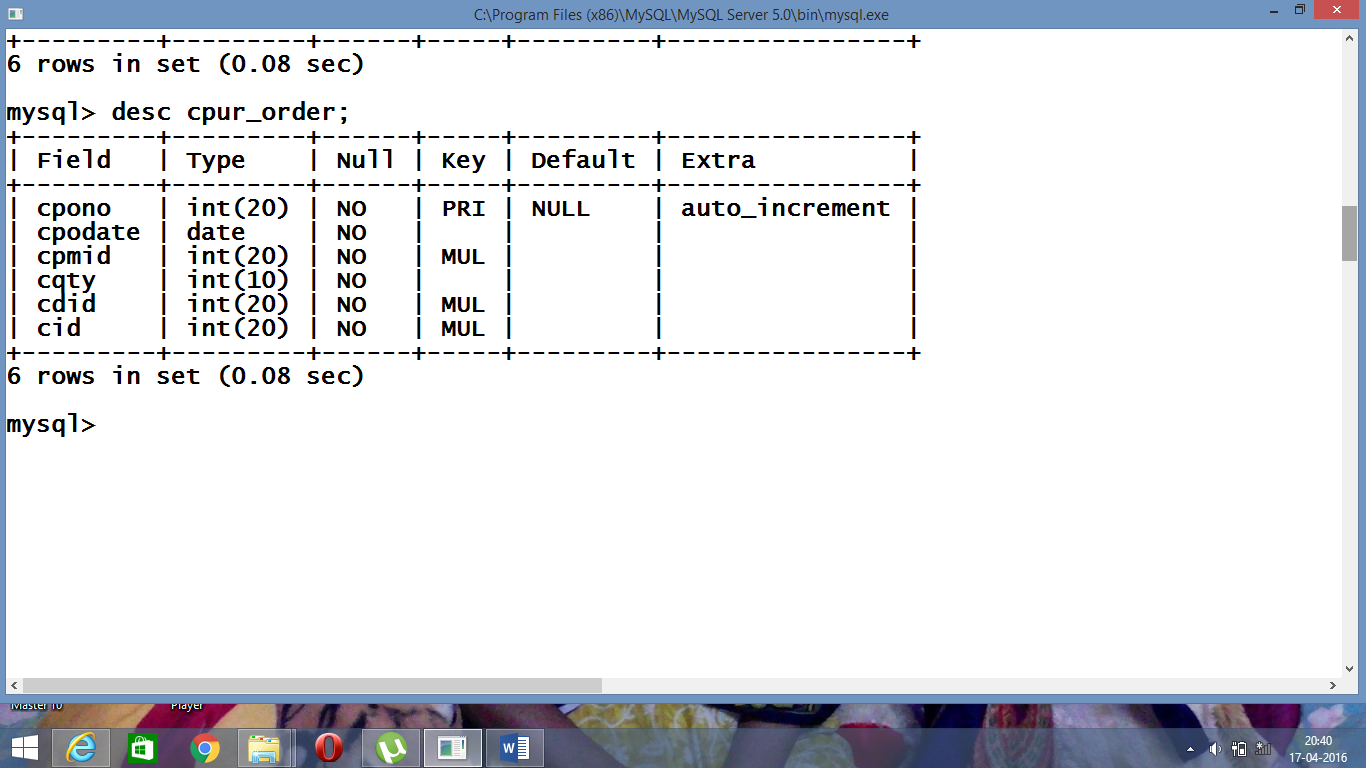
# TABLE: DPUR\_ORDER

# 

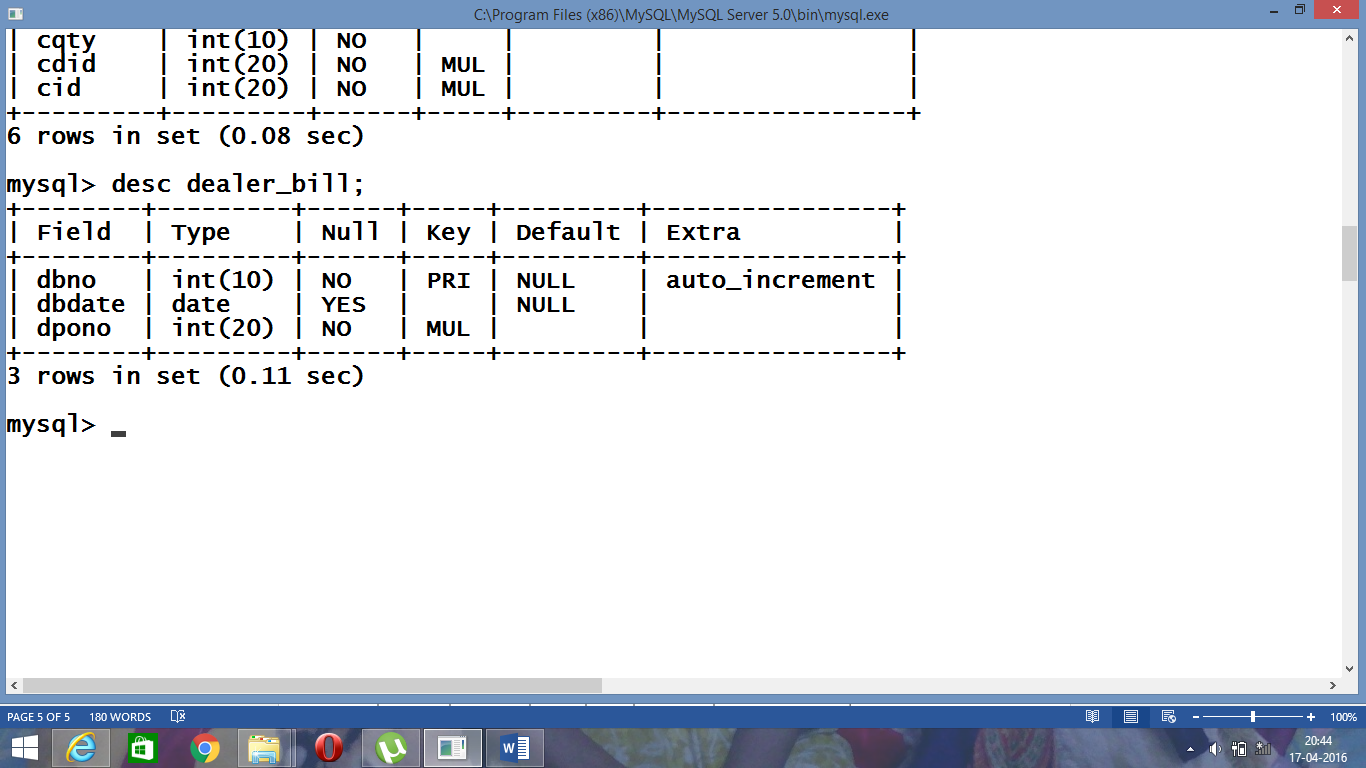
# TABLE: RPUR\_ORDER



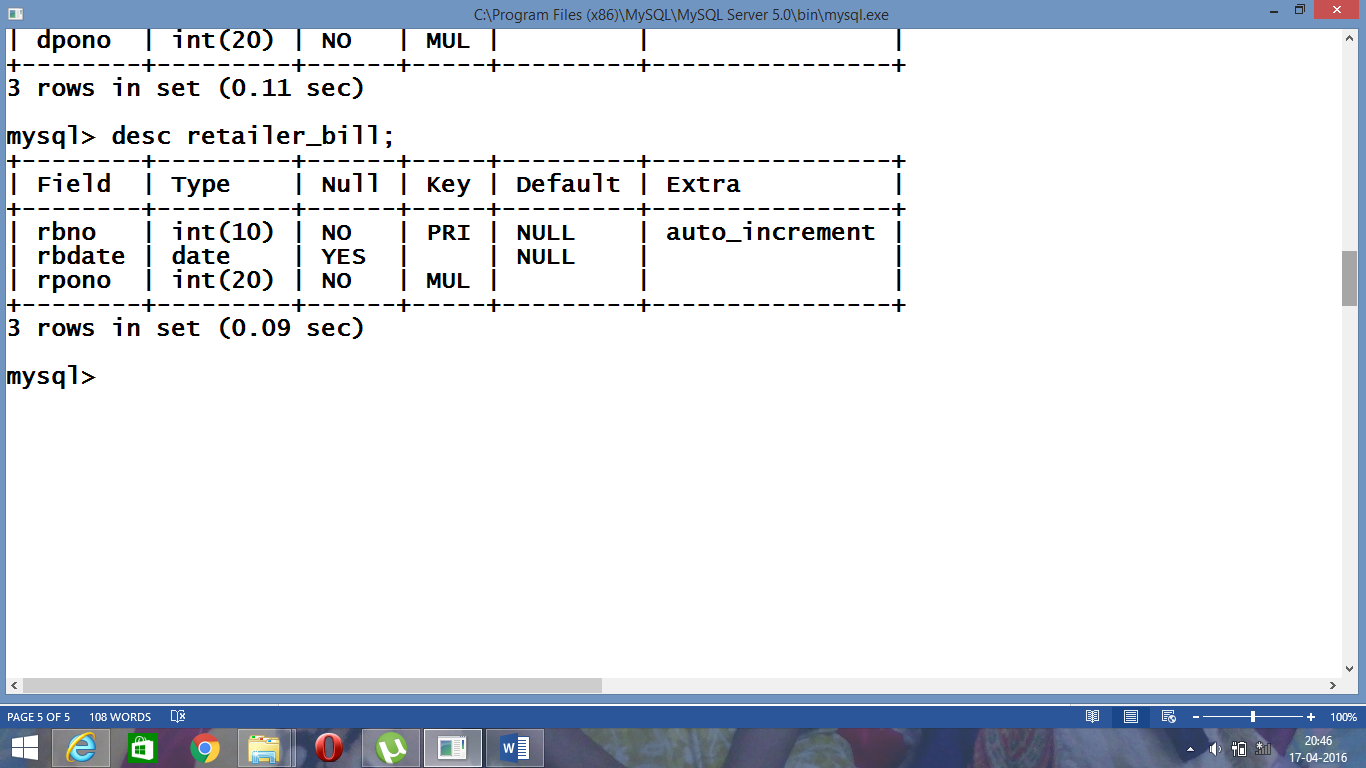
# TABLE: CPUR\_ORDER



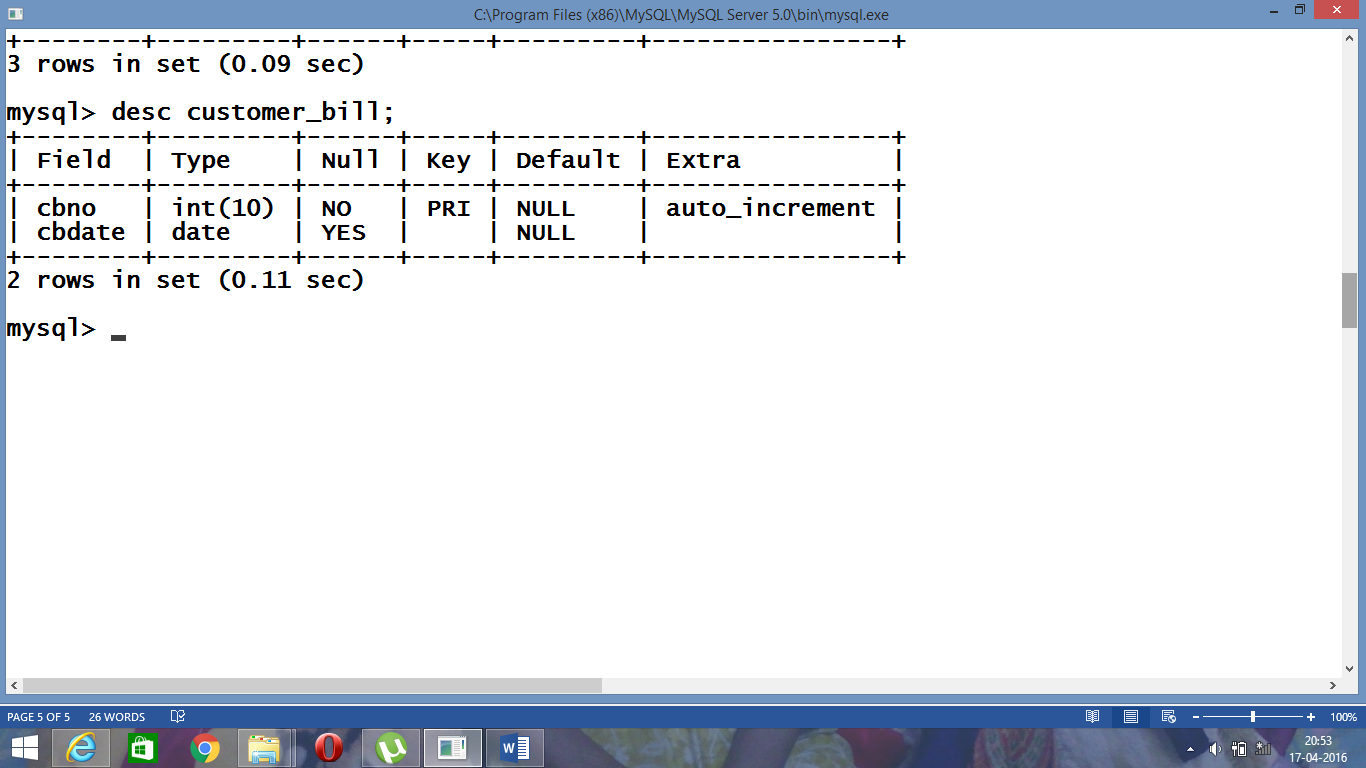
# TABLE: DEALER\_BILL



# TABLE: RETAILER\_BILL



# TABLE: CUSTOMER\_BILL



**4.2.1 ER-Diagram**

dbill

manufacturer

generate

appoints

has

orders

dealer

products

generate

appoints

rbill

orders

orders

retailer

customer

generate

rbill

**4.2.2 ER-Schema**

**Product**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| pid | Pname | Model | Product\_color | DPrice | RPrice | Cprice |

**Product Manufacturer Detail**

|  |  |  |
| --- | --- | --- |
| Pmid | pid | mid |

**Manufacturer**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| mid | Mname | Maddress | MEmail | Mphone | Mcountry |

**Dealer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Did | Dname | Daddress | DEmail | Dphone |

**Retailer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rid | Rname | Raddress | REmail | Rphone |

**Customer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cid | cname | caddress | cemail | cphone |

**DPurchaseOrder**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dpono | dpodate | dpmid | dquty | did |

**RPurchaseOrder**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| rpono | rpodate | rpmid | rquty | rdid | rid |

**CPurchaseOrder**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| cpono | cpodate | cpmid | cquty | crid | cid |

**DBill**

|  |  |  |
| --- | --- | --- |
| dbillno | dbilldate | dpono |

**RBill**

|  |  |  |
| --- | --- | --- |
| rbillno | rbilldate | rpono |

**CBill**

|  |  |
| --- | --- |
| cbillno | cbilldate |

**CBillDetails**

|  |  |
| --- | --- |
| cbillno | cpono |

**4.3 Frontend Design**

**4.3.1Form Design**

The front end forms which have been designed in online E-Advertising System are based on the requirements specified by the end users of the system been built. The design process of the forms was carried out using the technologies like HTML (Hyper Text Markup Language) along with CSS (Cascading Style Sheet) which is used to maintain the uniform and professional look throughout the website. JavaScript is been used to validate inputs by the users in all the forms. For a better graphical view Jquery is used for Login Screen.

## HTML (HYPER TEXT MARKUP LANGUAGE)

HTML is a language for describing web pages.

* HTML stands for **H**yper **T**ext **M**arkup **L**anguage and it is the language in which, until recently, virtually all Web pages were written.
* HTML is not a programming language, it is a **markup**language.
* A markup language is a set of markup**tags.**
* The tags **describe** document content.
* Hypertext refers to the way in which Web pages (HTML documents) are linked together. When you click a link in a Web page, you are using hypertext. It is this system of linking documents that has made the World Wide Web the global phenomenon it has become.
* HTML documents containHTML**tags** and plain **text**

## HTML TAGS

HTML markup tags are usually called HTML tags

* HTML tags are keywords surrounded by **angle brackets**like <html>
* HTML tags normally **come in pairs** like <b> and </b>
* The first tag in a pair is the **start tag,** the second tag is the **end tag**
* The end tag is written like the start tag, with a **forward slash** before the tag name
* Start and end tags are also called **opening tags** and **closing tags.**

**Example:-**

<p>Employee Fleet</p>

## HTML ELEMENTS

* "HTML tags" and "HTML elements" are often used to describe the same thing.
* But strictly speaking, an HTML element is everything between the start tag and the end tag, including the tags.

**Example:-**

If you want to create a table, you put the table information inside the table element <table> </table>.

Similarly To construct a form, you need the form element <form> </form>.

**ATTRIBUTES AND VALUES**

* Attributes are another important part of HTML markup.
* An attribute is used to define the characteristics of an element and is placed inside the element’s opening tag.

**Syntax:-**

<tag name attribute1=”atttibute1 value” attribute2=”atttibute2 value” ….>

**Example:-**

<img height="200" width="200" />

**HTML5**

* HTML5 is the latest standard for HTML.
* The previous version of HTML, HTML 4.01, came in 1999, and the internet has changed significantly since then.
* HTML5 was designed to replace both HTML 4, XHTML, and the HTML DOM Level 2.
* HTML5 is also cross-platform. It is designed to work whether you are using a PC, or a Tablet, a Smartphone, or a Smart TV.

**Example:-**

<video src=”../IMAGES/abc.mp4” autoplay></video>

## HTML5 - NEW FEATURES

* The <canvas> element for 2D drawing
* The <video> and <audio> elements for media playback
* Support for local storage
* New content-specific elements, like <article>, <footer>, <header>, <nav>, <section>
* New form controls, like calendar, date, time, email, url, search

## CSS(CASCADING STYLE SHEET)

* **CSS** stands for **C**ascading **S**tyle **S**heets
* Styles define **how to display** HTML elements
* Styles were added to HTML 4.0 **to solve a problem**
* **External Style Sheets** can save a lot of work
* External Style Sheets are stored in **CSS files.**

## CSS SYNTAX:

## h1 {Color:blue;font-size:12px;}

SelectorName { Declarations}

* The selector points to the HTML element you want to style.
* The declaration block contains one or more declarations separated by semicolons.
* Each declaration includes a property name and a value, separated by a colon.

# JAVASCRIPT

* JavaScript is the most popular programming language in the world.
* JavaScript is the language for the web, for HTML, for servers, PCs, laptops, tablets, cell phones, and more.
* A Java scripting language is a lightweight programming language.
* JavaScript code can be inserted into any HTML page, and it can be executed by all types of web browsers.
* You can place an unlimited number of scripts in an HTML document.
* Scripts can be in the <body> or in the <head> section of HTML, and/or in both.
* It is a common practice to put functions in the <head> section, or at the bottom of the page.
* Separating HTML and JavaScript, by putting all the code in one place, is always a good habit.

**WRITING JAVASCRIPT**

* JavaScript code is case sensitive
* White space between words and tabs are ignored
* Line breaks are ignored except within a statement
* JavaScript statements end with a semi- colon(;)

**THE SCRIPT TAG**

* The <SCRIPT> tag alerts a browser that JavaScript code follows. It is typically embedded in the HTML.
* **Syntax:-**

<Script type=”text/javascript”>

Statement1;

Statement2;

</Script>

* **Example**:-<Script type=”text/javascript”>Alert(“Employee Fleet Welcomes you”);</Script>

# 

## SERVLETS

* Java Servlets are programs that run on a Web or Application server and act as a middle layer between a request coming from a Web browser or other HTTP client and databases or applications on the HTTP server.
* Using Servlets, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.
* Java Servlets often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But Servlets offer several advantages in comparison with the CGI.
* Performance is significantly better.
* Servlets execute within the address space of a Web server. It is not necessary to create a separate process to handle each client request.

## JAVASERVER PAGES (JSP)

* JavaServer Pages (JSP) is a technology for developing web pages that support dynamic content which helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.
* A JavaServer Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands.
* Using JSP, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.
* JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages and sharing information between requests, pages etc.
* Performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself instead of having a separate CGI files.
* JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.

## RELATIONAL DATABASE MANAGEMENT SYSTEM

* Relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as foreign keys.
* Enables you to implement a database with tables, columns and indexes.
* Guarantees the Referential Integrity between rows of various tables.
* Updates the indexes automatically.
* Interprets an SQL query and combines information from various tables.

**TOMCAT SERVER:**

* Tomcat is an open source web server developed by Apache Group.
* Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies.
* The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process.
* Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server).
* To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.

## JAVA

* Initially the language was called as “oak” but it was renamed as “java” in 1995.
* The primary motivation of this language was the need for a platform-independent (i.e. architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.
* java expands the Universe of objects that can move about freely in Cyberspace.
* In the areas of Security and probability. Java addresses that , has opened the door to an exciting new form of program called the Applet.
* Java architecture provides a portable, robust, high performing environment for development.
* Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

**FEATURES OF JAVA**

* Object Oriented, Platform independent, Simple, Secure, Architectural- neutral, Portable, Robust, Multi-threaded, Interpreted, High Performance.

## JAVA BASIC SYNTAX

* **Object -** Objects have states and behaviors.
* **Class -** A class can be defined as a template/ blue print that describe the behaviors/states that object of its type support.
* **Methods -** A method is basically a behavior. A class can contain many methods. It is in methods where the logics are written, data is manipulated and all the actions are executed.
* **Instant Variables -** Each object has its unique set of instant variables. An object's state is created by the values assigned to these instant variables.

**JAVA DATABASE CONNECTIVITY(JDBC)**

* JDBC is a part of java enterprise API for executing SQL Statements. The ODBC API which is written in c language cannot be made use of by java directly because it involves the use of pointers and other constructs that the java does not support, therefore a JDBC-ODBC bridge was developed that translates the JDBC API to ODBC API.
* JDBC consists of a set of classes and interfaces written in the Java Programming language.
* JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API
* Using JDBC, it is easy to send SQL statements to virtually any relational database.

**SESSION’s**

* *Session’s* are a mechanism that servlets use to maintain state about a series of requests from the same user across some period of time Using sessions in servlets is straightforward.

**Session Involves four basic steps:-**

* *Accessing the session object associated with the current request(***request.getSession** )
* *Looking up information associated with a session(***getAttribute)**
* *Storing information in a session.(* **setAttribute)**
* *Discarding session data.(* **removeAttribute)**

**4.3.2 UML DESIGN:**

* **Administrator Use Case Diagram**

Admin

**Manufacturer Use Case Diagram**

Manufacture

**Dealer Use Case Diagram**

Dealer

**Retailer Use Case Diagram**

Retailer

**Customer Use Case Diagram**

Customer

**4.4 Module Description in brief**

* **Manufacturer module**

In this module the manufacturer can add the product, update/view/delete the product.he can also view the detail of dealer. Manufacturer can sell the product to the particular dealer. .

* **Dealer module**

In this module the dealer can view the available products and detail of manufacturer, dealer can know the information about the product he can also make a purchase order to the product. he can also know the cost of the particular product.

* **Retailer module**

In this module the retailer can view the available products and detail of dealers, he can also interact with dealers to know about the products and can make a purchase order to the product with the dealer, retailer can sell the product to the customer and generate the bill for the purchased product made by the customer.

* **Customer module**

In this module the customer can view the available product and can also make the purchase order for the product from the retailer, he can make a payment for the purchased product and can receive the bill from the retailer.