INFO6210 - Data Management and Database Design

RDBMS for Online learning platform Ecommerce

PROJECT REPORT

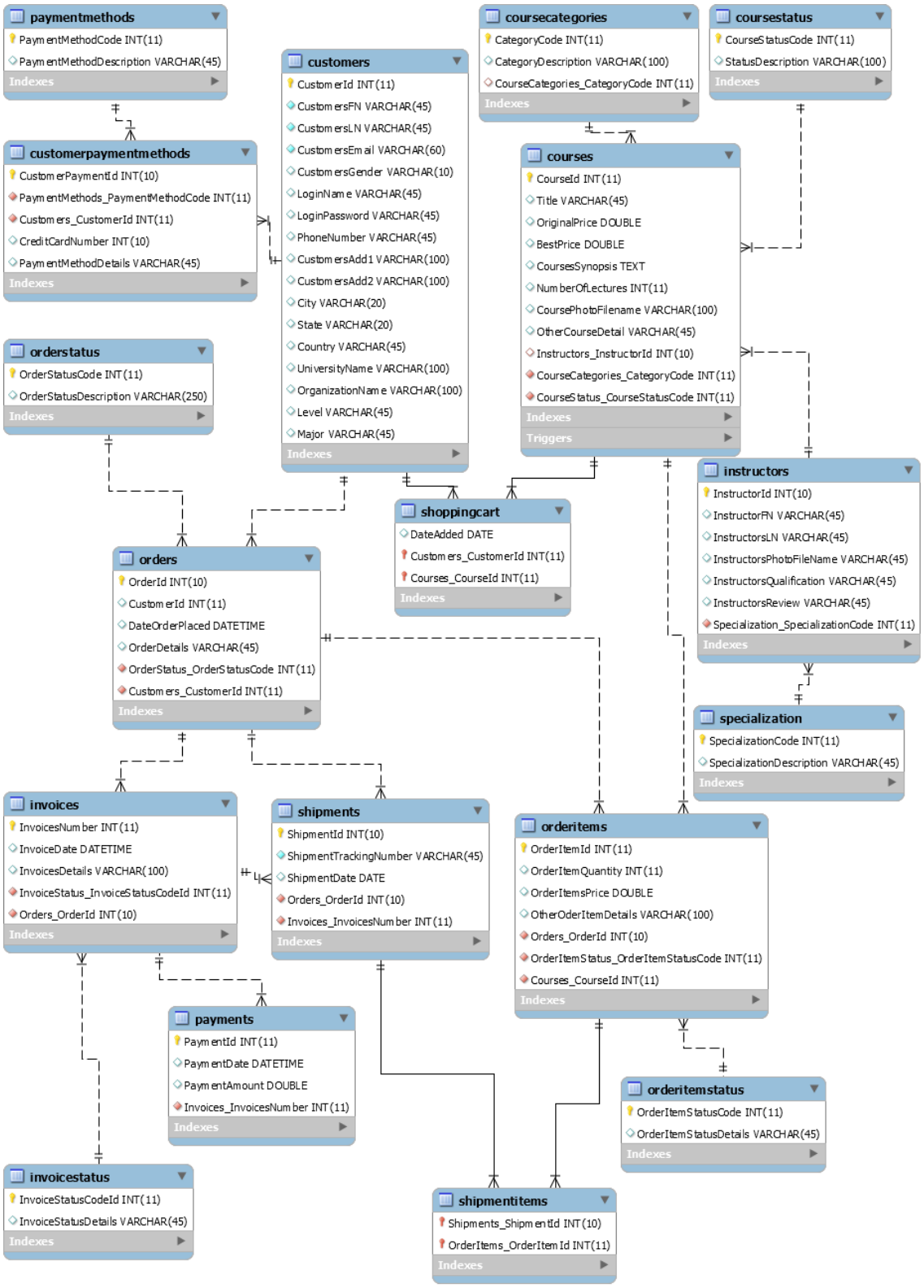
Submitted by:
Krapali Rai
NUID:
001813750

Problem Walkthrough:

- The vision is to create a relational database management system that will illustrate a entity relations amongst several real world entities of MOOC(Massive Open online courses) like ecommerce.
- The design covers the scenarios for online course selling sites which comes under a E2C type ecommerce.
- The model has been implemented using MySQL.

SOLUTION PROPOSED:

- The EER model for the ecommerce is designed using MySQL Workbench. The model will be able to manage data for several entities like customer, courses, instructors, payments, invoices and shipments.
- A customer are the students who will search for courses online where they can select courses to learn online, can opt for live classroom and a course material package will be shipped to the student on ordering.
- The database user will be able to check the instructors for corresponding courses. Also user will be able to enter values for entities like customer, orders, courses, instructors, specialization, invoices.
- Design consists of views, triggers, procedure for making the DB more user friendly and perform runtime user requests.


paymentmethods
PaymentMethodCode INT(11)
PaymentMethodDescription VARCHAR(45)
Indexes
customerpaymentmethods
CustomerPaymentId INT(10)
PaymentMethods_PaymentMethodCode INT(11)
Customers_CustomerId INT(11)
CreditCardNumber INT(10)
PaymentMethodDetails VARCHAR(45)
Indexes
customers
CustomerId INT(11)
CustomersFN VARCHAR(45)
CustomersLN VARCHAR(45)
CustomersEmail VARCHAR(60)
CustomersGender VARCHAR(10)
LoginName VARCHAR(45)
LoginPassword VARCHAR(45)
PhoneNumber VARCHAR(45)
CustomersAdd1 VARCHAR(100)
CustomersAdd2 VARCHAR(100)
City VARCHAR(20)
State VARCHAR(20)
Country VARCHAR(45)
UniversityName VARCHAR(100)
OrganizationName VARCHAR(100)
Level VARCHAR(45)
Major VARCHAR(45)
Indexes
coursecategories
CategoryCode INT(11)
CategoryDescription VARCHAR(100)
CourseCategories_CategoryCode INT(11)
Indexes
coursestatus
CourseStatusCode INT(11)
StatusDescription VARCHAR(100)
Indexes
courses
CourseId INT(11)
Title VARCHAR(45)
OriginalPrice DOUBLE
BestPrice DOUBLE
CoursesSynopsis TEXT
NumberOfLectures INT(11)
CoursePhotoFilename VARCHAR(100)
OtherCourseDetail VARCHAR(45)
Instructors_InstructorId INT(10)
CourseCategories_CategoryCode INT(11)
CourseStatus_CourseStatusCode INT(11)
Indexes
Triggers
orderstatus
OrderStatusCode INT(11)
OrderStatusDescription VARCHAR(250)
Indexes
instructors
InstructorId INT(10)
InstructorFN VARCHAR(45)
InstructorsLN VARCHAR(45)
InstructorsPhotoFileName VARCHAR(45)
InstructorsQualification VARCHAR(45)
InstructorsReview VARCHAR(45)
Specialization_SpecializationCode INT(11)
Indexes
shoppingcart
DateAdded DATE
Customers_CustomerId INT(11)
Courses_CourseId INT(11)
Indexes
orders
OrderId INT(10)
CustomerId INT(11)
DateOrderPlaced DATETIME
OrderDetails VARCHAR(45)
OrderStatus_OrderStatusCode INT(11)
Customers_CustomerId INT(11)
Indexes
specialization
SpecializationCode INT(11)
SpecializationDescription VARCHAR(45)
Indexes
invoices
InvoicesNumber INT(11)
InvoiceDate DATETIME
InvoicesDetails VARCHAR(100)
InvoiceStatus_InvoiceStatusCodeId INT(11)
Orders_OrderId INT(10)
Indexes
shipments
ShipmentId INT(10)
ShipmentTrackingNumber VARCHAR(45)
ShipmentDate DATE
Orders_OrderId INT(10)
Invoices_InvoicesNumber INT(11)
Indexes
orderitems
OrderItemId INT(11)
OrderItemQuantity INT(11)
OrderItemsPrice DOUBLE
OtherOderItemDetails VARCHAR(100)
Orders_OrderId INT(10)
OrderItemStatus_OrderItemStatusCode INT(11)
Courses_CourseId INT(11)
Indexes
payments
PaymentId INT(11)
PaymentDate DATETIME
PaymentAmount DOUBLE
Invoices_InvoicesNumber INT(11)
Indexes
orderitemstatus
OrderItemStatusCode INT(11)
OrderItemStatusDetails VARCHAR(45)
Indexes
invoicestatus
InvoiceStatusCodeId INT(11)
InvoiceStatusDetails VARCHAR(45)
Indexes
shipmentitems
Shipments_ShipmentId INT(10)
OrderItems_OrderItemId INT(11)
Indexes

Triggers:

Trigger 1: This Trigger is a **Before Insert** trigger implemented on **Courses** entity.

The trigger will change the course status to **coursestatuscode = 0**, which indicates that the course is not available if the there is no instructor assigned for the course.

Trigger 2: This trigger is a Before Insert trigger implemented on Courses table.

The trigger will fire when the **Best price** attribute is null. Then the best price of the course will enter a value that is present in the **Original Price** with a 20 % applied discount.

```
CREATE
DEFINER=`root`@`localhost`
TRIGGER `myproject`.`courses_BEFORE_INSERT`
BEFORE INSERT ON `myproject`.`courses`
FOR EACH ROW
BEGIN

IF new.Instructors_InstructorId Is null then
set new.CourseStatus_coursestatuscode = 0;

end if;


IF new.BestPrice Is null then
set new.BestPrice = (new.OriginalPrice * 0.8);

end if;

END
```

STORED PROCEDURES:

Procedure to fetch details of shipment of a customer.

```
CREATE DEFINER=`root`@`localhost` PROCEDURE
 `Shipment_Details`(in customerId int)
BEGIN

Select concat_ws(' ',customers.customersFN,
customers.customersLN) as StudentName,
customers.customerId ,
orders.orderId,
orders.orderdetails,
shipments.shipmentTrackingNumber,
shipments.shipmentDate

From Customers Inner join Orders
On customers.CustomerId = orders.CustomerId

Inner join shipments
On orders.orderid = shipments.Orders_OrderId
;

END
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentName	customerId	orderId	orderdetails	shipmentTrackingNumber	shipmentDate
krapali rai	70000001	10001	Onlinepurchase	UPS1234567890	2017-12-01
krapali rai	70000001	10002	ExamSupport	UPS1234567891	2017-12-02

STORED PROCEDURE 2:

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `Invoice_Payment_details`(in cust_ID int)
BEGIN

SELECT
    customers.CustomerId,
    customers.CustomersFN AS FirstName,
    customers.CustomersLN AS LastName,
    orders.OrderId,
    orders.OrderDetails,
    invoices.invoicesnumber,
    invoices.invoicesDetails,
    payments.paymentAmount,
    payments.paymentDate

FROM
    customers
    INNER JOIN
    orders
    ON customers.CustomerId
    = orders.CustomerId

    INNER JOIN
    invoices
    ON orders.OrderId
    = invoices.Orders_OrderId

    INNER JOIN
    payments
    ON invoices.invoicesNumber
    = payments.Invoices_InvoicesNumber

    where customers.CustomerId = cust_ID;

END
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CustomerId	FirstName	LastName	OrderId	OrderDetails	invoicesnumber	invoicesDetails	paymentAmount	paymentDate
70000001	krapali	rai	10001	Onlinepurchase	4701	payment recieved	80	2017-12-12 00:00:00
70000001	krapali	rai	10002	ExamSupport	4702	payment done	100	2017-12-07 00:00:00

VIEWS:

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `course_instructor_search` AS
    SELECT
        `courses`.`Title` AS `Title`,
        `courses`.`CoursesSynopsis` AS `CoursesSynopsis`,
        `courses`.`NumberOfLectures` AS `NumberOfLectures`,
        `instructors`.`InstructorFN` AS `InstructorFN`,
        `instructors`.`InstructorsLN` AS `InstructorsLN`,
        `instructors`.`InstructorsQualification` AS `InstructorsQualification`,
        `specialization`.`SpecializationDescription` AS `SpecializationDescription`
    FROM
        ((`courses`
        JOIN `instructors`
        ON ((`courses`.`Instructors_InstructorId` = `instructors`.`InstructorId`)))
        JOIN `specialization`
        ON ((`instructors`.`Specialization_SpecializationCode` = `specialization`.`SpecializationCode`)))
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Title	CoursesSynopsis	NumberOfLectures	InstructorFN	InstructorsLN	InstructorsQualification	SpecializationDescription
DB	DBMS	10	Yusuf	Ozbek	PhD	DBMS
Java	Netbeans	5	kal	bugrara	Phd	JAVA
DB2	DBMS	20	Yusuf	Ozbek	PhD	DBMS
DB2	DBMS	20	Yusuf	Ozbek	PhD	DBMS

VIEW 2:

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `custmerpayment_details` AS
    SELECT
        `customers`.`CustomersFN` AS `FirstName`,
        `customers`.`CustomersLN` AS `LastName`,
        `customerpaymentmethods`.`Customers_CustomerId` AS `Customers_CustomerId`,
        `paymentmethods`.`PaymentMethodDescription` AS `PaymentMethodDescription`,
        `customerpaymentmethods`.`CreditCardNumber` AS `CreditCardNumber`
    FROM
        ((`paymentmethods`
        JOIN `customerpaymentmethods`
        ON ((`paymentmethods`.`PaymentMethodCode` =
        `customerpaymentmethods`.`PaymentMethods_PaymentMethodCode`)))
        JOIN `customers`
        ON ((`customerpaymentmethods`.`Customers_CustomerId` = `customers`.`CustomerId`)))
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

FirstName	LastName	Customers_CustomerId	PaymentMethodDescription	CreditCardNumber
krapali	rai	70000001	Credit Card	1234567890
krapali	rai	70000001	Debit Card	1234567891
mayank	gangrade	70000002	cheque	1234567892

VIEW 3:

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `studentcoursedetails` AS
    SELECT
        `customers`.`CustomersFN` AS `FirstName`,
        `customers`.`CustomersLN` AS `LastName`,
        `orders`.`OrderDetails` AS `OrderDetails`,
        `orderitems`.`OrderItemsPrice` AS `OrderItemsPrice`,
        `courses`.`Title` AS `Title`,
        `courses`.`BestPrice` AS `BestPrice`,
        `coursecategories`.`CategoryDescription` AS `CategoryDescription`
    FROM
        ((((`customers`
        JOIN `orders`
        ON ((`customers`.`CustomerId` = `orders`.`CustomerId`)))
        JOIN `orderitems`
        ON ((`orders`.`OrderId` = `orderitems`.`Orders_OrderId`)))
        JOIN `courses`
        ON ((`orderitems`.`Courses_CourseId` = `courses`.`CourseId`)))
        JOIN `coursecategories`
        ON ((`courses`.`CourseCategories_CategoryCode` = `coursecategories`.`CategoryCode`)))
```

Output:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

FirstName	LastName	OrderDetails	OrderItemsPrice	Title	BestPrice	CategoryDescription
krapali	rai	Onlinepurchase	80	DB	80	Online Video Lecture
krapali	rai	ExamSupport	90	Java	800	Online Video Lecture
krapali	rai	Online purchase	80	DB2	800	Online Video Lecture
krapali	rai	Onlinepurchase	100	DB	80	Online Video Lecture