ONL

database design for ONLINE BOOK STORE

CS 6360.003 FINAL PROJECT

Table of Contents

[Requirements 2](#_Toc468483892)

[Services offered by Fire Department 2](#_Toc468483893)

[ Emergency Services 2](#_Toc468483894)

[ Non-Emergency services 2](#_Toc468483895)

[Department Structure 2](#_Toc468483896)

[Employee Structure and Minimum Wage Act 2](#_Toc468483897)

[Incidences 3](#_Toc468483898)

[Equipment and Classification 3](#_Toc468483899)

[Apparatus 3](#_Toc468483900)

[Compensation for LOD 4](#_Toc468483901)

[Modeling of Requirements as ER-Diagram 5](#_Toc468483902)

[Mapping of ERD in Relational Schema 7](#_Toc468483903)

[SQL Statements to create Relations in DB and Add Constraints 9](#_Toc468483904)

[PL/SQL – Triggers 14](#_Toc468483905)

[Trigger-I INVENTORY 14](#_Toc468483906)

[Trigger-II LODENTRY 15](#_Toc468483907)

[Trigger-III: Minimum Wage & Salary Constraints 15](#_Toc468483908)

[PL/SQL- Procedures 18](#_Toc468483909)

[Procedure-I Calculating DIC 18](#_Toc468483910)

[Procedure-II Auxiliary 19](#_Toc468483911)

[Procedure-III Reporting and Analysis 19](#_Toc468483912)

# Requirements

Services offered by Online Book store

This is a book purchasing online store where customers can go online, make their user profiles and make multiple purchases of the books. The services offered by the store are as follows.

* Services include –
  + Creating Customer profiles to the Online stores
  + Book Browsing with Image
  + Adding of books to customer specific Cart
  + Keeping a track of Book availability
  + Placing Orders
  + Reflecting the count of books in Available Books
  + Deleting the Customer Profile
  + Storing Order History of every customer

Customer Structure

Each Customer has a unique Customer ID which will be generated when they create their profiles. While creating their profiles as customers they will be asked for their Email IDs and will be prompted to choose a password. Email ID of the customer should be unique to each customer. Other information required to create a Customer will be their Phone numbers, First names, last names and their addresses.

Books

Each customer will browse for books of their choice. Books will be uniquely identified by their Book IDs. But there may be many books of the same ID. Thus, we keep a count of the books of same IDs. Each book will be associated with a Category to which it belongs. A book can be browsed by its Title. All the information regarding a book are available like its price, Publisher, Published date, version. If a customer purchases a book, it will reflect in the count of the book.

Author

There are many authors. One author may have written multiple books and vice versa. An author is identified uniquely by an Author ID. Also, information about author like Name, Email ID, and address of the author is available.

Category

Each book is allotted to a different category. For example: Physics, Science, Math etc. A category is distinguished by a Category ID and every book is allotted to one category. A category has characteristic like Name and popularity. Also, we keep a count of all the books which belong to a category.

Cart

A cart is associated with every customer directly. Each customer will have exactly one cart and can place purchase orders via it. Multiple books can be added to a cart by the customer and it will also show if the books in the cart are available at the time of adding the book to the cart. The book and price of the book will get added. Also the time of adding the book to the cart is noted.

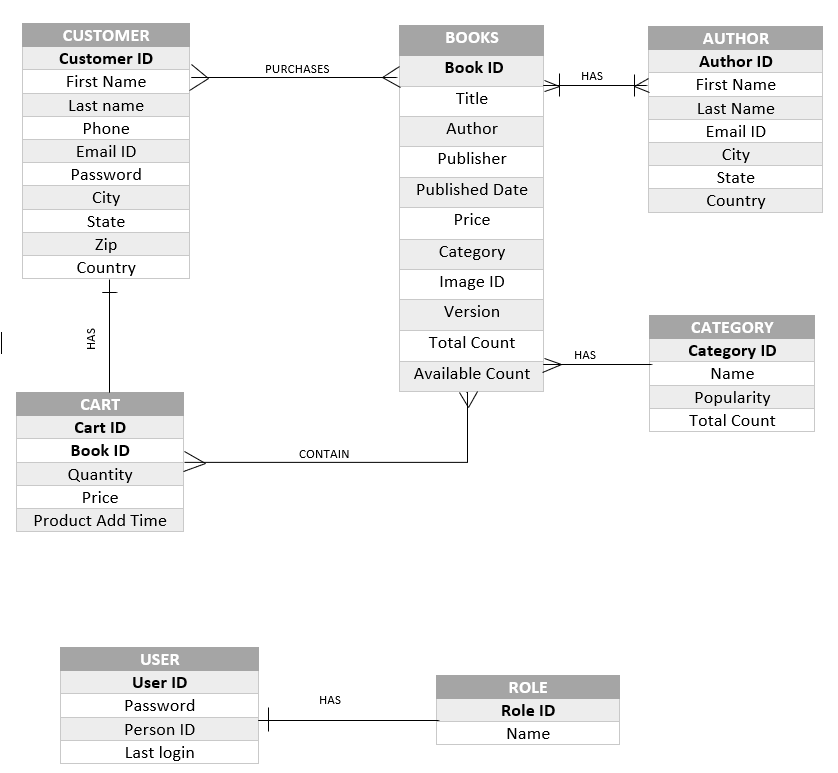
Purchase

A Customer can place the order with the items they have in their cart currently. The books in the order will be checked for availability. If the Books are available, the total price will be calculated. Each purchase is unique by a Purchase ID and the Customer buying the items. Time of purchase is recorded. A customer can choose any shipping address for the delivery of the books they ordered.

Users

The Users are different roles a profile may have in the system. Each role has different responsibility or roles. A user may be a Customer, or a seller, or Admin. All the Created accounts for a customer is by default assigned to a particular User type.

Modeling of Requirements as ER-Diagram:



The requirements can be summarized/ derived from ERD as –

1. A Customer can have one cart and a cart can belong to one customer (1:1).
2. A Book can have many authors and an author can write many books (M:N).
3. A book belongs to one category, But a Category can have many books listed under it. (1:M)
4. A book can be added in many Carts and a cart can have many books (M:N)
5. A Customer can purchase Many books and a book can be purchased by many customers(M:N).
6. A User has to have at least one role but a role might not have users (0:1)

Mapping of ERD in Relational Schema

1. CUSTOMER

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Customer ID | First name | Last Name | Phone | Email | Password | City | State | Country | Zip |

* Primary Key : Customer ID
* Foreign Keys : None

1. BOOKS

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Book ID | Title | Version | Publisher | Published | Price | Total count | Available count | Image ID | Category |

* Primary Key : BOOK\_ID
* Foreign Keys : FOREIGN KEY (Category) REFERENCES CATEGORY(CATEGORY\_ID)

1. AUTHOR

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Author ID | First name | Last name | Email | City | State | Country |

* Primary Key : AUTHOR\_ID
* Foreign Keys : None

1. CART

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cart ID | Book ID | Quantity | Price | Product add time |

* Primary Key : CART ID, BOOK ID
* Foreign Keys : FOREIGN KEY (BOOK\_ID) REFERENCES BOOKS(BOOK\_ID)

FOREIGN KEY (CART\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID)

1. CATEGORY

|  |  |  |  |
| --- | --- | --- | --- |
| Category ID | Name | Popularity | Total count |

* Primary Key : CATEGORY ID
* Foreign Keys : None

1. ROLE

|  |  |
| --- | --- |
| Role ID | Name |

* Primary Key : ROLE ID
* Foreign Keys : None

1. USER

|  |  |  |  |
| --- | --- | --- | --- |
| User ID | Password | Person ID | Last Login |

* Primary Key : USER ID
* Foreign Keys : FOREIGN KEY (DNO) REFERENCES FIREDEPARTMENT(DNUMBER)

1. BOOKS\_AUTHORS

|  |  |
| --- | --- |
| Book ID | Author ID |

* Primary Key : BOOK ID, AUTHOR ID
* Foreign Keys : FOREIGN KEY (BOOK ID) REFERENCES BOOKS(BOOK ID),

FOREIGN KEY (AUTHOR ID) REFERENCES AUTHOR (AUTHOR ID)

1. PURCHASE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Purchase ID | Customer ID | Total Price | Total items | Shipping Address | Purchase Date |

* Primary Key : PURCHASE ID, CUSTOMER ID
* Foreign Keys : FOREIGN KEY (CUSTOMER ID)REFERENCES CUSTOMER (CUSTOMER ID)

1. PURCHASE DETAILS

|  |  |  |  |
| --- | --- | --- | --- |
| Purchase ID | Book ID | Price | Quantity |

* Primary Key : PURCHASE ID, BOOK ID
* Foreign Keys : FOREIGN KEY (PURCHASE ID) REFERENCES PURCHASE(PURCHASE ID), FOREIGN KEY (BOOK ID) REFERENCES BOOKS(BOOK ID)

SQL Statements to create Relations in DB and Add Constraints

CREATE TABLE `remedscx\_`.`bs\_books` (

  `book\_id` INT NOT NULL,

  `title` VARCHAR(45) NULL,

  `price` FLOAT NULL,

  `version` VARCHAR(45) NULL,

  `publisher` VARCHAR(45) NULL,

  `published date` VARCHAR(45) NULL,

  `image\_id` VARCHAR(45) NULL,

  `total\_count` VARCHAR(45) NULL,

  `available\_count` VARCHAR(45) NULL,

  `category` INT DEFAULT NULL,

  PRIMARY KEY (`book\_id`));

ALTER TABLE `remedscx\_`.`bs\_books`

ADD CONSTRAINT `bs\_books\_fk1`

FOREIGN KEY (`category`)

REFERENCES `remedscx\_`.`bs\_category` (`category\_id`)

ON DELETE RESTRICT

ON UPDATE RESTRICT;

CREATE TABLE `remedscx\_`.`bs\_customer` (

  `customer\_id` INT NOT NULL,

  `email` VARCHAR(45) NULL,

  `phone` VARCHAR(45) NULL,

  `fname` VARCHAR(45) NULL,

  `lname` VARCHAR(45) NULL,

  `city` VARCHAR(45) NULL,

  `state` VARCHAR(45) NULL,

  `zip` VARCHAR(45) NULL,

  `country` VARCHAR(45) NULL,

  `cart\_id` INT DEFAULT NULL,

  PRIMARY KEY (`customer\_id`));

CREATE TABLE `remedscx\_`.`bs\_authors` (

  `author\_id` INT NOT NULL,

  `fname` VARCHAR(45) NULL,

  `lname` VARCHAR(45) NULL,

  `email` VARCHAR(45) NULL,

  `city` VARCHAR(45) NULL,

  `state` VARCHAR(45) NULL,

  `country` VARCHAR(45) NULL,

  PRIMARY KEY (`author\_id`));

 CREATE TABLE `remedscx\_`.`bs\_book\_authors` (

  `book\_id` INT NOT NULL,

  `author\_id` INT NOT NULL,

  PRIMARY KEY (`book\_id`, `author\_id`));

ALTER TABLE `remedscx\_`.`bs\_book\_authors`

ADD CONSTRAINT `bs\_book\_author\_fk1`

FOREIGN KEY (`author\_id`)

REFERENCES `remedscx\_`.`bs\_author` (`author\_id`)

ON DELETE RESTRICT

ON UPDATE RESTRICT,

ADD CONSTRAINT `bs\_book\_author\_fk2`

FOREIGN KEY (`book\_id`)

REFERENCES `remedscx\_`.`bs\_books` (`book\_id`)

ON DELETE RESTRICT

ON UPDATE RESTRICT;

 CREATE TABLE `bs\_category` (

   `category\_id` int(11) NOT NULL,

   `category\_name` varchar(45) DEFAULT NULL,

   `category\_popularity` varchar(45) DEFAULT NULL,

   `quantity` int(11) DEFAULT NULL,

   PRIMARY KEY (`category\_id`)

 ) ;

CREATE TABLE `remedscx\_`.`bs\_cart` (

  `cart\_id` INT NOT NULL,

  `book\_id` INT NOT NULL,

  `quantity` INT NULL,

  `price` VARCHAR(45) NULL,

  `product\_add\_time` TIMESTAMP NULL,

  PRIMARY KEY (`cart\_id`, `book\_id`));

ALTER TABLE `remedscx\_`.`bs\_cart`

ADD CONSTRAINT `bs\_cart)fk1`

FOREIGN KEY (`book\_id`)

REFERENCES `remedscx\_`.`bs\_books` (`book\_id`)

ON DELETE RESTRICT

ON UPDATE RESTRICT;

 CREATE TABLE `remedscx\_`.`bs\_purchase` (

  `purchase\_id` INT NOT NULL,

  `customer\_id` INT NULL,

  `shipping\_address` VARCHAR(45) NULL,

  `total\_cost` FLOAT NULL,

  `total\_items` INT NULL,

  `purchase\_date` TIMESTAMP NULL,

  PRIMARY KEY (`purchase\_id`));

ALTER TABLE `remedscx\_`.`bs\_purchase`

DROP FOREIGN KEY `bs\_purchase\_fk1`;

ALTER TABLE `remedscx\_`.`bs\_purchase`

ADD CONSTRAINT `bs\_purchase\_fk1`

FOREIGN KEY (`customer\_id`)

REFERENCES `remedscx\_`.`bs\_customer` (`customer\_id`)

ON DELETE RESTRICT

ON UPDATE RESTRICT;

CREATE TABLE `remedscx\_`.`bs\_purchase\_details` (

  `purchase\_id` INT NOT NULL,

  `book\_id` INT NOT NULL,

  `quantity` INT NULL,

  `price` FLOAT NULL,

  PRIMARY KEY (`purchase\_id`, `book\_id`));

ALTER TABLE `remedscx\_`.`bs\_purchase\_details`

ADD INDEX `bs\_purchase\_details\_fk2\_idx` (`book\_id` ASC);

ALTER TABLE `remedscx\_`.`bs\_purchase\_details`

ADD CONSTRAINT `bs\_purchase\_details\_fk1`

FOREIGN KEY (`purchase\_id`)

REFERENCES `remedscx\_`.`bs\_purchase` (`purchase\_id`)

ON DELETE RESTRICT

ON UPDATE RESTRICT,

ADD CONSTRAINT `bs\_purchase\_details\_fk2`

FOREIGN KEY (`book\_id`)

REFERENCES `remedscx\_`.`bs\_books` (`book\_id`)

ON DELETE RESTRICT

ON UPDATE RESTRICT;

  CREATE TABLE `bs\_user\_roles` (

   `role\_id` int(11) NOT NULL,

   `role\_name` varchar(45) DEFAULT NULL,

   PRIMARY KEY (`role\_id`)

 );

 CREATE TABLE `remedscx\_`.`bs\_users` (

  `user\_id` VARCHAR(10) NOT NULL,

  `password` VARCHAR(45) NULL,

  `role\_id` INT NULL,

  `person\_id` INT NULL,

  `last\_login` TIMESTAMP NULL,

  `deleted` INT DEFAULT 0,

  PRIMARY KEY (`user\_id`));

Normalization of Relational Schema

The following Functional Dependencies exists in the relational schema –

1. EMPLOYEE {SSN -> Fname, Mint, Lname, Bdate, Address, Sex, Salary, Phone, EType, Current-Status, Supervisor\_ssn, Dno}
2. FIREDEPARTMENT {Dnumber -> Dname, HelpLine, WorkingHours, NoOfCounsel}
3. EQUIPMENT {EquipmentID -> EquipmentType, Description, HandlingCaution}
4. APPARATUS {ApparatusID -> Description, HandlingCaution, Manufacturer, ProcurementDate, ApparatusExpDate, QtyProcured, CurrentStatus}
5. INCIDENCE {IncidenceID -> CurrentAsOf, IncidenceType, Cause, DateOfOrigin, Size\_Impact, PercentageContained, EstContainmentDate, FuelsInvolved, TotalFireEngines}
6. SERVICES {ServiceID -> Category, Description}
7. LODDEATH {DateOfDeath, Cause, Description, DIC}
8. DEPARTMENT LOCATIONS {Dno, Location -> Address}
9. INVENTORY\_MGMT {Serial -> IncID, AppartID, TotalQty}

The above functional dependencies cause the schema to be in third normal form.

# PL/SQL – Triggers

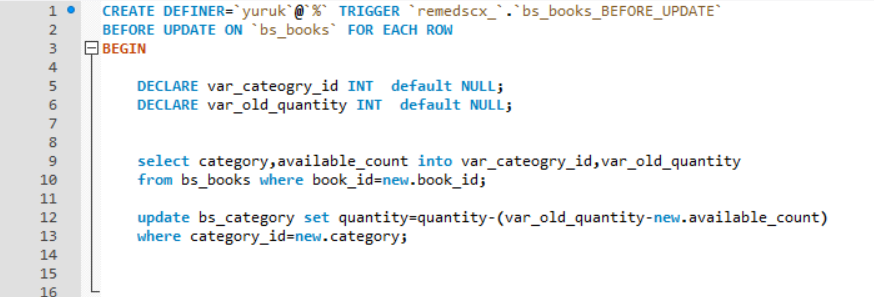
The following triggers are used to implement various requirements –

## Trigger-I CATEGORY

Whenever there is an update in the books , there will be a similar update to the category

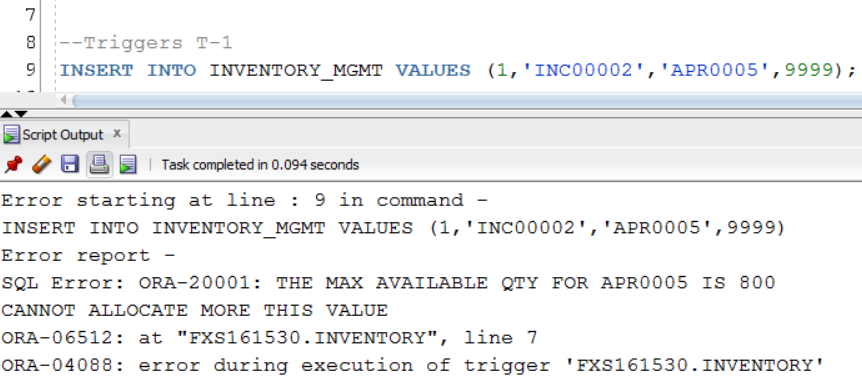
--Check if the qty is available before allocating

* Procedure Called : CHK\_QTY()
* Parameters passed : Apparatus ID, Quantity



* Negative Test Case SQL:

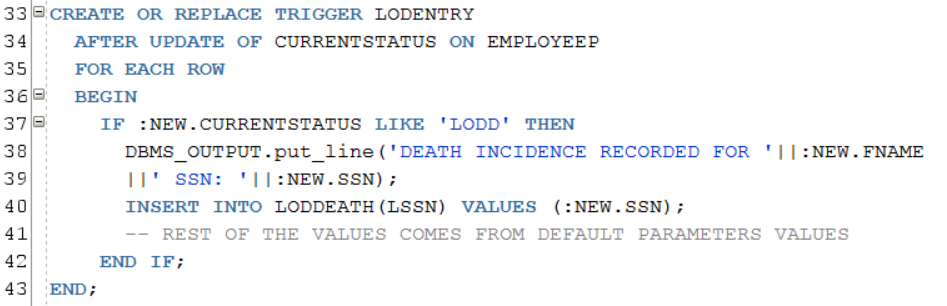
INSERT INTO INVENTORY\_MGMT VALUES (1,'INC00002','APR0005',9999);

* Negative Test Case Output:

## Trigger-II LODENTRY

Whenever status of an employee is updated as ‘LODD' meaning line of duty death, a record in his name is inserted automatically in Loddeath table with default values.

* Procedure Called: None



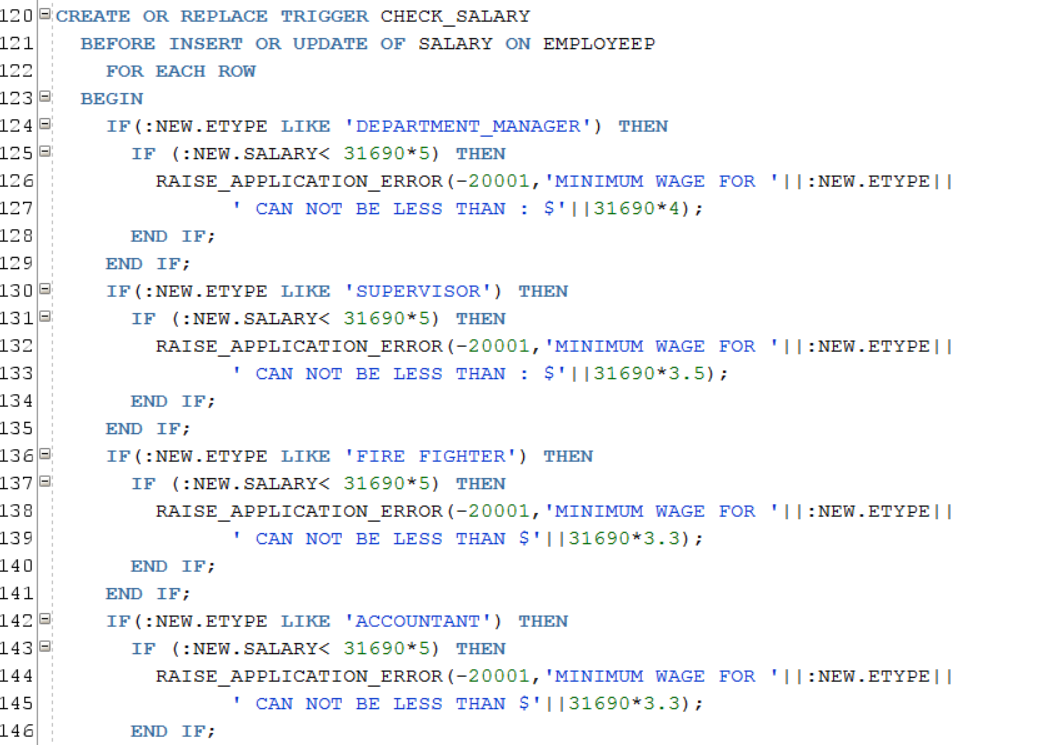
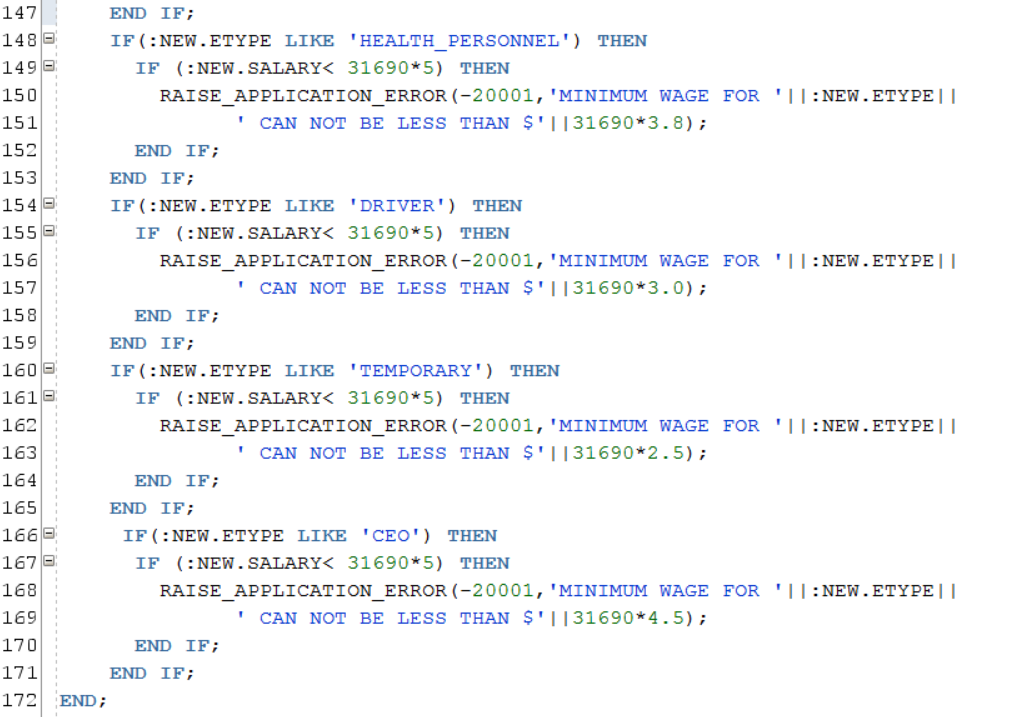
* Triggering SQL:

UPDATE EMPLOYEEP SET CURRENTSTATUS='LODD' WHERE EMPLOYEEP.SSN=123456107;

## Trigger-III: Minimum Wage & Salary Constraints

The above constraint is implemented using a trigger to check the salary if it is in accordance with FLSA with additional min salary requirement provided by Fire Department HR Dept.

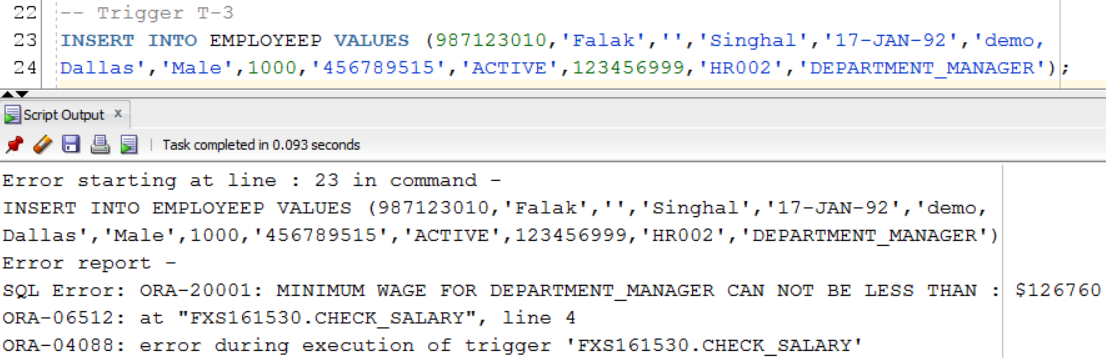
If the salaries are found to be not complying with FLSA, the trigger prevents the inserting of record by raising an application level error. (PTO)



* Negative Test Case SQL:

INSERT INTO EMPLOYEEP VALUES (987123010,'Falak','','Singhal','17-JAN-92','demo,

Dallas','Male',1000,'456789515','ACTIVE',123456999,'HR002','DEPARTMENT\_MANAGER');

* Negative Test Case Output:

# PL/SQL- Procedures

## Procedure-I Register a Customer

This procedure creates a customer profile by accepting the information required for creating a customer and generates a Customer ID for the Customer Profile.



* Arg: (Employee SSN IN)
* Test Case SQL:

DECLARE

THIS\_SSN NUMBER;

BEGIN

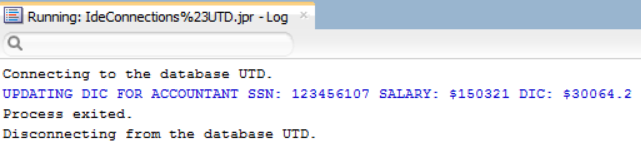
THIS\_SSN := 123456107;

CALCULATE\_DIC(

THIS\_SSN => THIS\_SSN

);

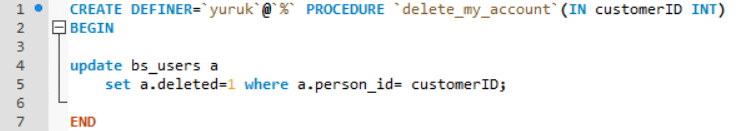
END;

* Output:  
  

Procedure-II Deleting a Customer profile

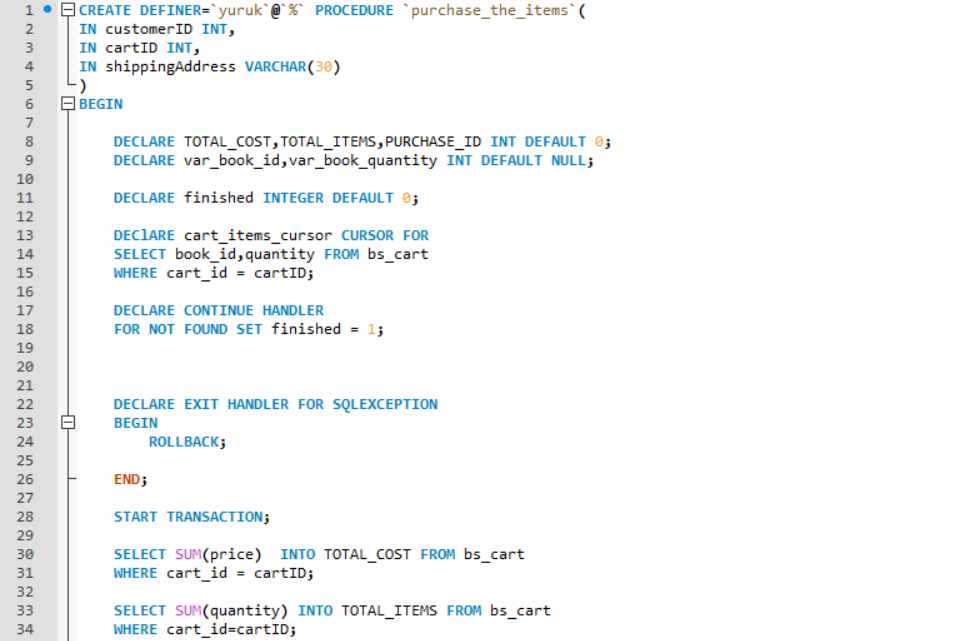
When A customer wants to quit the profile on the Online Book store, this procedure is called.

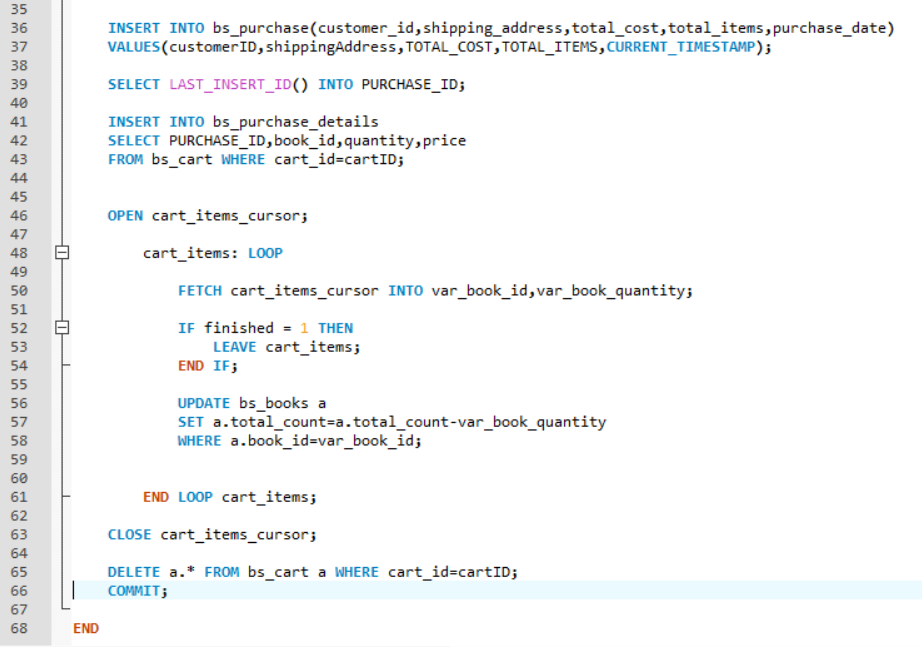
* Args : (Apparatus ID IN, Quantity OUT)



## Procedure-III Purchase the items from cart

Print the details of those incidences which are yet to be contained such that the estimated containment time takes 10 days or more. Also display the name and the department number of the department handling those incidences along with the impact region and type of primary equipment it is utilizing.





Arg: None

* Test Case SQL:

BEGIN

REPORTING\_ANALYSIS();

--rollback;

END;

* SQL Output:

Connecting to the database UTD.

INCIDENCE : INC00001 OF TYPE Wildfire, MANAGED BY DEPT: (FR001) FIRE; ORIGINATED ON: 23-NOV-16& EST CONTAINMENT BY: 03-DEC-16. IMPACT: 4347 Acres, %-CONTAINED: 95% USING PRIMARY EQIP OF TYPE :Water and Foam

INCIDENCE : INC00005 OF TYPE Industrial Fire, MANAGED BY DEPT: (FR001) FIRE; ORIGINATED ON: 27-NOV-16& EST CONTAINMENT BY: 07-DEC-16. IMPACT: 1523 Acres, %-CONTAINED: 95% USING PRIMARY EQIP OF TYPE :Dry Powder

INCIDENCE : INC00010 OF TYPE Wildfire, MANAGED BY DEPT: (FR001) FIRE; ORIGINATED ON: 16-NOV-16& EST CONTAINMENT BY: 10-DEC-16. IMPACT: 158 Acres, %-CONTAINED: 20% USING PRIMARY EQIP OF TYPE :Water and Foam

INCIDENCE : INC00011 OF TYPE Wildfire, MANAGED BY DEPT: (FR001) FIRE; ORIGINATED ON: 14-NOV-16& EST CONTAINMENT BY: 10-DEC-16. IMPACT: 529 Acres, %-CONTAINED: 36% USING PRIMARY EQIP OF TYPE :Water and Foam

INCIDENCE : INC00012 OF TYPE Homefire, MANAGED BY DEPT: (FR001) FIRE; ORIGINATED ON: 23-NOV-16& EST CONTAINMENT BY: 10-DEC-16. IMPACT: 30 Acres, %-CONTAINED: 97% USING PRIMARY EQIP OF TYPE :Dry Powder

INCIDENCE : INC00013 OF TYPE Wildfire, MANAGED BY DEPT: (FR001) FIRE; ORIGINATED ON: 10-NOV-16& EST CONTAINMENT BY: 10-DEC-16. IMPACT: 851 Acres, %-CONTAINED: 99% USING PRIMARY EQIP OF TYPE :Water and Foam

INCIDENCE : INC00014 OF TYPE Wildfire, MANAGED BY DEPT: (FR001) FIRE; ORIGINATED ON: 21-NOV-16& EST CONTAINMENT BY: 10-DEC-16. IMPACT: 654 Acres, %-CONTAINED: 95% USING PRIMARY EQIP OF TYPE :Water and Foam

Process exited.

Disconnecting from the database UTD.

|  |
| --- |
| CUSTOMER |
| Customer ID |
| First Name |
| Last name |
| Phone |
| Email ID |
| Password |
| City |
| State |
| Zip |
| Country |

|  |
| --- |
| AUTHOR |
| Author ID |
| First Name |
| Last Name |
| Email ID |
| City |
| State |
| Country |

PURCHASES

HAS

|  |
| --- |
| BOOKS |
| Book ID |
| Title |
| Author |
| Publisher |
| Published Date |
| Price |
| Category |
| Image ID |
| Version |
| Total Count |
| Available Count |

|  |
| --- |
| CATEGORY |
| Category ID |
| Name |
| Popularity |
| Total Count |

HAS

|  |
| --- |
| USER |
| User ID |
| Password |
| Person ID |
| Last login |

|  |
| --- |
| CART |
| Cart ID |
| Book ID |
| Quantity |
| Price |
| Product Add Time |

|  |
| --- |
| ROLE |
| Role ID |
| Name |

HAS

CONTAINS

HAS