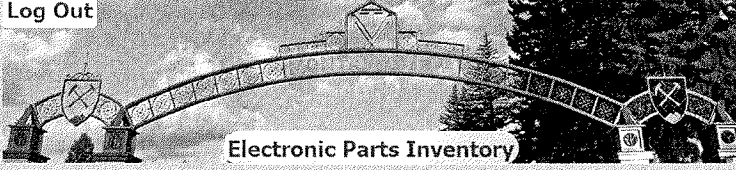


Creating web sites on the CS se... Electronics Parts Inventory

htaccess password

Log Out



Electronic Parts Inventory

Return Part (Mitchell Deplazes)

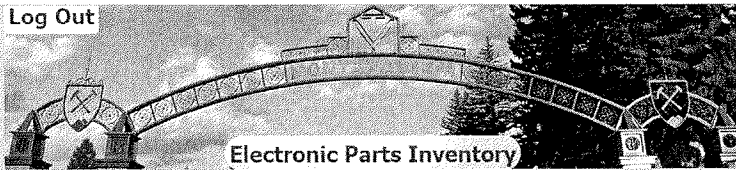
Scan Rental Barcode:

12:34 PM 5/4/2015

Creating web sites on the CS se... Electronics Parts Inventory

htaccess password

Log Out



Electronic Parts Inventory

Purchase Part (Mitchell Deplazes)

Scan Catalog Barcode:

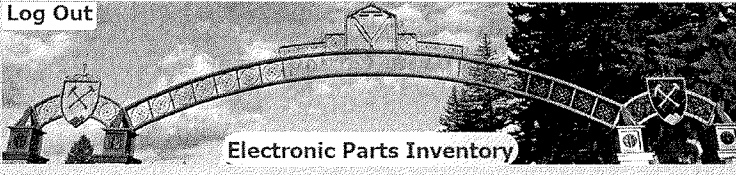
Quantity to Purchase:

12:34 PM 5/4/2015

Creating web sites on the CS se... Electronics Parts Inventory x +

http://katie.mtech.edu/~mpdeplazes/ElectronicParts/scanPart_reservePart.php htaccess password

Log Out



Electronic Parts Inventory

Reserve Part (Mitchell Deplazes)

Scan Rental Barcode:
reservationStart (yyyy/mm/dd): _____
reservationEnd (yyyy/mm/dd): _____
Description: _____

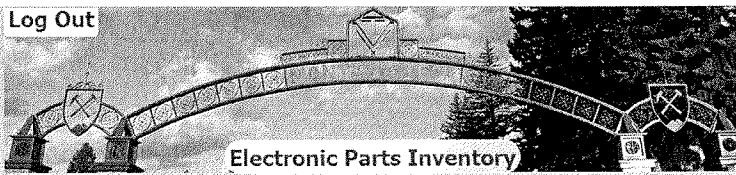
Reserve Part

12:34 PM 9/4/2015

Creating web sites on the CS se... Electronics Parts Inventory x +

http://katie.mtech.edu/~mpdeplazes/ElectronicParts/findPart_searchPart.php htaccess password

Log Out



Electronic Parts Inventory

Search for Part

Category Name: All Filter

PartCatalogName: _____

Size: _____

Footprint: _____

Max Price: _____

Minimum Price: _____

Quantity Available: _____

Filter

12:34 PM 9/4/2015

```
// NOTE: The PHP only shows columns beginning with capital letters. That is why everything has an AS.
```

```
// Queries for creating the tables
```

```
    OP TABLE Purchase;
    DROP TABLE Rental;
    DROP TABLE Reservation;
    DROP TABLE User;
    DROP TABLE RentalPart;
    DROP TABLE PartInfo;
    DROP TABLE PartCatalog;
    DROP TABLE Subcategory;
    DROP TABLE Category;
```

```
CREATE TABLE Category
```

```
(
    categoryId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(40) UNIQUE NOT NULL
);
```

```
CREATE TABLE Subcategory
```

```
(
    subcategoryId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(40) NOT NULL,

    categoryId INT NOT NULL,
    FOREIGN KEY (categoryId) REFERENCES Category(categoryId)
```

```
CREATE TABLE PartCatalog
```

```
(
    catalogId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR (40) NOT NULL,
    size VARCHAR (20),
    footprint VARCHAR (20),
    description VARCHAR (400),
    price NUMERIC (10,2) NOT NULL DEFAULT 0,
    quantity INT NOT NULL DEFAULT 0,
    location VARCHAR (40),
    minimumQuantity INT DEFAULT 0,
    rentalDuration INT,

    subcategoryId INT NOT NULL,
    FOREIGN KEY (subcategoryId) REFERENCES Subcategory(subcategoryId)
);
```

```
CREATE TABLE PartInfo
```

```
(
    infoId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    brand VARCHAR (40),
    modelNo VARCHAR (40),
    datasheetUrl VARCHAR (200),
    orderable BOOL NOT NULL DEFAULT TRUE,
```

```
catalogId INT NOT NULL,  
FOREIGN KEY (catalogId) REFERENCES PartCatalog(catalogId)
```

```
CREATE TABLE RentalPart
```

```
(  
    rentalPartId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    description VARCHAR (400),  
    location VARCHAR (40),  
  
    infoId INT NOT NULL,  
    FOREIGN KEY (infoId) REFERENCES PartInfo(infoId)  
);
```

```
CREATE TABLE User
```

```
(  
    userId VARCHAR (12) NOT NULL PRIMARY KEY,  
    firstName VARCHAR (40),  
    lastName VARCHAR (40),  
    email VARCHAR (200) NOT NULL,  
    balance NUMERIC (10,2),  
    hold BOOL NOT NULL DEFAULT FALSE,  
    isStaff BOOL NOT NULL DEFAULT FALSE  
    dateCreated datetime NOT NULL  
);
```

```
CREATE TABLE Purchase
```

```
(  
    purchaseId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    date DATETIME NOT NULL,  
    amountPaid NUMERIC (10,2) NOT NULL,  
    quantityPurchased INT NOT NULL DEFAULT 1,  
    paymentReceived BOOL NOT NULL,  
  
    catalogId INT NOT NULL,  
    FOREIGN KEY (catalogId) REFERENCES PartCatalog(catalogId),  
  
    userId VARCHAR (12) NOT NULL,  
    FOREIGN KEY (userId) REFERENCES User(userId)  
);
```

```
CREATE TABLE Rental
```

```
(  
    rentalId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    dateRented DATETIME NOT NULL,  
    dateDue DATETIME NOT NULL,  
    dateReturned DATETIME,  
    amountPaid NUMERIC (10,2) NOT NULL,  
    paymentReceived BOOL NOT NULL,  
    description VARCHAR (400),
```

```
rentalPartId INT NOT NULL,
FOREIGN KEY (rentalPartId) REFERENCES RentalPart (rentalPartId),

userId VARCHAR (12) NOT NULL,
FOREIGN KEY (userId) REFERENCES User(userId)
);

CREATE TABLE Reservation
(
    reservationId INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    reservationStart DATETIME NOT NULL,
    reservationEnd DATETIME NOT NULL,
    dateReturned DATETIME,
    description VARCHAR (400),

    rentalPartId INT NOT NULL,
    FOREIGN KEY (rentalPartId) REFERENCES RentalPart (rentalPartId),

    userId VARCHAR (12) NOT NULL,
    FOREIGN KEY (userId) REFERENCES User(userId)
);

CREATE VIEW DisplayPartCatalog
AS (
SELECT Category.name AS "Category", Subcategory.name AS "Subcategory", PartCatalog.name AS
"PartCatalogName", size AS "Size", footprint AS "Footprint", description AS Description,
    ice AS "Price", quantity AS "QTY", minimumQuantity AS "MinQTY", location AS Location,
PartCatalog.catalogId AS CatalogID, CatalogBarcode
FROM PartCatalog
JOIN Subcategory ON PartCatalog.subcategoryId = Subcategory.subcategoryId
JOIN Category ON Subcategory.categoryId = Category.categoryId
JOIN CatalogBarcode ON PartCatalog.catalogId = CatalogBarcode.catalogId
);

CREATE VIEW DisplayRentalPart
AS (
SELECT description AS Description, location AS Location, RentalPart.rentalpartId AS
RentalPartID, RentalBarcode, infoId
FROM RentalPart
JOIN RentalBarcode ON RentalPart.rentalPartId = RentalBarcode.rentalPartId
);

// View drops. May need to be reordered.

DROP VIEW InventoryReport;
DROP VIEW UnderQuantityReport;
DROP VIEW PurchaseHistory;
DROP VIEW RentalHistory;
DROP VIEW CurrentRentals;
DROP VIEW OverdueRentals;

DROP VIEW DisplayPartCatalog;
```

```
DROP VIEW DisplayPartInfo;
DROP VIEW DisplayRentalPart;

DROP VIEW OldPurchase;
DROP VIEW OldRental;
DROP VIEW OldReservation;
DROP VIEW OldInactiveUser;

DROP VIEW ReservationHistory;
DROP VIEW CurrentReservations;
DROP VIEW OverdueReservations;
DROP VIEW DisplayUser;
DROP VIEW CatalogBarcode;
DROP VIEW RentalBarcode;

// Gets information on the current part catalog inventory
CREATE VIEW InventoryReport
AS (
SELECT Category.name AS "Category", Subcategory.name AS "Subcategory", PartCatalog.name AS
"PartCatalogName", PartCatalog.catalogId AS CatalogID, size AS "Size", footprint AS
"Footprint", price AS "Price", quantity AS "QTY", minimumQuantity AS "MinQTY", location AS
"Location"
FROM PartCatalog
JOIN Subcategory ON PartCatalog.subcategoryId = Subcategory.subcategoryId
JOIN Category ON Subcategory.categoryId = Category.categoryId
);

// Reports inventory that is under quantity
CREATE VIEW UnderQuantityReport
AS (
SELECT * FROM InventoryReport
WHERE QTY < MinQTY
);

// All purchases in the database
CREATE VIEW PurchaseHistory
AS (
SELECT firstName AS "First", lastName AS "Last", name AS "PartName", date AS
"PurchaseDate", amountPaid AS "AmountPaid", quantityPurchased AS "QTYPurchased",
paymentReceived AS "PaymentReceived", Purchase.purchaseId AS PurchaseID
FROM Purchase
JOIN User ON Purchase.userId = User.userId
JOIN PartCatalog ON Purchase.catalogId = PartCatalog.catalogId
);

// All rentals in the database
CREATE VIEW RentalHistory
AS (
SELECT firstName AS "First", lastName AS "Last", PartCatalog.name AS "PartCatalogName",
PartInfo.brand AS "Brand", dateRented AS "DateRented", dateDue AS "DueDate", dateReturned
AS "DateReturned", amountPaid AS "AmountPaid", Rental.description AS "RentalDescription",
paymentReceived AS "PaymentReceived", Rental.rentalId AS RentalID, RentalPart.rentalPartId
```

```

AS RentalPartId
FROM Rental
JOIN User ON Rental.userId = User.userId
    IN RentalPart ON Rental.rentalPartId = RentalPart.rentalPartId
JOIN PartInfo ON RentalPart.infoId = PartInfo.infoId
JOIN PartCatalog ON PartInfo.catalogId = PartCatalog.catalogId
);

// All rentals currently active
CREATE VIEW CurrentRentals
AS (
SELECT * FROM RentalHistory
WHERE DateReturned IS NULL
);

// All rentals that are overdue
CREATE VIEW OverdueRentals
AS (
SELECT * FROM CurrentRentals
WHERE DueDate < NOW()
);

// Nicely displays part catalog info
CREATE VIEW DisplayPartCatalog
AS (
SELECT Category.name AS "Category", Subcategory.name AS "Subcategory", PartCatalog.name AS
    artCatalogName", size AS "Size", footprint AS "Footprint", price AS "Price", quantity AS
"QTY", minimumQuantity AS "MinQTY", location AS "Location", description AS Description,
PartCatalog.catalogId AS CatalogID, CatalogBarcode AS CatalogBarcode
FROM PartCatalog
JOIN Subcategory ON PartCatalog.subcategoryId = Subcategory.subcategoryId
JOIN Category ON Subcategory.categoryId = Category.categoryId
JOIN CatalogBarcode ON PartCatalog.catalogId = CatalogBarcode.CatalogID
);

// Nicely displays part info info
CREATE VIEW DisplayPartInfo
AS (
SELECT brand AS Brand, modelNo AS ModelNo, datasheetUrl AS DatasheetURL, orderable AS
Orderable, catalogId, infoId AS InfoID
FROM PartInfo
);

// Nicely displays rental part info
CREATE VIEW DisplayRentalPart
AS (
SELECT description AS Description, location AS Location, RentalPart.rentalPartId AS
RentalPartID, RentalBarcode, NOT EXISTS (SELECT * FROM CurrentRentals WHERE
    rrentRentals.RentalPartId = RentalPart.rentalPartId) AND NOT EXISTS (SELECT * FROM
CurrentReservations WHERE CurrentReservations.RentalPartId = RentalPart.rentalPartId) AS
Available
FROM RentalPart

```

```
JOIN RentalBarcode ON RentalPart.rentalPartId = RentalBarcode.RentalPartID
```

```
);
```

```
All reservations in database
```

```
CREATE VIEW ReservationHistory
```

```
AS (
SELECT firstName AS "First", lastName AS "Last", PartCatalog.name AS "PartCatalogName",
PartInfo.brand AS "Brand", PartCatalog.catalogId AS CatalogID, reservationStart AS
"ReservationStart", reservationEnd AS "ReservationEnd", dateReturned AS "DateReturned",
Reservation.description AS "RentalDescription", Reservation.reservationId AS ReservationID,
Reservation.rentalPartId AS RentalPartID
FROM Reservation
JOIN User ON Reservation.userId = User.userId
JOIN RentalPart ON Reservation.rentalPartId = RentalPart.rentalPartId
JOIN PartInfo ON RentalPart.infoId = PartInfo.infoId
JOIN PartCatalog ON PartInfo.catalogId = PartCatalog.catalogId
);
```

```
// Reservations currently active
```

```
CREATE VIEW CurrentReservations
```

```
AS (
SELECT * FROM ReservationHistory
WHERE DateReturned IS NULL
);
```

```
// Reservations
```

```
CREATE VIEW OverdueReservations
```

```
AS (
SELECT * FROM CurrentReservations
WHERE ReservationEnd < NOW()
);
```

```
// Nicely displays users
```

```
CREATE VIEW DisplayUser
```

```
AS (
SELECT userId AS UserID, firstName AS FirstName, lastName AS LastName, email AS Email,
balance AS Balance, hold AS Hold, isStaff AS Staff, dateCreated AS DateCreated
FROM User
);
```

```
// Creates barcodes for part catalog entries
```

```
CREATE VIEW CatalogBarcode
```

```
AS (
SELECT catalogId, CONCAT(RPAD(name, 4, '-'), RPAD(size, 4, '-'), RPAD(footprint, 4, '-'),
0, catalogId) AS CatalogBarcode
FROM PartCatalog
);
```

```
' Creates barcodes for rental entries
```

```
CREATE VIEW RentalBarcode
```

```
AS (
SELECT rentalPartId, CONCAT(RPAD(CatalogBarcode, 12, '-'), 1, rentalPartId) AS RentalBarcode
```



```
FROM RentalPart rp
JOIN PartInfo pi ON rp.infoId = pi.infoId
JOIN CatalogBarcode cb ON pi.catalogId = cb.catalogId

// Can delete all entries matching IDs from these view to clear old data
// All purchases older than 2 years
CREATE VIEW OldPurchase
AS (
SELECT purchaseId FROM Purchase
WHERE date < DATE_ADD(NOW(), INTERVAL -2 YEAR)
);

// All rentals older than 2 years
CREATE VIEW OldRental
AS (
SELECT rentalId FROM Rental
WHERE dateReturned < DATE_ADD(NOW(), INTERVAL -2 YEAR)
);

// All reservations older than 2 years
CREATE VIEW OldReservation
AS (
SELECT reservationId FROM Reservation
WHERE dateReturned < DATE_ADD(NOW(), INTERVAL -2 YEAR)
);

// All users without activity in the last 2 years
CREATE VIEW OldInactiveUser
AS (
SELECT User.userId FROM User
WHERE dateCreated < DATE_ADD(NOW(), INTERVAL -2 YEAR)
AND NOT EXISTS (SELECT * FROM Purchase WHERE User.userId = Purchase.userId)
AND NOT EXISTS (SELECT * FROM Rental WHERE User.userId = Rental.userId)
AND NOT EXISTS (SELECT * FROM Reservation WHERE User.userId = Reservation.userId)
);
```

Past
materials