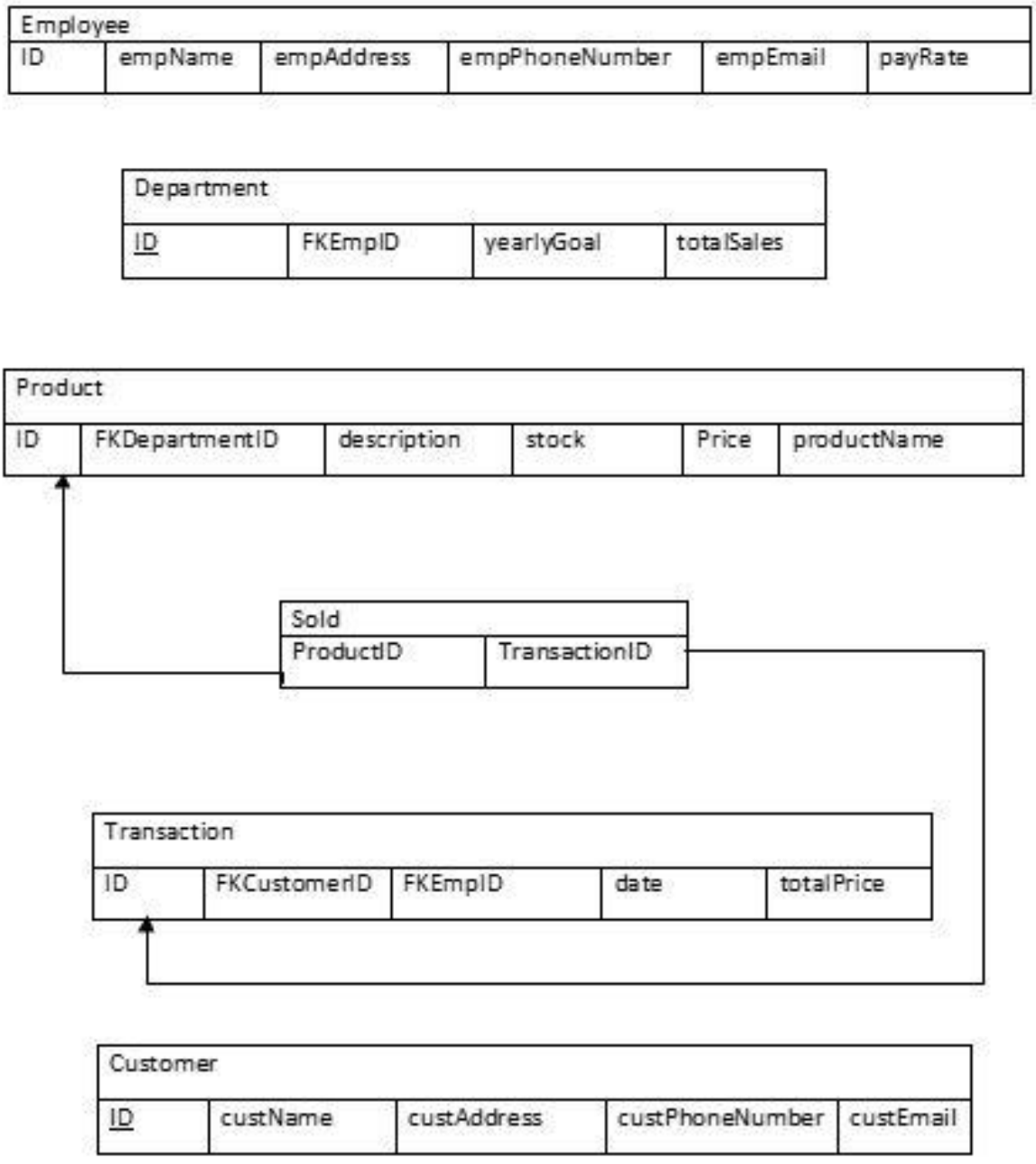
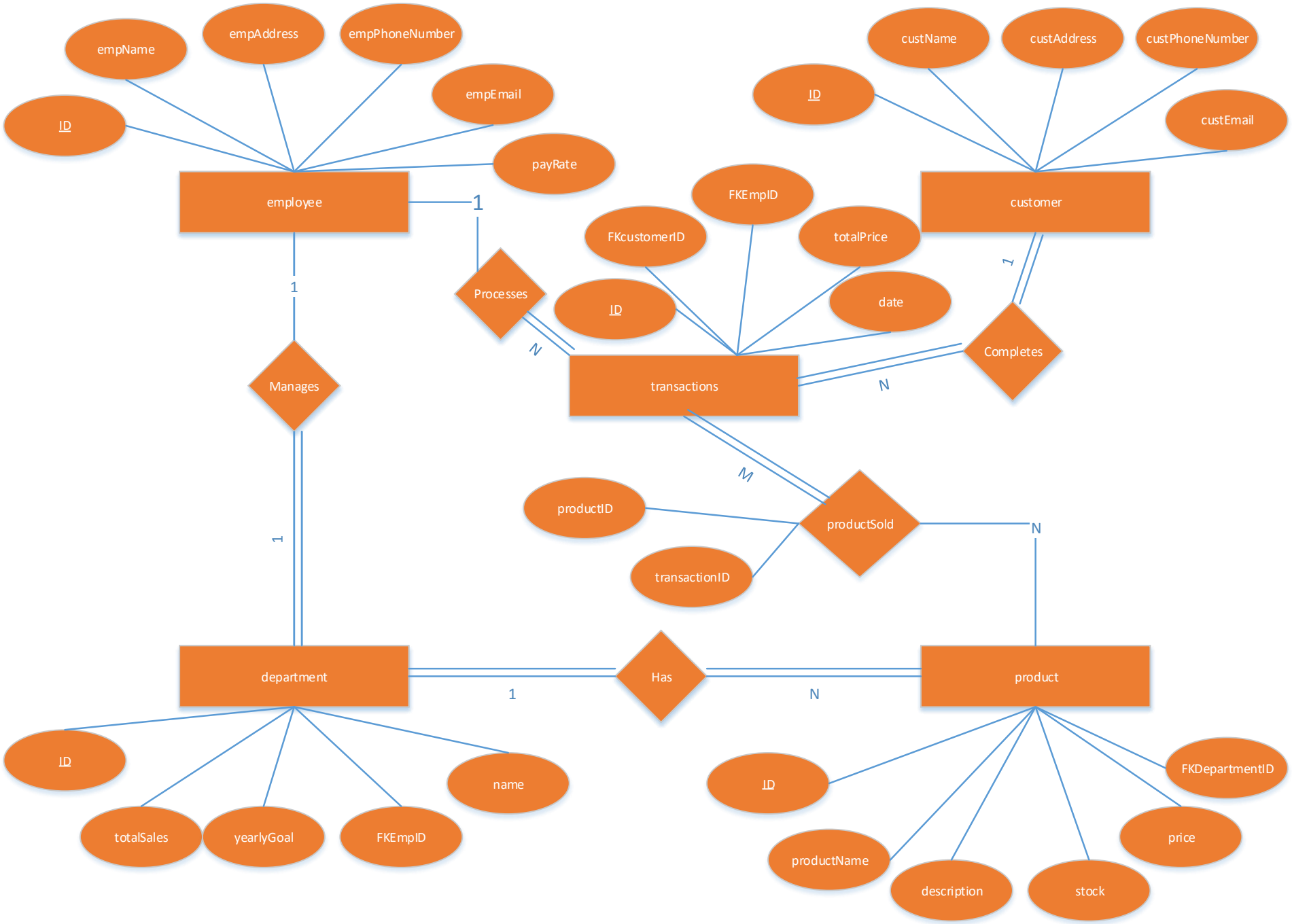
CS 340 - Final Project Schema Jacob Leno



Jacob Leno CS 340 Databases Final Project

CS 340 - Final Project ER Diagram Jacob Leno



CS 340 Database Project Proposal

HARDWARE STORE DATABASE

I will be making a database that stores information for a hardware store such as Lowes or Home Depot. There will be lots of complexity because a hardware store consists of employees, departments, customers, inventory and transactions. I am switching from my original idea in the first project proposal I submitted because I think this is a more fitting idea for a database.

The entities I will have in my database are:

- Employees (Name, address, phone number, email, pay rate)
- Departments (Total sales, yearly goal)
- Customer (Name, address, phone, email)
- Products (Product name, description, quantity, price, stock)
- Transactions (Total price, date, customerID)

The relationships I will have are:

- Manages (Employee to Department)
- Processes (Employee to Transaction)
- Has (Department to Products)
- Completes (Customer to Transaction)
- Sold (Product to Transaction)

Final Project Outline

This project is a fake database for hardware/home improvement store. It will store information about employees, customers, departments, transactions, and products in the store. There will be one many to many relationship between products and transactions and the rest of the entities in the table will be joined through foreign keys.

Employees will be able to use this database to look up transactions for every customer that has purchased something from the store. Each item purchased through a transaction will be recorded along with the transaction. A record will also be kept of which employees manage which departments and which products are available in which departments.

Database Outline

The database will have five entities: Employee, Transaction, Customer, Department, and Products. There will also be one table called Sold for the many to many relationship between Product and Transaction.

Attributes

The **Employee** entity will have six attributes: ID (a primary key), empName, empAddress, empPhoneNumber, empEmail, and payrate.

The **Department** entity will have 4 attributes: ID (a primary key), totalSales, yearlyGoal and FKEmpID.

The **Product** entity will have 6 attributes: ID (a primary key), productName, description, stock, price, and FKDepartmentID.

The **Transaction** entity will have 5 attributes: ID (a primary key), FKCustomerID, FKEmpID, totalPrice, and date.

The **Customer** entity will have 5 attributes: ID (a primary key), custName, custAddress, custPhoneNumber, and custEmail.

The **Sold** relationship will have 2 attributes: ProductID and TransactionID

Relations:

Employee will have partial participation in its relation to Department because an employee does not need to manage a department. Department will have total participation in its relation to Employee because a department needs to be managed by at least one employee. The cardinality between Employee and Department and Department to Employee will be 1 to 1 because only one employee may manage a department and a department may be managed by only one employee.

Department will have total participation in its relation to Product because a department must have at least one product (these are not departments like marketing, service, returns but departments like paint, lumber, tools). Product will have a total participation in its relation to Department because a product must belong to a Department. The cardinality of Department to Product will be *N* because a department may have many products. The cardinality of Product to Department will be *1* because a product may have only one department.

Product will have partial participation with Transaction because a product does not need to be a part of a transaction. Transaction will have total participation with Product because a product must have been sold in a transaction. The cardinality of Product to Transaction will be *M* because an individual product can be a part of many transactions. The cardinality of Transaction to Product will be *N* because a transaction can include many different kinds of products.

Transaction will have total participation with Employee because a transaction must have been processed by an employee. Employee will have partial participation with Transaction because an employee does not have to process any transactions (Cashiers will likely be the only ones processing transactions, not clerks, managers, salesmen etc). The cardinality of Employee to Transaction will be *N* because an employee may process many transactions. The cardinality of Transaction to Employee will be *1* because a transaction may only be processed by one employee.

Customer will have total participation with Transaction because in order for someone to be a customer the must have made a purchase. Transaction will have total participation with customer because in order for there to have been a transaction there must have been a customer. The cardinality of Customer to Transaction will be *N* because a customer may have many transactions. The cardinality of Transaction to Customer will be *1* because a transaction can only belong to one customer.

Each table will have the ability to add to it. The transactions table will have the ability to filter transactions by customer name to fulfill the search or filter requirement. The transaction, customer, and product table will have the ability to have their entries deleted to fulfill the delete requirement. The products, customer, employees, and department tables will have the ability to be updated to fulfill the update requirement. Finally, the transaction table will have the ability to have items added and removed, adding and removing things from this table will also add and remove them from the many to many relationship 'productSold' thus fulfilling the many to many add/remove requirement.

Data Definition Queries

```
DROP TABLE IF EXISTS `customer`;

CREATE TABLE `customer`(
        `id` INT(11) PRIMARY KEY AUTO_INCREMENT,
        `custName` varchar(80) NOT NULL,
        `custAddress` varchar(80) DEFAULT NULL,
        `custPhoneNumber` varchar(80) DEFAULT NULL,
        `custEmail` varchar(80) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

insert  into `customer`(`id`, `custName`,`custAddress`,`custPhoneNumber`, `custEmail`) values

(1, 'Thomas Shealtiel', '123 6th St.', '(555) 605-9430', ''),
(2, 'Sally Dorthy', '4 Goldfield Rd.', '(555) 168-2717', 'webdragon@comcast.net'),
(3, 'Wanda Walsh', '71 Pilgrim Avenue', '(555) 384-4631', 'wwalsh@icloud.com'),
(4, 'Edgar Beck', '44 Shirley Ave. ', '(555) 587-7519', ''),
(5, 'Cody Shaw ', '70 Bowman St.', '(555) 569-0082', ''),
(6, 'Rosalie Norton', '', '', ''),
(7, 'Velma Morton', '514 S. Magnolia St.', '(555) 913-7478', 'vmorton@gmail.com'),
(8, 'Peggy Boone', '502 Ketch Harbour Drive ', '(555) 316-0103', ''),
```

```
(9, 'Mike Gross ', '957 Willow Street ', '(555) 582-7877', ''),
(10, 'Dwayne Wilkins', '676 53rd Road ', '(555) 302-4238', 'dwilkins@outlook.com'),
(11, 'Sonia Rose ', '72 E. Halifax Street ', '(555) 479-4012', ''),
(12, 'Chris Nguyen', '8003 Bear Hill Lane', '(555) 121-4593', ''),
(13, 'Kristina Quinn', '187 South Cambridge Lane', '(555) 631-8505', ''),
(14, 'Freddie Parker', '9804 Adams Lane', '(555) 618-1561', ''),
(15, 'Toby Frazier', '768 Pierce St. ', '(555) 347-3421', '');

DROP TABLE IF EXISTS `employee`;

CREATE TABLE `employee`(
        `id` INT(11) PRIMARY KEY AUTO_INCREMENT,
        `empName` varchar(80) NOT NULL,
        `empAddress` varchar(80) NOT NULL,
        `empPhoneNumber` varchar(80) NOT NULL,
        `empEmail` varchar(80) DEFAULT NULL,
        `hourlyRate` decimal(10,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

insert  into `employee`(`id`, `empName`,`empAddress`,`empPhoneNumber`, `empEmail`, `hourlyRate`) values

(1, 'Nathan Mcilvaine', '670 Church Street', '(555) 588-3240', 'nmcilvaine@outlook.com', 12.5),
(2, 'Keena Connally', '273 Golden Star Circle', '(555) 809-6223', '', 12.5),
(3, 'Douglas Douglas', '225 Tower Lane', '(555) 205-8102', 'dougdoug@gmail.com', 18.5),
(4, 'Jana Massey', '661 Trout Dr.', '(555) 601-2172', '', 20),
(5, 'Carrie Willis', '446 Riverside St. ', '(555) 643-6499', '', 18.5),
(6, 'Alfred Price', '242 Hill Field Road', '(555) 865-3391', '', 30),
(7, 'Jean Luna', '445 Homewood Street ', '(555) 373-0856', '', 12.5),
(8, 'Nichole Joseph', '94 Queen St.', '(555) 377-7958', '', 30),
(9, 'Jeff Oliver', '471 West Hartford St.', '(555) 408-5497', 'joliver@icloud.com', 30),
(10, 'Miguel Burton', '72 Hillside Court', '(555) 656-5753', '', 20),
(11, 'Wilfred Osborne', '7933 3rd St.', '(555) 217-4267', '', 22.5),
(12, 'Andres Cummings', '7623 Carriage Road', '(555) 471-7651', '', 40),
(13, 'Michael Jones', '8861 South Union Ave.', '(555) 419-1721', 'mjones@gmail.com', 20.5),
(14, 'Leo Boyd', '638 Swanson St.', '(555) 128-1533', '', 17.5),
(15, 'Alicia Cook', '9248 North 6th St.', '(555) 361-5295', '', 15.5);

DROP TABLE IF EXISTS `transactions`;

CREATE TABLE `transactions`(
        `id` INT(11) PRIMARY KEY NOT NULL AUTO_INCREMENT,
        `fkCustomerID` INT(11) NOT NULL,
        `fkEmpID` INT(11) NOT NULL,
        `totalPrice` decimal(10,2) NOT NULL,
        `transactionDate` date DEFAULT NULL,
        FOREIGN KEY (`fkCustomerID`) REFERENCES `customer` (`id`),
        FOREIGN KEY (`fkEmpID`) REFERENCES `employee` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
insert  into `transactions`(`id`, `fkCustomerID`, `fkEmpID`, `totalPrice`, `transactionDate`) values
(1, 1, 1, 121.69, '2014-05-29'),
(2, 1, 2, 289.99, '2014-07-02'),
(3, 2, 2, 549.00, '2014-09-17'),
(4, 3, 5, 278.99, '2014-11-17'),
(5, 4, 1, 390.44, '2015-01-29'),
(6, 5, 14, 20.00, '2015-08-06'),
(7, 6, 14, 75.15, '2015-08-14'),
(8, 7, 1, 4.95, '2015-12-07'),
(9, 8, 5, 4997.95, '2015-12-07'),
(10, 8, 2, 39.95, '2016-02-10'),
(11, 9, 2, 45.95, '2016-06-18'),
(12, 10, 5, 20.00, '2016-09-26'),
(13, 10, 1, 299.99, '2016-11-26'),
(14, 11, 14, 93.74, '2017-04-18'),
(15, 12, 14, 508.90, '2017-05-03'),
(16, 13, 14, 35.20, '2017-06-12'),
(17, 14, 5, 549.00, '2017-12-08'),
(18, 15, 2, 27.95, '2018-05-12'),
(19, 15, 1, 99.69, '2018-05-25'),
(20, 15, 1, 569.94, '2018-08-27');


DROP TABLE IF EXISTS `department`;

CREATE TABLE `department`(
        `id` INT(11) PRIMARY KEY AUTO_INCREMENT,
        `fkEmpID` INT(11) NOT NULL,
        `name` varchar(80) NOT NULL,
        `totalSales` decimal(10,2) DEFAULT NULL,
        `yearlyGoal` decimal(10,2) DEFAULT NULL,
        FOREIGN KEY (`fkEmpID`) REFERENCES `employee` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

insert  into `department`(`id`, `fkEmpID`, `name`, `totalSales`,`yearlyGoal`) values

(1, 12, 'Paint', 12723.80, 15000),
(2, 4,  'Electrical',23481.71, 20000),
(3, 6, 'Hardware', 18301.23, 30000),
(4, 8, 'Lawn and Garden', 10379.09, 15000),
(5, 9, 'Plumbing', 14038.51, 20000),
(6, 3, 'Tools', 35239.72, 40000);
```

```
DROP TABLE IF EXISTS `product`;

CREATE TABLE `product`(
    `id` INT(11) PRIMARY KEY AUTO_INCREMENT,
    `fkDepartmentID` INT(11) NOT NULL,
    `productName` varchar(80) NOT NULL,
    `description` varchar(255) DEFAULT NULL,
    `price` decimal(10,2) NOT NULL,
    `stock` INT(11) DEFAULT NULL,
    FOREIGN KEY (`fkDepartmentID`) REFERENCES `department` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

insert  into `product`(`id`, `fkDepartmentID`,`productName`,`description`, `price`, `stock`) values

(1, 1, 'Ultra Pure White Satin Enamel Exterior Paint', ' This exciting product, featuring Nanoguard technology, provides an extra-protective shell that guards against damage from sunlight, moisture, stains and dirt.', 35.20, 10),
(2, 1, 'Bright Cornflower Blue Satin Interior Paint with Primer', 'Protects and beautifies just about any room in your home, including basements and high-traffic areas.', 27.95, 5),
(3, 1, 'Epoxy Concrete and Garage Floor Paint', 'High-performance, ready-to-use, water-based floor paint that resists tire marks and marring', 39.95, 7),
(4, 2, '250 ft. 14/2 Gray Solid Wire', 'Used in applications that include outside lamp posts, pumps and other loads. It can also be used for outbuildings such as garages and barns.', 93.74, 10),
(5, 2, '60-Watt Equivalent Non-Dimmable LED Light Bulb', 'The bulb is designed for indoor use and features an A-line shape and medium base. ', 4.95, 5),
(6, 2, '4-in-1 Home Security Kit', 'HD camera, smoke detector, motion detector and 90 DB siren in 1 sensor, security app allows user to call police, dismiss alarm, sound siren and more', 299.99, 4),
(7, 3, '#9 x 3 in. Star Wood Deck Screws', 'A lifetime polymer coating provides extra corrosion resistance that is guaranteed for the life of your outdoor projec', 5.95, 16),
(8, 3, 'Satin Nickel Smart Lock with Alarm', 'Combines advanced features and compatibility with your home automation system, allowing you to control your home from anywhere and freeing you from the hassles of keys.', 249.99, 6),
(9, 3, '1/2 in. x 100 ft. Heavy Duty Rope', 'This general purpose rope is flexible and lightweight, rot proof, resists oil, water, gasoline and most chemicals.', 20.00, 10),
(10, 4, '50 Gal. Rain Barrel with Brass Spigot', 'Oak barrel texture is molded into each barrel and will not fade, rot or risk insect infestation.', 90.49, 3),
(11, 4, '2-Burner Propane Gas Grill', 'Two durable stainless steel burners and two stainless steel tables that fold down for easy maneuverability and storage.', 299.95, 4),
(12, 4, '5-Piece Padded Patio Dining Set', 'Durable with rust-resistant steel frames powder-coated in pewter, the set includes four swivel motion chairs with UV resistant material', 388.95, 2),
(13, 5, 'Heavy-Duty 100 Gal. 270,000 BTU Natural Gas Water Heater', 'Designed to fit into tight retrofit applications. This model requires a 120-Volt power source.', 4997.95, 3),
(14, 5, '6-Station Easy-Set Logic Indoor/Outdoor Sprinkler Timer', '6-Station Sprinkler Timer features Easy-Set programming logic and a large LCD and dial to make programming your watering system easy.', 39.95, 15),
(15, 5, '1 HP Shallow Well Jet Pump', '1 HP durable, cast iron shallow well jet pump packaged with a comprehensive installation manual to help anyone build a safe and long-lasting well pump system.', 278.99, 2),
(16, 6, '20-Volt Lithium-Ion Cordless Compact Drill/Driver', 'It has a slim handle with a contoured, ergonomic grip to provide excellent balance and superb control during use.', 119.95, 5),
```

```
(17, 6, '9-Drawer Mobile Workbench with Solid Wood Top', 'Designed to handle rough usage without compromising integrity.', 289.99, 2),
(18, 6, '13 Amp 10 in. Professional Cast Iron Table Saw', '13 Amp motor for speeds up to 3450 RPM. The cast iron tables milled and polished surface minimizes vibration and provides a flat, level surface for trouble-free cutting.', 549.00, 3);
DROP TABLE IF EXISTS `productSold`;

CREATE TABLE `productSold`(
    `productID` INT(11),
    `transactionID` INT(11),
    PRIMARY KEY (`productID`, `transactionID`),
    KEY `productID` (`productID`),
    KEY `transactionID` (`transactionID`),
    CONSTRAINT `productSoldfk1` FOREIGN KEY (`productID`) REFERENCES `product` (`id`),
    CONSTRAINT `productSoldfk2` FOREIGN KEY (`transactionID`) REFERENCES `transactions` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

insert  into `productSold`(`productID`,`transactionID`) values

(2 ,1),
(4 ,1),
(17,2),
(18 ,3),
(15 ,4),
(11 ,5),
(10 ,5),
(9 ,6),
(1 ,7),
(3 ,7),
(5 ,8),
(13 ,9),
(14 ,10),
(7 ,11),
(3 ,11),
(9 ,12),
(6 ,13),
(4 ,14),
(12 ,15),
(16 ,15),
(1 ,16),
(18 ,17),
(2 ,18),
(4 ,19),
(7 ,19),
(11 ,20),
(8 ,20),
(9 ,20);
```

Data Definition Queries

For the customer table:

SELECT id, custName, custAddress, custPhoneNumber, custEmail FROM customer;

SELECT id, custName, custAddress, custPhoneNumber, custEmail FROM customer WHERE id = [customerId];

INSERT INTO customer (custName, custAddress, custPhoneNumber, custEmail) VALUES [customerName, CustomerAddress, CustomerPhoneNumber, CustomerEmail];

UPDATE customer SET custName=[customerName], custAddress=[customerAddress], custPhoneNumber=[customerPhoneNumber], custEmail=[customerEmail] WHERE id=[customerId];

DELETE FROM customer WHERE id = [customerId];

SELECT id, custName FROM customer;

For the employee table:

SELECT id, empName, empAddress, empPhoneNumber, empEmail, hourlyRate FROM employee;

SELECT id, empName, empAddress, empPhoneNumber, empEmail, hourlyRate FROM employee WHERE id = [employeeId];

INSERT INTO employee (empName, empAddress, empPhoneNumber, empEmail, hourlyRate) VALUES [EmployeeName, EmployeeAddress, EmployeePhoneNumber, EmployeeEmail, HourlyRate];

UPDATE employee SET empName=[EmployeeName] empAddress=[EmployeeAddress], empPhoneNumber=[EmployeePhoneNumber], empEmail=[EmployeeEmail], hourlyRate=[HourlyRate] WHERE id=[EmployeeId];

SELECT id, empName FROM employee;

For the department table:

SELECT d.id, name, e.empName, totalSales, yearlyGoal FROM department d INNER JOIN employee e on e.id = d.fkEmpID;

SELECT id, empName FROM employee WHERE id NOT IN (SELECT e.id FROM department d INNER JOIN employee e on e.id = d.fkEmpID); //Get names of employees that haven't been assigned to a department

SELECT d.id, name, e.empName, totalSales, yearlyGoal FROM department d INNER JOIN employee e on e.id = d.fkEmpID WHERE d.id = [DepartmentId];

INSERT INTO department (fkEmpID, name, totalSales, yearlyGoal) VALUES [EmployeeId, DepartmentName, TotalSales, YearlyGoal];

UPDATE department SET fkEmpID=[EmployeeId], name=[DepartmentName], totalSales=[TotalSales], yearlyGoal=[YearlyGoal] WHERE id=[DepartmentId];

SELECT id, name as deptName FROM department;

For the product table:

SELECT p.id, d.name as deptName, productName, description, price, stock FROM product p INNER JOIN department d on d.id = p.fkDepartmentID;

SELECT p.id, d.name as deptName, productName, description, price, stock FROM product p INNER JOIN department d on d.id = p.fkDepartmentID WHERE t.id = [TransactionId];

INSERT INTO product (fkDepartmentID, productName, description, price, stock) VALUES [DepartmentId, ProductName, ProductDescription, Price, Stock];

UPDATE product SET fkDepartmentID=[DepartmentId], productName=[ProductName], description=[ProductDescription], price=[Price], stock=[Stock] WHERE id=[ProductId];

DELETE FROM product WHERE id = [ProductId];

SELECT id, productName FROM product;

For the transactions table:

//Get the product name, the transaction id, the customer name, the total price, and the transaction date from the result //of joining all three tables together

SELECT p.productName, t.id, e.empName, c.custName, totalPrice, transactionDate FROM transactions t INNER JOIN employee e on e.id = t.fkEmpId INNER JOIN customer c on c.id = t.fkCustomerID INNER JOIN productSold ps on ps.transactionID = t.id INNER JOIN product p on p.id = ps.productID ORDER BY t.id;

//Get the product name, the transaction id, the customer name, the total price, and the transaction date of one //transaction

SELECT p.productName, t.id, e.empName, c.custName, totalPrice, transactionDate FROM transactions t INNER JOIN employee e on e.id = t.fkEmpId INNER JOIN customer c on c.id = t.fkCustomerID INNER JOIN productSold ps on ps.transactionID = t.id INNER JOIN product p on p.id = ps.productID WHERE t.id = [TransactionId] ORDER BY t.id;

//Get the transaction information for one certain customer

SELECT p.productName, t.id, e.empName, c.custName, totalPrice, transactionDate FROM transactions t INNER JOIN employee e on e.id = t.fkEmpId INNER JOIN customer c on c.id = t.fkCustomerID INNER JOIN productSold ps on ps.transactionID = t.id INNER JOIN product p on p.id = ps.productID WHERE c.id = [CustomerId];

INSERT INTO transaction (fkCustomerID, fkEmpId, totalPrice, transactionDate) VALUES [CustomerId, EmployeeId, TotalPrice, TransactionDate];

DELETE FROM transactions WHERE id = [TransactionId];

For the productSold relationship table:

INSERT INTO productSold (productID, transactionID) VALUES [ProductId, TransactionId];