**Acréscimos à Base de Dados Relativos à Nota de Falta**

**Novas Tabelas**

|  |  |  |
| --- | --- | --- |
| Tabelas | | |
| **Ordem** | **Tabela** | **Script** |
| 1 | nota\_faltas | DROP TABLE IF EXISTS `nota\_faltas`;  CREATE TABLE IF NOT EXISTS `nota\_faltas` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `data` timestamp NULL DEFAULT CURRENT\_TIMESTAMP,  `user\_id` int(11) NOT NULL,  `cliente\_id` int(11) NOT NULL,  `valor\_total` decimal(8,2) DEFAULT NULL,  `iva` decimal(8,2) NOT NULL,  `valor\_iva` decimal(8,2) NOT NULL,  `motivo\_iva\_id` int(11) DEFAULT NULL,  `aplicacao\_motivo\_iva` tinyint(1) NOT NULL,  `saida\_id` int(11) NOT NULL,  PRIMARY KEY (`id`),  UNIQUE KEY `saida\_id\_UNIQUE` (`saida\_id`),  KEY `fk\_notas\_falta\_clientes1\_idx` (`cliente\_id`),  KEY `fk\_notas\_falta\_user1\_idx` (`user\_id`),  KEY `fk\_notas\_falta\_motivo\_ivas1\_idx` (`motivo\_iva\_id`),  KEY `fk\_nota\_faltas\_saidas1\_idx` (`saida\_id`),  CONSTRAINT `fk\_nota\_faltas\_clientes1` FOREIGN KEY (`cliente\_id`) REFERENCES `clientes` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,  CONSTRAINT `fk\_nota\_faltas\_motivo\_ivas1` FOREIGN KEY (`motivo\_iva\_id`) REFERENCES `motivo\_ivas` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,  CONSTRAINT `fk\_nota\_faltas\_saidas1` FOREIGN KEY (`saida\_id`) REFERENCES `saidas` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION,  CONSTRAINT `fk\_nota\_faltas\_users1` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=18 DEFAULT CHARSET=latin1 COMMENT='Unique para a saida\_id porque se considera um relacionamento de um para um, desta forma, a saida deve aparecer uma unica vez para cada nota\_falta.' |
| 2 | iten\_nota\_faltas | DROP TABLE IF EXISTS `iten\_nota\_faltas`;  CREATE TABLE IF NOT EXISTS `iten\_nota\_faltas`  (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `produto\_id` int(11) NOT NULL,  `quantidade` int(11) NOT NULL,  `valor` decimal(8,2) NOT NULL,  `desconto` int(11) NOT NULL,  `subtotal` decimal(8,2) NOT NULL,  `nota\_falta\_id` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_iten\_nota\_faltas\_nota\_faltas1\_idx` (`nota\_falta\_id`),  CONSTRAINT `fk\_iten\_nota\_faltas\_nota\_faltas1` FOREIGN KEY (`nota\_falta\_id`) REFERENCES `nota\_faltas` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=38 DEFAULT CHARSET=latin1 |

**Novos Procedimentos**

|  |  |  |
| --- | --- | --- |
| **Novos Procedimentos** | | |
| **Ordem** | **Nome do Procedimento** | **Script** |
| 1 | SP\_decrementar\_valor\_total\_nota\_falta | DELIMITER $$  DROP PROCEDURE IF EXISTS `SP\_decrementar\_valor\_total\_nota\_falta`$$  CREATE PROCEDURE `SP\_decrementar\_valor\_total\_nota\_falta`(`valor` DECIMAL(8,2), `nota\_falta\_id` INT)  BEGIN  DECLARE valor\_iva DECIMAL(8,2);  DECLARE iva DECIMAL(8,2);  SET @valor\_iva = (valor+(valor\*17)/100);  SET @iva = ((valor\*17)/100);  UPDATE nota\_faltas SET nota\_faltas.valor\_total = (nota\_faltas.valor\_total - valor),  nota\_faltas.iva = (nota\_faltas.iva - @iva),  nota\_faltas.valor\_iva = (nota\_faltas.valor\_iva - @valor\_iva) WHERE nota\_faltas.id = nota\_falta\_id;  END$$  DELIMITER ; |
| 2 | SP\_incrementar\_valor\_total\_nota\_falta | DELIMITER $$  DROP PROCEDURE IF EXISTS `SP\_incrementar\_valor\_total\_nota\_falta`$$  CREATE PROCEDURE `SP\_incrementar\_valor\_total\_nota\_falta`(`valor` DECIMAL(8,2), `nota\_falta\_id` INT)  BEGIN  DECLARE valor\_iva DECIMAL(8,2);  DECLARE iva DECIMAL(8,2);  SET @valor\_iva = (valor+(valor\*17)/100);  SET @iva = ((valor\*17)/100);  UPDATE nota\_faltas SET nota\_faltas.valor\_total = (nota\_faltas.valor\_total + valor),  nota\_faltas.iva = (nota\_faltas.iva + @iva),  nota\_faltas.valor\_iva = (nota\_faltas.valor\_iva + @valor\_iva) WHERE nota\_faltas.id = nota\_falta\_id;  END$$  DELIMITER ; |
| 3 | SP\_decrementar\_iten\_nota\_faltas | DELIMITER $$  DROP PROCEDURE IF EXISTS `SP\_decrementar\_iten\_nota\_faltas`$$  CREATE PROCEDURE `SP\_decrementar\_iten\_nota\_faltas`(`produto\_id` INT, `qtd` INT, `desconto` INT, `saida\_id` INT)  BEGIN    DECLARE qtd\_rest INT;  DECLARE qtd\_old INT;  DECLARE preco\_venda DECIMAL(8,2);  DECLARE subtotal decimal(8,2);  DECLARE valor decimal(8,2);  DECLARE nota\_falta\_id INT;    SET @preco\_venda = (SELECT produtos.preco\_venda FROM produtos WHERE produtos.id = produto\_id);  SET @nota\_falta\_id = (SELECT nota\_faltas.id FROM nota\_faltas WHERE nota\_faltas.saida\_id = saida\_id);      SET @qtd\_old = (SELECT iten\_nota\_faltas.quantidade FROM iten\_nota\_faltas  WHERE iten\_nota\_faltas.nota\_falta\_id = @nota\_falta\_id AND  iten\_nota\_faltas.produto\_id = produto\_id);    SET @qtd\_rest = (@qtd\_old - qtd);    SET @valor = (@qtd\_rest \* @preco\_venda);  SET @subtotal = ((@qtd\_rest \* @preco\_venda) - (@qtd\_rest \* @preco\_venda \* desconto) / 100);    UPDATE iten\_nota\_faltas SET  iten\_nota\_faltas.quantidade = @qtd\_rest,  iten\_nota\_faltas.valor = @valor,  iten\_nota\_faltas.desconto = desconto,  iten\_nota\_faltas.subtotal = @subtotal  WHERE iten\_nota\_faltas.nota\_falta\_id = @nota\_falta\_id AND  iten\_nota\_faltas.produto\_id = produto\_id;  END$$  DELIMITER ; |
| 4 | SP\_incrementar\_iten\_nota\_faltas | DELIMITER $$  DROP PROCEDURE IF EXISTS `SP\_incrementar\_iten\_nota\_faltas`$$  CREATE PROCEDURE `SP\_incrementar\_iten\_nota\_faltas`(`produto\_id` INT, `qtd` INT, `desconto` INT, `saida\_id` INT)  BEGIN    DECLARE qtd\_rest INT;  DECLARE qtd\_old INT;  DECLARE preco\_venda DECIMAL(8,2);  DECLARE subtotal decimal(8,2);  DECLARE valor decimal(8,2);  DECLARE nota\_falta\_id INT;    SET @preco\_venda = (SELECT produtos.preco\_venda FROM produtos WHERE produtos.id = produto\_id);  SET @nota\_falta\_id = (SELECT nota\_faltas.id FROM nota\_faltas WHERE nota\_faltas.saida\_id = saida\_id);      SET @qtd\_old = (SELECT iten\_nota\_faltas.quantidade FROM iten\_nota\_faltas  WHERE iten\_nota\_faltas.nota\_falta\_id = @nota\_falta\_id AND  iten\_nota\_faltas.produto\_id = produto\_id);    SET @qtd\_rest = (@qtd\_old + qtd);    SET @valor = (@qtd\_rest \* @preco\_venda);  SET @subtotal = ((@qtd\_rest \* @preco\_venda) - (@qtd\_rest \* @preco\_venda \* desconto) / 100);    UPDATE iten\_nota\_faltas SET  iten\_nota\_faltas.quantidade = @qtd\_rest,  iten\_nota\_faltas.valor = @valor,  iten\_nota\_faltas.desconto = desconto,  iten\_nota\_faltas.subtotal = @subtotal  WHERE iten\_nota\_faltas.nota\_falta\_id = @nota\_falta\_id AND  iten\_nota\_faltas.produto\_id = produto\_id;  END$$  DELIMITER ; |
| 5 | SP\_eliminar\_iten\_nota\_faltas | DELIMITER $$  DROP PROCEDURE IF EXISTS `SP\_eliminar\_iten\_nota\_faltas`$$  CREATE PROCEDURE `SP\_eliminar\_iten\_nota\_faltas`(`produto\_id` INT, `saida\_id` INT)  BEGIN  DECLARE nota\_falta\_id INT;    SET @nota\_falta\_id = (SELECT nota\_faltas.id FROM nota\_faltas WHERE nota\_faltas.saida\_id = saida\_id);    DELETE FROM iten\_nota\_faltas  WHERE iten\_nota\_faltas.nota\_falta\_id = @nota\_falta\_id AND  iten\_nota\_faltas.produto\_id = produto\_id;  END$$  DELIMITER ; |

**Novos Triggers**

**Triggers - Tabela “iten\_nota\_faltas”**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Novos Triigers na **Tabela “iten\_nota\_faltas”** | | | | |
|  | **Nome do Trigger** | **Time** | **Nome da Tabela** | **Script** |
| 1 | tr\_actuatlizar\_valor\_total\_nota\_falta\_after\_insert | AFTER INSERT | iten\_nota\_faltas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_valor\_total\_nota\_falta\_after\_insert`$$  CREATE TRIGGER `tr\_actuatlizar\_valor\_total\_nota\_falta\_after\_insert` AFTER INSERT ON `iten\_nota\_faltas` FOR EACH ROW BEGIN  CALL SP\_incrementar\_valor\_total\_nota\_falta(NEW.subtotal, NEW.nota\_falta\_id);  END$$  DELIMITER ; |
| 2 | tr\_actuatlizar\_iten\_nota\_falta\_after\_update | AFTER UPDATE | iten\_nota\_faltas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_iten\_nota\_falta\_after\_update`$$  CREATE TRIGGER `tr\_actuatlizar\_iten\_nota\_falta\_after\_update` AFTER UPDATE ON `iten\_nota\_faltas` FOR EACH ROW BEGIN  DECLARE subtotal\_dif decimal;  DECLARE produto\_id int;  IF NEW.subtotal > OLD.subtotal THEN  SET @subtotal\_dif = NEW.subtotal - OLD.subtotal;  CALL SP\_incrementar\_valor\_total\_nota\_falta(@subtotal\_dif, NEW.nota\_falta\_id);  ELSEIF NEW.subtotal < OLD.subtotal THEN  SET @subtotal\_dif = OLD.subtotal - NEW.subtotal;  CALL SP\_decrementar\_valor\_total\_nota\_falta(@subtotal\_dif, NEW.nota\_falta\_id);    END IF;  END$$  DELIMITER ; |
| 3 | tr\_actuatlizar\_valor\_total\_nota\_falta\_after\_delete | AFTER DELETE | iten\_nota\_faltas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_valor\_total\_nota\_falta\_after\_delete`$$  CREATE TRIGGER `tr\_actuatlizar\_valor\_total\_nota\_falta\_after\_delete` AFTER DELETE ON `iten\_nota\_faltas` FOR EACH ROW BEGIN  CALL SP\_decrementar\_valor\_total\_nota\_falta(OLD.subtotal, OLD.nota\_falta\_id);  END$$  DELIMITER ; |

**Atualização dos Triggers Existentes**

**Triggers – Tabela “iten\_saidas”**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atualização dos Triggers Existentes, Triigers na **Tabela “iten\_saidas”** | | | | |
|  | **Trigger** | **Time** | **Tabela** | **Script** |
| 1 | tr\_actuatlizar\_valor\_total\_saida\_after\_insert | AFTER INSERT | iten\_saidas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_valor\_total\_saida\_after\_insert`$$  CREATE TRIGGER `tr\_actuatlizar\_valor\_total\_saida\_after\_insert` AFTER INSERT ON `iten\_saidas` FOR EACH ROW BEGIN    CALL SP\_incrementar\_valor\_total\_saida(NEW.subtotal, NEW.saida\_id);  CALL SP\_incrementar\_iten\_nota\_faltas(NEW.produto\_id, NEW.quantidade, NEW.desconto, NEW.saida\_id);  END  DELIMITER ; |
| 2 | tr\_actuatlizar\_iten\_saidas\_after\_update | AFTER UPDATE | iten\_saidas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_iten\_saidas\_after\_update`$$  CREATE TRIGGER `tr\_actuatlizar\_iten\_saidas\_after\_update` AFTER UPDATE ON `iten\_saidas` FOR EACH ROW BEGIN  DECLARE quantidade INT;    IF NEW.quantidade > OLD.quantidade THEN  SET @quantidade = NEW.quantidade - OLD.quantidade;  CALL SP\_incrementar\_iten\_nota\_faltas(NEW.produto\_id, @quantidade, NEW.desconto, NEW.saida\_id);  ELSEIF NEW.quantidade < OLD.quantidade THEN  SET @quantidade = OLD.quantidade - NEW.quantidade;  CALL SP\_decrementar\_iten\_nota\_faltas(NEW.produto\_id, @quantidade, NEW.desconto, NEW.saida\_id);  END IF;  /\* Nao temos igual porque garante que o iten\_nota\_faltas possa ser actualizado correctamente, pois haveria conflito a partir da actualizacao do iten\_guiaentrega que por sua vez chama o actualizar\_rest\_iten\_saidas\*/    IF NEW.subtotal > OLD.subtotal THEN  SET @subtotal\_dif = NEW.subtotal - OLD.subtotal;  CALL SP\_incrementar\_valor\_total\_saida(@subtotal\_dif, NEW.saida\_id);  ELSEIF NEW.subtotal < OLD.subtotal THEN  SET @subtotal\_dif = OLD.subtotal - NEW.subtotal;  CALL SP\_decrementar\_valor\_total\_saida(@subtotal\_dif, NEW.saida\_id);    END IF;  END$$  DELIMITER ; |
| 3 | tr\_actuatlizar\_valor\_total\_saida\_after\_delete | AFTER DELETE | iten\_saidas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_valor\_total\_saida\_after\_delete`$$  CREATE TRIGGER `tr\_actuatlizar\_valor\_total\_saida\_after\_delete` AFTER DELETE ON `iten\_saidas` FOR EACH ROW BEGIN  CALL SP\_decrementar\_valor\_total\_saida(OLD.subtotal, OLD.saida\_id);  CALL SP\_decrementar\_iten\_nota\_faltas(OLD.produto\_id, OLD.quantidade, OLD.desconto, OLD.saida\_id);  CALL SP\_eliminar\_iten\_nota\_faltas(OLD.produto\_id, OLD.saida\_id);  END$$  DELIMITER ; |

**Triggers – Tabela “iten\_guiaentregas”**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atualização dos Triggers Existentes, Triigers na **Tabela “iten\_guiaentregas”** | | | | |
|  | **Trigger** | **Time** | **Tabela** | **Script** |
| 1 | tr\_actuatlizar\_rest\_iten\_saidas\_after\_insert | AFTER INSERT | iten\_guiaentregas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_rest\_iten\_saidas\_after\_insert`$$  CREATE TRIGGER `tr\_actuatlizar\_rest\_iten\_saidas\_after\_insert` AFTER INSERT ON `iten\_guiaentregas` FOR EACH ROW BEGIN  DECLARE saida\_id int;  DECLARE produto\_id int;  SET @saida\_id = (SELECT guia\_entregas.saida\_id FROM guia\_entregas, iten\_guiaentregas  WHERE iten\_guiaentregas.guia\_entrega\_id = guia\_entregas.id AND  iten\_guiaentregas.guia\_entrega\_id = NEW.guia\_entrega\_id AND  iten\_guiaentregas.produto\_id = NEW.produto\_id);    SET @produto\_id = NEW.produto\_id;  CALL SP\_incrementar\_valor\_total\_guia\_entrega(NEW.subtotal, NEW.guia\_entrega\_id);  CALL SP\_decrementar\_rest\_iten\_saidas(NEW.quantidade, @saida\_id, @produto\_id);  CALL SP\_decrementar\_iten\_nota\_faltas(@produto\_id, NEW.quantidade, NEW.desconto, @saida\_id);  CALL SP\_decrementar\_qtd\_produto(NEW.quantidade, NEW.produto\_id);  END$$  DELIMITER ; |
| 2 | tr\_actuatlizar\_rest\_iten\_saidas\_after\_update | AFTER UPDATE | iten\_guiaentregas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_rest\_iten\_saidas\_after\_update`$$  CREATE TRIGGER `tr\_actuatlizar\_rest\_iten\_saidas\_after\_update` AFTER UPDATE ON `iten\_guiaentregas` FOR EACH ROW BEGIN  DECLARE saida\_id int;  DECLARE produto\_id int;  DECLARE quantidade int;  DECLARE subtotal decimal;  SET @saida\_id = (SELECT guia\_entregas.saida\_id FROM guia\_entregas, iten\_guiaentregas  WHERE iten\_guiaentregas.guia\_entrega\_id = guia\_entregas.id AND  iten\_guiaentregas.guia\_entrega\_id = NEW.guia\_entrega\_id AND  iten\_guiaentregas.produto\_id = NEW.produto\_id);    SET @produto\_id = NEW.produto\_id;    IF NEW.quantidade > OLD.quantidade THEN  SET @quantidade = NEW.quantidade - OLD.quantidade;  CALL SP\_decrementar\_qtd\_produto(@quantidade, NEW.produto\_id);  CALL SP\_decrementar\_rest\_iten\_saidas(@quantidade, @saida\_id, @produto\_id);  CALL SP\_decrementar\_iten\_nota\_faltas(@produto\_id, @quantidade, NEW.desconto, @saida\_id);  ELSEIF NEW.quantidade < OLD.quantidade THEN  SET @quantidade = OLD.quantidade - NEW.quantidade;  CALL SP\_incrementar\_qtd\_produto(@quantidade, NEW.produto\_id);  CALL SP\_incrementar\_rest\_iten\_saidas(@quantidade, @saida\_id, @produto\_id);  CALL SP\_incrementar\_iten\_nota\_faltas(@produto\_id, @quantidade, NEW.desconto, @saida\_id);  END IF;  IF NEW.subtotal > OLD.subtotal THEN  SET @subtotal = NEW.subtotal - OLD.subtotal;  CALL SP\_incrementar\_valor\_total\_guia\_entrega(@subtotal, NEW.guia\_entrega\_id);  ELSEIF NEW.subtotal < OLD.subtotal THEN  SET @subtotal = OLD.subtotal - NEW.subtotal;  CALL SP\_decrementar\_valor\_total\_guia\_entrega(@subtotal, NEW.guia\_entrega\_id);  END IF;    END$$  DELIMITER ; |
| 3 | tr\_actuatlizar\_rest\_iten\_saidas\_before\_delete | BEFORE DELETE | iten\_guiaentregas | DELIMITER $$  DROP TRIGGER IF EXISTS `tr\_actuatlizar\_rest\_iten\_saidas\_before\_delete`$$  CREATE TRIGGER `tr\_actuatlizar\_rest\_iten\_saidas\_before\_delete` BEFORE DELETE ON `iten\_guiaentregas` FOR EACH ROW BEGIN  DECLARE saida\_id int;  DECLARE produto\_id int;  DECLARE nota\_falta\_id int;    SET @saida\_id = (SELECT DISTINCT guia\_entregas.saida\_id FROM guia\_entregas, iten\_guiaentregas  WHERE guia\_entregas.id = OLD.guia\_entrega\_id AND  iten\_guiaentregas.produto\_id = OLD.produto\_id);    SET @nota\_falta\_id = (SELECT nota\_faltas.id FROM nota\_faltas WHERE nota\_faltas.saida\_id = @saida\_id);  SET @produto\_id = OLD.produto\_id;  CALL SP\_incrementar\_rest\_iten\_saidas(OLD.quantidade, @saida\_id, @produto\_id);  CALL SP\_incrementar\_iten\_nota\_faltas(@produto\_id, OLD.quantidade, OLD.desconto, @saida\_id);  CALL SP\_incrementar\_qtd\_produto(OLD.quantidade, @produto\_id);  END$$  DELIMITER ; |

**OUTROS ACRESCIMOS/ACTUALIZAÇÕES:**

É necessário inserir a permissão “visualizar\_nota\_de\_falta” na tabela permissões pelo SGBD e atribuir ao tipo de usuário esta permissão no Sistema para que o mesmo possa aceder as notas de falta pelo Sistema;

Atualizações de Bugs

Após finalizar a Nota de Falta, foi possível constatar que não era possível eliminar a saída. Isto acontece porque a saída esta atrelada a nota de falta, a nota de falta é criada sempre que se cria uma saída e esta saída, por sua vez passa a fazer parte desta nota de falta. O sistema ao eliminar a siada ainda não é capaz de eliminar a nota de falta antes desta accão de eliminar a saída (situação de chave estrangeira).

Solução:

1. Criar um trigger “before delete” na saída, para que este seja responsável por eliminar primeiro a nota de falta, e só depois elimina-se a saída.
2. Solução implementada com sucesso.