Relazione Progetto di Basi di Dati

Cash Flow Web

Giovanni Maria Savoca – 970094

[Giovannimaria.savoca@studio.unibo.it](mailto:Giovannimaria.savoca@studio.unibo.it)

**Introduzione**

Breve descrizione del progetto, includendo lo scopo e gli obiettivi principali del sistema informativo creato, come la gestione di risparmi, transazioni, conti, e monitoraggio di budget.

**1. Raccolta e Analisi dei Requisiti**

Specifiche sui Dati

Elenco e descrizione dei dati gestiti dal sistema, come:

* Conti bancari
* Transazioni (entrate e uscite)
* Risparmi programmati
* Budget massimi per categorie di spesa
* Profili utente

Descrizione delle funzionalità chiave come:

* Allocare automaticamente i risparmi giornalieri
* Creare transazioni basate su template
* Gestire il credito e il debito con relative transazioni
* Monitorare e impedire superamenti di budget
* Tavola Media dei Volumi

Glossario dei Dati

Definizioni dei termini tecnici utilizzati, assicurandosi che siano comprensibili anche per chi non ha familiarità con il dominio del progetto.

**2. Progettazione Concettuale**

Diagramma E-R

Inserire il diagramma Entità-Relazione che rappresenta visivamente le entità del database e le loro relazioni.

Dizionario delle Entità/Relazioni

Descrizione dettagliata delle entità e delle relazioni, incluse le proprietà e i vincoli associati a ciascuna.

**3. Progettazione Logica**

Lista delle Tabelle con Vincoli di Chiavi

Descrizione delle tabelle SQL, con dettagli sui vincoli di chiave primaria e chiave esterna.

**4. Normalizzazione**

Discussione su eventuali passaggi di normalizzazione effettuati per ridurre la ridondanza e migliorare l'integrità dei dati.

**5. Descrizione delle Funzionalità dell'Applicazione Web**

Panoramica ad alto livello delle funzionalità dell'applicazione web, spiegando come ciascuna funzionalità interagisce con il database.

Appendice

Codice SQL Completo

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET time\_zone = "+00:00";

CREATE DATABASE IF NOT EXISTS `cashflowweb` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci;

USE `cashflowweb`;

DELIMITER $$

DROP PROCEDURE IF EXISTS `AllocateSavings`$$

CREATE DEFINER=`root`@`localhost` PROCEDURE `AllocateSavings` (IN `SavingsID` INT) BEGIN

DECLARE StartDate DATE;

DECLARE EndDate DATE;

DECLARE TotalAmount DECIMAL(10,2);

DECLARE AccountID INT;

DECLARE Days INT;

DECLARE DailyAmount DECIMAL(10,2);

SELECT DataInizio, DataFine, ImportoRisparmiato, IDConto

INTO StartDate, EndDate, TotalAmount, AccountID

FROM risparmi

WHERE ID = SavingsID;

SET Days = DATEDIFF(EndDate, StartDate);

SET DailyAmount = TotalAmount / Days;

WHILE StartDate <= EndDate DO

UPDATE conto SET Saldo = Saldo - DailyAmount WHERE ID = AccountID;

-- Aggiungi una transazione per il risparmio giornaliero

INSERT INTO transazione (Is\_Expense, Importo, IDConto, DataTransazione, IDCategoriaPrimaria, IDCategoriaSecondaria)

VALUES (1, DailyAmount, AccountID, StartDate, NULL, NULL); -- Assumiendo che non ci siano categorie specifiche per questi risparmi

SET StartDate = DATE\_ADD(StartDate, INTERVAL 1 DAY);

END WHILE;

END$$

DROP PROCEDURE IF EXISTS `AllocateSavingsDaily`$$

CREATE DEFINER=`root`@`localhost` PROCEDURE `AllocateSavingsDaily` () BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE aSavingsID INT;

DECLARE cur CURSOR FOR SELECT ID FROM risparmi;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO aSavingsID;

IF done THEN

LEAVE read\_loop;

END IF;

CALL AllocateSavings(aSavingsID); -- Assumi che questa sia la funzione esistente che gestisce un singolo ID

END LOOP;

CLOSE cur;

END$$

DROP PROCEDURE IF EXISTS `CreateTransactionFromTemplate`$$

CREATE DEFINER=`root`@`localhost` PROCEDURE `CreateTransactionFromTemplate` (IN `TemplateID` INT) BEGIN

DECLARE ExpenseType TINYINT;

DECLARE Amount DECIMAL(10,2);

DECLARE AccountID INT;

DECLARE PrimaryCategoryID INT;

DECLARE SecondaryCategoryID INT;

DECLARE Description VARCHAR(255);

SELECT Is\_Expense, Importo, IDConto, IDCategoriaPrimaria, IDCategoriaSecondaria, Descrizione

INTO ExpenseType, Amount, AccountID, PrimaryCategoryID, SecondaryCategoryID, Description

FROM template\_transazioni

WHERE ID = TemplateID;

INSERT INTO transazione (Is\_Expense, Importo, IDTemplate, IDConto, DataTransazione, IDCategoriaPrimaria, IDCategoriaSecondaria)

VALUES (ExpenseType, Amount, TemplateID, AccountID, CURDATE(), PrimaryCategoryID, SecondaryCategoryID);

END$$

DELIMITER ;

DROP TABLE IF EXISTS `assconti`;

CREATE TABLE `assconti` (

`IDProfilo` int(11) DEFAULT NULL,

`IDConto` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `budgetmax`;

CREATE TABLE `budgetmax` (

`ID` int(11) NOT NULL,

`NomeBudget` varchar(255) DEFAULT NULL,

`ImportoMax` decimal(10,2) DEFAULT NULL,

`DataInizio` date DEFAULT NULL,

`DataFine` date DEFAULT NULL,

`IDPrimaryCategory` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `categoriaprimaria`;

CREATE TABLE `categoriaprimaria` (

`ID` int(11) NOT NULL,

`NomeCategoria` varchar(255) DEFAULT NULL,

`DescrizioneCategoria` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `categoriasecondaria`;

CREATE TABLE `categoriasecondaria` (

`ID` int(11) NOT NULL,

`IDCategoriaPrimaria` int(11) DEFAULT NULL,

`NomeCategoria` varchar(255) DEFAULT NULL,

`DescrizioneCategoria` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `conto`;

CREATE TABLE `conto` (

`ID` int(11) NOT NULL,

`NomeConto` varchar(255) DEFAULT NULL,

`Saldo` decimal(10,2) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `credit`;

CREATE TABLE `credit` (

`ID` int(11) NOT NULL,

`ImportoCredito` decimal(10,2) DEFAULT NULL,

`NomeImporto` varchar(255) DEFAULT NULL,

`DataConcessione` date DEFAULT NULL,

`DataEstinsione` date DEFAULT NULL,

`Note` varchar(255) DEFAULT NULL,

`IDConto` int(11) DEFAULT NULL,

`IDCategoriaPrimaria` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TRIGGER IF EXISTS `create\_transaction\_on\_credit\_insert`;

DELIMITER $$

CREATE TRIGGER `create\_transaction\_on\_credit\_insert` AFTER INSERT ON `credit` FOR EACH ROW BEGIN

INSERT INTO transazione (Is\_Expense, Importo, IDConto, DataTransazione, IDCategoriaPrimaria)

VALUES (1, NEW.ImportoCredito, NEW.IDConto, NEW.DataConcessione, NEW.IDCategoriaPrimaria);

END

$$

DELIMITER ;

DROP TABLE IF EXISTS `debit`;

CREATE TABLE `debit` (

`ID` int(11) NOT NULL,

`ImportoDebito` decimal(10,2) DEFAULT NULL,

`NomeImporto` varchar(255) DEFAULT NULL,

`DataConcessione` date DEFAULT NULL,

`DataEstinsione` date DEFAULT NULL,

`Note` varchar(255) DEFAULT NULL,

`IDConto` int(11) DEFAULT NULL,

`IDCategoriaPrimaria` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TRIGGER IF EXISTS `create\_transaction\_on\_debit\_insert`;

DELIMITER $$

CREATE TRIGGER `create\_transaction\_on\_debit\_insert` AFTER INSERT ON `debit` FOR EACH ROW BEGIN

INSERT INTO transazione (Is\_Expense, Importo, IDConto, DataTransazione, IDCategoriaPrimaria)

VALUES (0, NEW.ImportoDebito, NEW.IDConto, NEW.DataConcessione, NEW.IDCategoriaPrimaria);

END

$$

DELIMITER ;

DROP TABLE IF EXISTS `obiettivifinanziari`;

CREATE TABLE `obiettivifinanziari` (

`ID` int(11) NOT NULL,

`NomeObiettivo` varchar(255) DEFAULT NULL,

`ImportoObiettivo` decimal(10,2) DEFAULT NULL,

`DataScadenza` date DEFAULT NULL,

`IDConto` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `profili`;

CREATE TABLE `profili` (

`ID` int(11) NOT NULL,

`NomeProfilo` varchar(255) DEFAULT NULL,

`Saldo\_totale` decimal(10,2) DEFAULT NULL,

`Email` varchar(255) DEFAULT NULL,

`Password` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `profili\_categoriaprimaria`;

CREATE TABLE `profili\_categoriaprimaria` (

`IDProfilo` int(11) NOT NULL,

`IDCategoriaPrimaria` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `risparmi`;

CREATE TABLE `risparmi` (

`ID` int(11) NOT NULL,

`ImportoRisparmiato` decimal(10,2) DEFAULT NULL,

`DataInizio` date DEFAULT NULL,

`DataFine` date DEFAULT NULL,

`IDConto` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `template\_transazioni`;

CREATE TABLE `template\_transazioni` (

`ID` int(11) NOT NULL,

`NomeTemplate` varchar(255) DEFAULT NULL,

`Is\_Expense` tinyint(1) DEFAULT NULL,

`Importo` decimal(10,2) DEFAULT NULL,

`IDConto` int(11) DEFAULT NULL,

`IDCategoriaPrimaria` int(11) DEFAULT NULL,

`IDCategoriaSecondaria` int(11) DEFAULT NULL,

`Descrizione` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TABLE IF EXISTS `transazione`;

CREATE TABLE `transazione` (

`ID` int(11) NOT NULL,

`Is\_Expense` tinyint(1) DEFAULT NULL,

`Importo` decimal(10,2) DEFAULT NULL,

`IDTemplate` int(11) DEFAULT NULL,

`IDConto` int(11) DEFAULT NULL,

`DataTransazione` date DEFAULT NULL,

`IDCategoriaPrimaria` int(11) DEFAULT NULL,

`IDCategoriaSecondaria` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

DROP TRIGGER IF EXISTS `CheckBudgetBeforeTransaction`;

DELIMITER $$

CREATE TRIGGER `CheckBudgetBeforeTransaction` BEFORE INSERT ON `transazione` FOR EACH ROW BEGIN

DECLARE MaxAmount DECIMAL(10,2);

DECLARE TotalSpent DECIMAL(10,2);

SELECT ImportoMax INTO MaxAmount

FROM budgetmax

WHERE IDPrimaryCategory = NEW.IDCategoriaPrimaria AND CURDATE() BETWEEN DataInizio AND DataFine;

IF MaxAmount IS NOT NULL THEN

SELECT SUM(Importo) INTO TotalSpent

FROM transazione

WHERE IDCategoriaPrimaria = NEW.IDCategoriaPrimaria AND Is\_Expense = 1;

IF (TotalSpent + NEW.Importo > MaxAmount) THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Budget limit exceeded for this category.';

END IF;

END IF;

END

$$

DELIMITER ;

DROP TRIGGER IF EXISTS `after\_transazione\_delete`;

DELIMITER $$

CREATE TRIGGER `after\_transazione\_delete` AFTER DELETE ON `transazione` FOR EACH ROW BEGIN

IF OLD.Is\_Expense = 1 THEN

UPDATE conto SET Saldo = Saldo + OLD.Importo

WHERE ID = OLD.IDConto;

ELSE

UPDATE conto SET Saldo = Saldo - OLD.Importo

WHERE ID = OLD.IDConto;

END IF;

END

$$

DELIMITER ;

DROP TRIGGER IF EXISTS `after\_transazione\_insert`;

DELIMITER $$

CREATE TRIGGER `after\_transazione\_insert` AFTER INSERT ON `transazione` FOR EACH ROW BEGIN

IF NEW.Is\_Expense = 1 THEN

UPDATE conto SET Saldo = Saldo - NEW.Importo

WHERE ID = NEW.IDConto;

ELSE

UPDATE conto SET Saldo = Saldo + NEW.Importo

WHERE ID = NEW.IDConto;

END IF;

END

$$

DELIMITER ;

DROP TRIGGER IF EXISTS `after\_transazione\_update`;

DELIMITER $$

CREATE TRIGGER `after\_transazione\_update` AFTER UPDATE ON `transazione` FOR EACH ROW BEGIN

IF OLD.Is\_Expense = 1 THEN

UPDATE conto SET Saldo = Saldo + OLD.Importo

WHERE ID = OLD.IDConto;

ELSE

UPDATE conto SET Saldo = Saldo - OLD.Importo

WHERE ID = OLD.IDConto;

END IF;

IF NEW.Is\_Expense = 1 THEN

UPDATE conto SET Saldo = Saldo - NEW.Importo

WHERE ID = NEW.IDConto;

ELSE

UPDATE conto SET Saldo = Saldo + NEW.Importo

WHERE ID = NEW.IDConto;

END IF;

END

$$

DELIMITER ;

ALTER TABLE `assconti`

ADD KEY `fk\_assconti\_profilo` (`IDProfilo`),

ADD KEY `fk\_assconti\_conto` (`IDConto`);

ALTER TABLE `budgetmax`

ADD PRIMARY KEY (`ID`),

ADD KEY `fk\_budgetmax\_primarycategory` (`IDPrimaryCategory`);

ALTER TABLE `categoriaprimaria`

ADD PRIMARY KEY (`ID`);

ALTER TABLE `categoriasecondaria`

ADD PRIMARY KEY (`ID`),

ADD KEY `categoriasecondaria\_primaria\_fk` (`IDCategoriaPrimaria`);

ALTER TABLE `conto`

ADD PRIMARY KEY (`ID`);

ALTER TABLE `credit`

ADD PRIMARY KEY (`ID`),

ADD KEY `credit\_conto\_fk` (`IDConto`),

ADD KEY `fk\_credit\_categoriaprimaria` (`IDCategoriaPrimaria`);

ALTER TABLE `debit`

ADD PRIMARY KEY (`ID`),

ADD KEY `debit\_conto\_fk` (`IDConto`),

ADD KEY `fk\_debit\_categoriaprimaria` (`IDCategoriaPrimaria`);

ALTER TABLE `obiettivifinanziari`

ADD PRIMARY KEY (`ID`),

ADD KEY `obiettivi\_conto\_fk` (`IDConto`);

ALTER TABLE `profili`

ADD PRIMARY KEY (`ID`);

ALTER TABLE `profili\_categoriaprimaria`

ADD PRIMARY KEY (`IDProfilo`,`IDCategoriaPrimaria`),

ADD KEY `fk\_profili\_categoriaprimaria\_profilo` (`IDProfilo`),

ADD KEY `fk\_profili\_categoriaprimaria\_categoria` (`IDCategoriaPrimaria`);

ALTER TABLE `risparmi`

ADD PRIMARY KEY (`ID`),

ADD KEY `risparmi\_conto\_fk` (`IDConto`);

ALTER TABLE `template\_transazioni`

ADD PRIMARY KEY (`ID`),

ADD KEY `template\_transazioni\_conto\_fk` (`IDConto`),

ADD KEY `template\_transazioni\_primaria\_fk` (`IDCategoriaPrimaria`),

ADD KEY `template\_transazioni\_secondaria\_fk` (`IDCategoriaSecondaria`);

ALTER TABLE `transazione`

ADD PRIMARY KEY (`ID`),

ADD KEY `transazione\_template\_fk` (`IDTemplate`),

ADD KEY `transazione\_conto\_fk` (`IDConto`),

ADD KEY `transazione\_primaria\_fk` (`IDCategoriaPrimaria`),

ADD KEY `transazione\_secondaria\_fk` (`IDCategoriaSecondaria`);

ALTER TABLE `budgetmax`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `categoriaprimaria`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `categoriasecondaria`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `conto`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `credit`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `debit`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `obiettivifinanziari`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `profili`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `risparmi`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `template\_transazioni`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `transazione`

MODIFY `ID` int(11) NOT NULL AUTO\_INCREMENT;

ALTER TABLE `assconti`

ADD CONSTRAINT `fk\_assconti\_conto` FOREIGN KEY (`IDConto`) REFERENCES `conto` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_assconti\_profilo` FOREIGN KEY (`IDProfilo`) REFERENCES `profili` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE `budgetmax`

ADD CONSTRAINT `fk\_budgetmax\_primarycategory` FOREIGN KEY (`IDPrimaryCategory`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE `categoriasecondaria`

ADD CONSTRAINT `categoriasecondaria\_primaria\_fk` FOREIGN KEY (`IDCategoriaPrimaria`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE `credit`

ADD CONSTRAINT `credit\_conto\_fk` FOREIGN KEY (`IDConto`) REFERENCES `conto` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_credit\_categoriaprimaria` FOREIGN KEY (`IDCategoriaPrimaria`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE `debit`

ADD CONSTRAINT `debit\_conto\_fk` FOREIGN KEY (`IDConto`) REFERENCES `conto` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_debit\_categoriaprimaria` FOREIGN KEY (`IDCategoriaPrimaria`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE `profili\_categoriaprimaria`

ADD CONSTRAINT `fk\_profili\_categoriaprimaria\_categoria` FOREIGN KEY (`IDCategoriaPrimaria`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_profili\_categoriaprimaria\_profilo` FOREIGN KEY (`IDProfilo`) REFERENCES `profili` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE `risparmi`

ADD CONSTRAINT `risparmi\_conto\_fk` FOREIGN KEY (`IDConto`) REFERENCES `conto` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE `template\_transazioni`

ADD CONSTRAINT `fk\_template\_transazioni\_primaria` FOREIGN KEY (`IDCategoriaPrimaria`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_template\_transazioni\_primaria\_new` FOREIGN KEY (`IDCategoriaPrimaria`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_template\_transazioni\_secondaria` FOREIGN KEY (`IDCategoriaSecondaria`) REFERENCES `categoriasecondaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_template\_transazioni\_secondaria\_new` FOREIGN KEY (`IDCategoriaSecondaria`) REFERENCES `categoriasecondaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE,

ADD CONSTRAINT `template\_transazioni\_conto\_fk` FOREIGN KEY (`IDConto`) REFERENCES `conto` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE `transazione`

ADD CONSTRAINT `fk\_transazione\_primaria` FOREIGN KEY (`IDCategoriaPrimaria`) REFERENCES `categoriaprimaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE,

ADD CONSTRAINT `fk\_transazione\_secondaria` FOREIGN KEY (`IDCategoriaSecondaria`) REFERENCES `categoriasecondaria` (`ID`) ON DELETE SET NULL ON UPDATE CASCADE,

ADD CONSTRAINT `transazione\_conto\_fk` FOREIGN KEY (`IDConto`) REFERENCES `conto` (`ID`) ON DELETE CASCADE ON UPDATE CASCADE;

DELIMITER $$

DROP EVENT IF EXISTS `allocateSavingsEvent`$$

CREATE DEFINER=`root`@`localhost` EVENT `allocateSavingsEvent` ON SCHEDULE EVERY 1 DAY STARTS '2023-01-01 00:00:00' ON COMPLETION NOT PRESERVE ENABLE DO CALL AllocateSavingsDaily()$$

DROP EVENT IF EXISTS `check\_debit\_credit\_expiry\_event`$$

CREATE DEFINER=`root`@`localhost` EVENT `check\_debit\_credit\_expiry\_event` ON SCHEDULE EVERY 1 DAY STARTS '2024-05-08 16:46:17' ON COMPLETION NOT PRESERVE ENABLE DO BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE debtCreditID INT;

DECLARE debtCreditType VARCHAR(10);

DECLARE cur CURSOR FOR

SELECT ID, 'debit' AS type FROM debit WHERE DataEstinsione = CURDATE()

UNION ALL

SELECT ID, 'credit' AS type FROM credit WHERE DataEstinsione = CURDATE();

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN cur;

read\_loop: LOOP

FETCH cur INTO debtCreditID, debtCreditType;

IF done THEN

LEAVE read\_loop;

END IF;

IF debtCreditType = 'debit' THEN

INSERT INTO transazione (Is\_Expense, Importo, IDConto, DataTransazione, IDCategoriaPrimaria)

SELECT 1, ImportoDebito, IDConto, DataEstinsione, IDCategoriaPrimaria

FROM debit WHERE ID = debtCreditID;

ELSE

INSERT INTO transazione (Is\_Expense, Importo, IDConto, DataTransazione, IDCategoriaPrimaria)

SELECT 0, ImportoCredito, IDConto, DataEstinsione, IDCategoriaPrimaria

FROM credit WHERE ID = debtCreditID;

END IF;

END LOOP;

CLOSE cur;

END$$

DELIMITER ;