**Database Group Project Team 11 Final Submission**

In our “Budget Application Project”, we tried to understand the importance of budget management and creating the best application for our users.

**Our Mission Statement:**

* Our mission is to enable people and businesses to attain financial well-being by providing them with smart budget analysis and effective management.

1. An unnormalized set of data, including

* A list of the data items being maintained
* A set of data in a grid or spreadsheet

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. E-R Diagram for our 3NF database (include the participation/cardinality information… 0…\* , 1..1 , etc.)

metin, diyagram, paralel, plan içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. SQL queries for creating tables and inserting data from the UNF database table into the tables created.

**User**

CREATE TABLE User AS

SELECT first\_name, last\_name,

username, gender,email,

phone, password, income\_frequency, expense\_frequency , user\_total\_balance

FROM UNF;

ALTER TABLE User

ADD COLUMN profile\_image MEDIUMBLOB,

ADD COLUMN job VARCHAR(255),

ADD COLUMN user\_id BIGINT NOT NULL PRIMARY KEY AUTO\_INCREMENT;

ALTER TABLE UNF

DROP COLUMN first\_name,last\_name, username, gender,email,

phone, password, income\_frequency, expense\_frequency , user\_total\_balance;

**GOAL**

CREATE table goal AS

SELECT name\_of\_goal, target\_amount, start\_date\_goal, end\_date\_goal, goal\_current\_balance

from unf;

ALTER table goal

ADD COLUMN user\_id BIGINT NOT NULL,

ADD COLUMN goal\_id bigint not null PRIMARY key AUTO\_INCREMENT;

ALTER TABLE goal ADD CONSTRAINT user\_id FOREIGN KEY (user\_id) REFERENCES user(user\_id) ON UPDATE CASCADE;

ALTER TABLE UNF DROP COLUMN name\_of\_goal, target\_amount, start\_date\_goal, end\_date\_goal, goal\_current\_balance;

**BUDGET**

CREATE TABLE Budget AS

SELECT budgeted\_amount, start\_date\_budget,

end\_date\_budget

FROM UNF;

ALTER TABLE Budget

ADD COLUMN user\_id BIGINT NOT NULL,

ADD COLUMN budget\_id BIGINT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

ADD COLUMN expected\_budget INT;

ALTER TABLE Budget

ADD CONSTRAINT PK\_budget\_user\_id FOREIGN KEY (user\_id) REFERENCES User (user\_id) ON UPDATE CASCADE;

ALTER TABLE UNF DROP COLUMN budgeted\_amount, start\_date\_budget,end\_date\_budget ;

**INCOME**

CREATE TABLE income AS SELECT source\_name, date\_received, date\_recorded, pay\_period\_start\_date, pay\_period\_end\_date,amount\_received FROM UNF;

ALTER TABLE income

ADD COLUMN income\_id bigint NOT NULL AUTO\_INCREMENT PRIMARY KEY,

ADD COLUMN budget\_id BIGINT NOT NULL;

ALTER TABLE INCOME ADD CONSTRAINT FK\_income\_budget\_id FOREIGN KEY (budget\_id) REFERENCES Budget (budget\_id) ON UPDATE CASCADE;

ALTER TABLE UNF DROP COLUMN source\_name, date\_received, date\_recorded, pay\_period\_start\_date, pay\_period\_end\_date,amount\_received;

**EXPENSE CATEGORY**

CREATE TABLE Expense\_Category AS

SELECT budget\_type, exp\_category\_name

FROM UNF;

ALTER TABLE Expense\_Category

ADD COLUMN category\_id BIGINT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

ADD COLUMN budget\_id BIGINT NOT NULL,;

ALTER TABLE Expense\_Category ADD CONSTRAINT FK\_expense\_category\_budget\_id FOREIGN KEY (budget\_id) REFERENCES Budget (budget\_id) ON UPDATE CASCADE;

ALTER TABLE UNF DROP COLUMN budget\_type, exp\_category\_name;

**EXPENSE**

CREATE TABLE Expense AS

SELECT name\_of\_expense, date\_of\_expense, amount\_expense,payment\_method, recurring

FROM UNF;

ALTER TABLE Expense

ADD COLUMN category\_id BIGINT,

ADD COLUMN budget\_id BIGINT NOT NULL,

ADD COLUMN expense\_id BIGINT NOT NULL AUTO\_INCREMENT PRIMARY KEY;

ALTER TABLE Expense

ADD CONSTRAINT FK\_exps\_cat\_id FOREIGN KEY (category\_id) REFERENCES Expense\_Category (category\_id) ON DELETE SET NULL;

ALTER TABLE expense

ADD CONSTRAINT FK\_budget\_id FOREIGN KEY (budget\_id) REFERENCES Budget (budget\_id);

ALTER TABLE UNF DROP COLUMN name\_of\_expense, date\_of\_expense, amount\_expense,payment\_method, recurring;

**Planned Expense**

CREATE TABLE Planned\_Expense (

planned\_date DATE,

planned\_amount INT,

description varchar(255),

planned\_expense\_id BIGINT AUTO\_INCREMENT PRIMARY KEY,

expense\_id BIGINT NOT NULL);

ALTER TABLE Planned\_Expense

ADD CONSTRAINT FK\_expense\_id FOREIGN KEY (expense\_id) REFERENCES Expense (expense\_id) ON UPDATE CASCADE;

**Actual Expense**

CREATE TABLE Planned\_Expense (

difference DOUBLE,

description varchar(255),

actual\_expense\_id BIGINT AUTO\_INCREMENT PRIMARY KEY,

expense\_id BIGINT NOT NULL);

ALTER TABLE Actual\_Expense

ADD CONSTRAINT FK\_expense\_id FOREIGN KEY (expense\_id) REFERENCES Expense (expense\_id ON UPDATE CASCADE);

**Payment Method**

CREATE TABLE Payment\_Method (

payment\_method\_id BIGINT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

payment\_type varchar(255));

**DELETING UNF TABLE**

DROP TABLE UNF

1. A view to recreate the original dataset from our 3NF Tables (this will be used for comparison purposes to the original dataset we provide as number 1 of this deliverable)

CREATE VIEW ComparisonView AS

SELECT

u.first\_name, u.last\_name, u.username, u.gender, u.email, u.phone, u.password, u.income\_frequency, u.expense\_frequency, u.user\_total\_balance, g.name\_of\_goal, g.target\_amount, g.start\_date\_goal, g.end\_date\_goal, g.goal\_current\_balance, b.budgeted\_amount, b.start\_date\_budget, b.end\_date\_budget, i.source\_name, i.date\_received, i.date\_recorded, i.pay\_period\_start\_date, i.pay\_period\_end\_date, i.amount\_received, ec.exp\_category\_name, e.name\_of\_expense, e.date\_of\_expense, e.amount\_expense, e.payment\_method, e.recurring

FROM User u

LEFT JOIN Goal g ON u.user\_id = g.user\_id

LEFT JOIN Budget b ON u.user\_id = b.user\_id

LEFT JOIN Income i ON b.budget\_id = i.budget\_id

LEFT JOIN Expense\_Category ec ON b.budget\_id = ec.budget\_id AND u.user\_id = ec.user\_id

LEFT JOIN Expense e ON ec.category\_id = e.category\_id AND u.user\_id = e.user\_id AND b.budget\_id = ec.budget\_id;

1. Customer Requests
2. List all planned expenses for the next month
3. Show total amount spent per month on each category for the previous year
4. Change an amount budgeted for a particular expense
5. Add a new budget expectation for next month based on previous month's expenses
6. Remove a category from the database
7. List all planned expenses for the next month:,

SELECT name\_of\_expense, planned\_amount, planned\_date

FROM Expense, Planned\_Expense

WHERE

Expense.expense\_id = Planned\_Expense.expense\_id

AND MONTH(planned\_date) = MONTH(CURDATE() + INTERVAL 1 MONTH)

AND YEAR(planned\_date) = YEAR(CURDATE() + INTERVAL 1 MONTH);

1. Show total amount spent per month on each category for the previous year:

SELECT exp\_category\_name, MONTHNAME(date\_of\_expense), SUM(amount\_expense) AS total\_spent

FROM Expense\_Category,Expense

WHERE Expense\_Category.category\_id = Expense.category\_id AND YEAR(date\_of\_expense) = YEAR(CURDATE() - INTERVAL 1 YEAR)

GROUP BY exp\_category\_name, MONTH(date\_of\_expense)

ORDER BY exp\_category\_name, MONTH(date\_of\_expense);

1. Change an amount budgeted for a particular expense:

UPDATE Budget

SET budgeted\_amount = /\*Specified Amount\*/

WHERE budget\_id = (SELECT budget\_id FROM Expense

WHERE expense\_id = /\*Specified Expense\*/);

1. Add a new budget expectation for next month based on previous month's expenses

4.1 Getting the list of expenses:

SELECT exp\_category\_name, SUM(amount\_expense) AS total\_spent\_last\_month

FROM Expense\_Category, Expense,Budget,User

WHERE budget.user\_id = /\* Specified User \*/ AND

User.user\_id = budget.user\_id AND

Expense.budget\_id = budget.budget\_id AND

Expense\_Category.category\_id = Expense.category\_id AND

YEAR(date\_of\_expsense) = YEAR(CURDATE() - INTERVAL 1 MONTH) AND

MONTH(date\_of\_expsense) = MONTH(CURDATE() - INTERVAL 1 MONTH)

GROUP BY exp\_category\_name;

4.2 Inserting it into our database:

INSERT INTO Budget (user\_id, budgeted\_amount, start\_date\_budget, end\_date\_budget, expected\_budget)

VALUES (user\_id, total\_spent\_last\_month, CURDATE() + INTERVAL 1 MONTH, CURDATE() + INTERVAL 2 MONTH, 'Planned');

INSERT INTO Expense (budget\_id,category\_id,payment\_method\_id,name\_of\_expense,date\_of\_expense,amount\_expense,recurring)

VALUES(budget\_id, category\_id, payment\_method\_id, name\_of\_expense, CURDATE() + INTERVAL 1 MONTH, total\_spent\_last\_month, recurring)

1. Remove a category from the database:

DELETE FROM Expense\_Category

WHERE category\_id = /\*Desired Category ID to Remove\*/;

6. Extra Queries

1. Change username of a User:

UPDATE User

SET username = /\*Desired Username\*/

WHERE user\_id = /\*Desired UserID\*/;

1. For a specified user, budget list for recent year:

SELECT budget\_id ,budgeted\_amount

FROM Budget

WHERE Budget.user\_id = /\* Selected user \*/

AND DATEDIFF(CURDATE(),start\_date\_budget) <= 365;

1. List all the incomes per specific user per specific amount greater or equal than other:

SELECT income\_id,source\_name,amount\_received

FROM Income, Budget, User

WHERE Income.budget\_id = Budget.budget\_id AND Budget.user\_id = User.user\_id AND User.user\_id = /\*Specified User\*/ AND Income.amount\_received >= /\* Specified Amount \*/

ORDER BY amount\_received;