Joe Wemyss 20068336

Joewemyss3@gmail.com

Abstract

This is a database designed to be the backbone of a home financial planner system. This document serves as the design document for said database.

Home Financial Planner Database

SSD3 Advanced Databases 2017

Contents

[Introduction 3](#_Toc476061148)

[ER Diagram 3](#_Toc476061149)

[EER Diagram 4](#_Toc476061150)

[Food Section 5](#_Toc476061151)

[Finance Section 5](#_Toc476061152)

[Table Design/EER Description 6](#_Toc476061153)

[Relationships 6](#_Toc476061154)

[External 6](#_Toc476061155)

[Food 6](#_Toc476061156)

[Finance 7](#_Toc476061157)

[Table Specifications 8](#_Toc476061158)

[User Group 8](#_Toc476061159)

[User 8](#_Toc476061160)

[Supplier 9](#_Toc476061161)

[Food Category 9](#_Toc476061162)

[Food Type 10](#_Toc476061163)

[Food Type Units 10](#_Toc476061164)

[Current Food Stocks 11](#_Toc476061165)

[Recipe 11](#_Toc476061166)

[Recipe Ingredient 11](#_Toc476061167)

[Food Transaction 12](#_Toc476061168)

[Food Purchase 12](#_Toc476061169)

[Income 13](#_Toc476061170)

[Recurring Income 13](#_Toc476061171)

[Unplanned Expenditure 14](#_Toc476061172)

[Planned Expenditure 14](#_Toc476061173)

[Recurring Expenditure 15](#_Toc476061174)

[Data Entry 15](#_Toc476061175)

[Commonly used Queries 18](#_Toc476061176)

[Select total cost of all food transactions by a single user. 18](#_Toc476061177)

[Spending by User and User Group 19](#_Toc476061178)

[Purchases by User 20](#_Toc476061179)

[Total Spent per Food Item 20](#_Toc476061180)

[Total Non-Food Expenditure per Supplier 21](#_Toc476061181)

[Total Food Expenditure per Supplier 21](#_Toc476061182)

[Total Recurring Expenditure Grouped by Supplier 21](#_Toc476061183)

[Total Annual cost of all monthly bills 22](#_Toc476061184)

[Total Annual Income per user 22](#_Toc476061185)

[Total Annual Income per User Group 22](#_Toc476061186)

[Total Annual Balance per User Group 23](#_Toc476061187)

# Introduction

This database is designed for my Advanced Databases module of Software Systems Development year 3. For this project, I built a database using MySQL. The database is for a home financial planner system and is heavily focused on powerful queries. It will be used to store data mainly on financial transactions, and the amount of food in the house.

The database will be split into two logical areas. These areas will represent the two logical functional areas of the database, Finance and Food. There will also be a third logical fragment that will be used as an external data store. The bulk of the data will be stored on site in the two main segments. Some data will need to be stored externally for external transactions.

The idea is that this database would serve as a master data record for several application systems. For this reason, several tables will be written to very heavily, while most tables will experience heavy read loads. This means that the majority of tables may make liberal use of indices to speed up query time and data aggregation/consolidation, but some tables will need to use the minimum of indices

For demonstration, the data that was put into the database was done so in the context of two couples living in the same house. This is to demonstrate how the data is segregated between user groups. With the dummy data, there are two groups of two users. Transactions processed for either member of a user group can be processed for the entirety of the group, or aggregate data can be gathered for the system as a whole.

This design means that the data that needs to be stored on the external fragment of the database can be easily coerced into NoSQL format. If the data were in a NoSQL format, it could be hosted on Firebases Real-time Database for data synchronisation.

# ER Diagram

The first step in creating this database was to create an ER diagram to figure out the main entities. From this step, I deduced that there would be eight main entities that would need to be recorded. These entities would be:

* Users
* User Groups
* Suppliers
* Incomes
* Expenditures
* Food Types
* Food Categories
* Recipes

There will also need to be tables to store units of measurement for food, as well as tables to record transactions. There will also be several secondary tables for managing foreign key constraints.

During this process, I also designed the structure for a NoSQL segment for external data store for the reasons mentioned. This would simply be a JSON object using a combination of table names and record primary keys as keys that could be easily adapted to a NoSQL database. The structure would be like this:

* <USER\_GROUP\_ID>
  + <USER\_ID>
    - <INCOME>
      * <INCOME\_TRANSACTION\_1>
      * <INCOME\_TRANSACTION\_2>
    - <EXPENDITURE>
      * <FOOD\_EXPENDITURE>
        + <FOOD\_TRANSACTION\_1>
      * <OTHER>
        + <EXPENDITURE\_1>

# EER Diagram

Once the ER diagram was finished, I began the EER diagram. It was clear from the beginning that transactions concerning food should be recorded in their own tables. This was because there would be far more data required from food related transactions than for generic expenses. This resulted in the decision to create the *food\_purchase* table for recording the purchase of a single grocery item., and the *food\_tran*s*action* table to record a full grocery shop.

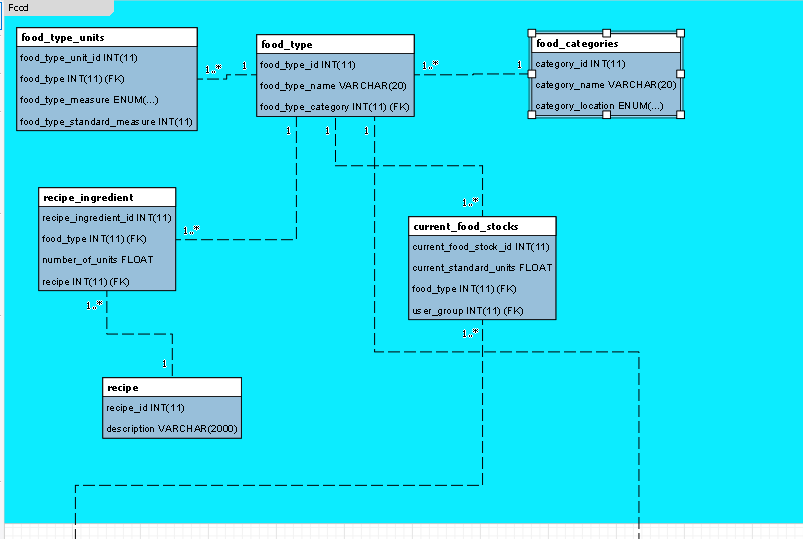
It was also clear that there would need to be a hierarchy in place for incomes and expenditures. This would be to track recurring incomes/expenditures and to differentiate between planned/unplanned expenditures. This resulted in the *income, recurring\_income, unplanned\_expenditure, planned\_expenditure* and *recurring\_expenditure* tables. These tables are designed in a SQL interpretation of OOP’s inheritance principal. In this case, *recurring\_income* will inherit from (hold an optional one-to-one relationship with) with *income*, since a recurring income will only represent an income that specifies how often that income will occur. The same is true of the expenditure, which inherits thusly: *unplanned\_expenditure -> planned\_expenditure -> recurring\_expenditure.* This is because an unplanned expenditure is will not have a set date in the future, whereas a planned expenditure will. The recurring expenditure will have both a date and an interval.

There would also be a requirement to store the current amount of food in the house. For this to be done, there will need to be a measure in place to track units of food. This resulted in the *current\_food\_stocks and* *food\_type\_unit’s* tables. The food type units’ table would be used to track the amount of a specific type of food in the house. The *food\_type\_units* table has a mandatory one-to-one relationship with the *­food\_type* table. For example, Milk would be measured in millilitres and one standard measure of milk would be 1000, or 1 litre. Both the food type and the food type unit’s tables will be experiencing heavy reads, and as such the correct use of indexes should give a performance boost.

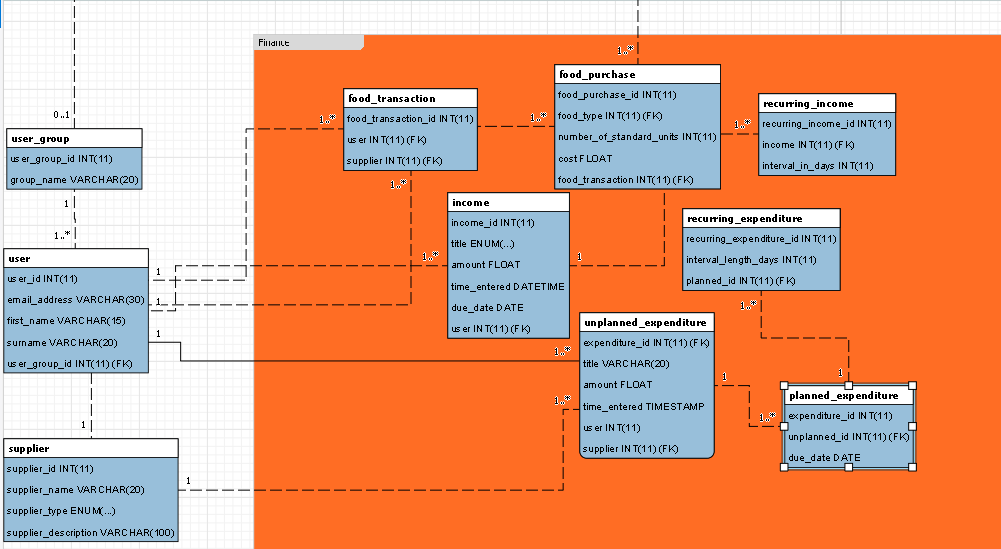
The current food stock table would be used to keep track of the amount of food currently in the house. This table needs to be optimised for high volumes of writes since it will be updated every time a food transaction takes place, or a recipe is used. This means limiting usage of indexes where possible, as this can have adverse performance effects on write-heavy tables.

There will also need to be a recipes table to keep track of recipes that can be used. There will be a one-to-many relationship between the *recipe* and *recipe\_ingredient* tables. The recipe ingredient table will be used to track the amounts of each food type required.

This gave me a total of 16 tables to implement in the database.

Food Section

## Finance Section



# Table Design/EER Description

## Relationships

### External

User Group = {user\_group\_id, group\_name}

|  |  |
| --- | --- |
| **Primary Key** | *user\_group\_id* |
|  | |

User = {user\_id, email\_address, first\_name, surname, user\_group\_id}

|  |  |
| --- | --- |
| **Primary Key** | *user\_id* |
| **Foreign Key** | *user\_group\_id references user\_group.user\_group\_id* |
|  | | |

Supplier = {supplier\_id, supplier\_name, supplier\_type, supplier\_description}

|  |  |
| --- | --- |
| **Primary Key** | *supplier\_id* |
|  | | |

### Food

Food Category = {category\_id, category\_name, category\_location}

|  |  |
| --- | --- |
| **Primary Key** | *category\_id* |
|  | | |

Food Type = {food\_type\_id, food\_type\_name, units\_of\_measurement, standard\_units, food\_type\_category}

|  |  |
| --- | --- |
| **Primary Key** | *food\_type\_id* |
| **Foreign Key** | *food\_type-category references food\_category.category\_id* |
|  | | |

Current Food Stocks = {current\_food\_stock\_id, current\_standard\_units, food\_type, user\_group}

|  |  |
| --- | --- |
| **Primary Key** | *current\_food\_stock\_id* |
| **Foreign Key** | *food\_type references food\_type.food\_type\_id* |
| **Foreign Key** | *user\_group references user\_group.user\_group\_id* |
|  | | |

Recipe = {recipe\_id, description}

|  |  |
| --- | --- |
| **Primary Key** | *recipe\_id* |
|  | | |

Recipe Ingredient = {recipe\_ingredient\_id, food\_type, number\_of\_units, recipe}

|  |  |
| --- | --- |
| **Primary Key** | *recipe\_id* |
| **Foreign Key** | *food\_type references food\_type.food\_type\_id* |
| **Foreign Key** | *recipe references recipe.recipe\_id* |
|  | | |

### Finance

Income = {income\_id, title, amount, time\_entered, due\_date, user}

|  |  |
| --- | --- |
| **Primary Key** | *income\_id* |
| **Foreign Key** | *user references user.user\_id* |
|  | | |

Recurring Income = {recurring\_income\_id, interval\_length\_days}

|  |  |
| --- | --- |
| **Primary Key** | *recurring\_income\_id* |
|  | | |

Unplanned Expenditure = {expenditure\_id, title, amount, time\_entered, user, supplier}

|  |  |
| --- | --- |
| **Primary Key** | *expenditure\_id* |
| **Foreign Key** | *user references user.user\_id* |
| **Foreign Key** | *supplier references supplier.supplier\_id* |
|  | | |

Planned Expenditure = {expenditure\_id, unplanned\_id, due\_date}

|  |  |
| --- | --- |
| **Primary Key** | *expenditure\_id* |
| **Foreign Key** | *unplanned\_id references unplanned\_expenditure.expenditure\_id* |
|  | | |

Recurring Expenditure = {recurring\_expenditure\_id, planned\_id, interval\_length\_days }

|  |  |
| --- | --- |
| **Primary Key** | *recurring\_expenditure\_id* |
| **Foreign Key** | *planned\_id references planned\_expenditure.expenditure\_id* |
|  | | |

Food Transaction = {food\_transaction\_id, user, supplier}

|  |  |
| --- | --- |
| **Primary Key** | *food\_transaction\_id* |
| **Foreign Key** | *user references user.user\_id* |
| **Foreign Key** | *supplier references supplier.supplier\_id* |
|  | | |

Food Purchase = {food\_purchase\_id, food\_type, number\_of\_standard\_units, cost, food\_transaction}

|  |  |
| --- | --- |
| **Primary Key** | *food\_purchase\_id* |
| **Foreign Key** | *food\_type references food\_type.food\_type\_id* |
| **Foreign Key** | *food\_transaction references food\_transaction.food\_transaction\_id* |
|  | | |

## Table Specifications

Below is the full specification for all tables in the database. It is worth noting that some of the longer field names have been truncated for brevity.

### User Group

This table will represent a group of users. This table will not be written to very much, so it makes use a unique index on the group name field, for faster queries using that field.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| user\_group\_id | Integer | 11 | Not-null |  | Auto-Increment | PK | Primary key |
| group\_name | Varchar | 20 | Not-null |  |  |  | The name of the user group |

**CREATE TABLE** `user\_group` (  
 **`user\_group\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`group\_name` varchar**(20) **NOT NULL**,  
 **PRIMARY KEY** (**`user\_group\_id`**),  
 **UNIQUE KEY** `user\_group\_group\_name\_uindex` (**`group\_name`**)  
) **ENGINE**=InnoDB **AUTO\_INCREMENT**=3 **DEFAULT CHARSET**=latin1 **COMMENT**=**'This will represent user groups that will be able to view one another’s transactions'**

### User

This table will represent the users that will be on the system. This is another table that will not be written to much, so it also makes use of indices to ensure that it is optimised for high read traffic. There is a unique index on email address, since this would usually be required to be unique within a system. There is also an index on the first name/surname combination, since this is a combination that will be frequently searched.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| user\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| email\_address | Varchar | 30 | Not-null |  | Unique |  | User email address |
| first\_name | Varchar | 15 | Not-null |  |  |  | User first name |
| surname | Varchar | 20 | Not-null |  |  |  | User surname |
| user\_group\_id | Integer | 11 | Not-null |  |  | FK | User group |

**CREATE TABLE** `user` (  
 **`user\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`email\_address` varchar**(30) **NOT NULL**,  
 **`first\_name` varchar**(15) **NOT NULL**,  
 **`surname` varchar**(20) **NOT NULL**,  
 **`user\_group\_id` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`user\_id`**),  
 **UNIQUE KEY** `user\_email\_address\_uindex` (**`email\_address`**),  
 **KEY** `user\_group\_\_\_fk` (**`user\_group\_id`**),  
 **KEY** `user\_first\_name\_surname\_index` (**`first\_name`**,**`surname`**),  
 **CONSTRAINT** `user\_group\_\_\_fk` **FOREIGN KEY** (**`user\_group\_id`**) **REFERENCES** `user\_group` (**`user\_group\_id`**) **ON DELETE CASCADE ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=5 **DEFAULT CHARSET**=latin1 **COMMENT**=**'This will store all of the users in the database'**

### Supplier

This table represents a supplier of goods or services. This table will be used for correlating transactions to determine where the best deals are. This is another table that will need to be optimised more for reads that writes. For this reason, there is a unique index on the supplier name. This is the field that will be most often queried so the index should help give it a performance boost.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| supplier\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| supplier\_name | Varchar | 20 | Not-null |  | Unique |  | Supplier name |
| supplier\_type | Enum | 4 | Not-null | grocery |  |  | Supplier type |
| supplier\_description | Varchar | 100 |  | Null |  |  | Supplier description |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Supplier Type Enum** |  |  |  |  |  |
| **Values** | grocery | utilities | clothing | entertainment | other |

**CREATE TABLE** `supplier` (  
 **`supplier\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`supplier\_name` varchar**(20) **NOT NULL**,  
 **`supplier\_type` enum**(**'grocery'**,**'utilities'**,**'clothing'**,**'entertainment'**,**'other'**) **NOT NULL DEFAULT 'grocery'**,  
 **`supplier\_description` varchar**(100) **DEFAULT NULL**,  
 **PRIMARY KEY** (**`supplier\_id`**),  
 **UNIQUE KEY** `supplier\_supplier\_name\_uindex` (**`supplier\_name`**),  
 **KEY** `supplier\_supplier\_name\_index` (**`supplier\_name`**)  
) **ENGINE**=InnoDB **AUTO\_INCREMENT**=7 **DEFAULT CHARSET**=latin1

### Food Category

This table will hold details of all the food categories. This will be used to categorise all the foods, for data aggregation. This is another table that will have infrequent writes, therefore, it makes use of a unique index on the name field, as this will most likely be a search parameter.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| category\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| category\_name | Varchar | 20 | Not-null |  | Unique |  | Category name |
| category\_location | Enum | 5 | Not-null | fridge |  |  | Location in kitchen |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Category Name Enum** |  |  |  |  |  |
| **Values** | fridge | freezer | shelf | cupboard | other |

**CREATE TABLE** `food\_categories` (  
 **`category\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`category\_name` varchar**(20) **NOT NULL**,  
 **`category\_location` enum**(**'fridge'**,**'freezer'**,**'shelf'**,**'cupboard'**,**'other'**) **NOT NULL DEFAULT 'fridge'**,  
 **PRIMARY KEY** (**`category\_id`**),  
 **UNIQUE KEY** `food\_categories\_category\_name\_uindex` (**`category\_name`**),  
 **UNIQUE KEY** `food\_categories\_category\_name\_category\_location\_uindex` (**`category\_name`**,**`category\_location`**)  
) **ENGINE**=InnoDB **AUTO\_INCREMENT**=11 **DEFAULT CHARSET**=latin1

### Food Type

This table will hold details of all the food types that have ever been purchased. This will be another table that should have a significantly higher volume of reads than writes. This is a reason why this table makes use of a unique index on the name field of the table.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| food\_type\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| food\_type\_name | Varchar | 20 | Not-null |  | Unique |  | Food type name |
| standard\_measure | Integer | 10 | Not-null | 1 | Unsigned |  | The standard measure |
| unit\_of\_measurement | Enum | 4 | Not-null | gram |  |  | Unit of measurement |
| food\_type\_category | Integer | 11 | Not-null |  |  | FK | Category |

**CREATE TABLE** `food\_type` (  
 **`food\_type\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`food\_type\_name` varchar**(20) **NOT NULL**,  
 **`standard\_measure` int**(10) **unsigned NOT NULL DEFAULT '1'**,  
 **`unit\_of\_measurement` enum**(**'unit'**,**'gram'**,**'milliliter'**,**'other'**) **NOT NULL DEFAULT 'gram'**,  
 **`food\_type\_category` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`food\_type\_id`**),  
 **UNIQUE KEY** `food\_type\_food\_type\_name\_uindex` (**`food\_type\_name`**),  
 **KEY** `food\_type\_category\_\_\_fk` (**`food\_type\_category`**),  
 **CONSTRAINT** `food\_type\_category\_\_\_fk` **FOREIGN KEY** (**`food\_type\_category`**) **REFERENCES** `food\_categories` (**`category\_id`**) **ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=28 **DEFAULT CHARSET**=latin1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unit of Measurement Enum** | |  |  |  |
| **Values** | unit | gram | millilitre | other |

### Current Food Stocks

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| current\_food\_stock\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| current\_standard\_units | Float | (6,3) | Not-null |  | Unsigned |  | Standard units |
| food\_type | Integer | 11 | Not-null |  |  | FK | Food Type |
| user\_group | Integer | 11 | Not-null |  |  | FK | User group |

**CREATE TABLE** `current\_food\_stocks` (  
 **`current\_food\_stock\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`current\_standard\_units` float NOT NULL**,  
 **`food\_type` int**(11) **NOT NULL**,  
 **`user\_group` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`current\_food\_stock\_id`**),  
 **KEY** `current\_food\_stocks\_type\_\_\_fk` (**`food\_type`**),  
 **KEY** `current\_food\_stocks\_user\_group\_user\_group\_id\_fk` (**`user\_group`**),  
 **CONSTRAINT** `current\_food\_stocks\_type\_\_\_fk` **FOREIGN KEY** (**`food\_type`**) **REFERENCES** `food\_type` (**`food\_type\_id`**) **ON UPDATE CASCADE**,  
 **CONSTRAINT** `current\_food\_stocks\_user\_group\_user\_group\_id\_fk` **FOREIGN KEY** (**`user\_group`**) **REFERENCES** `user\_group` (**`user\_group\_id`**) **ON DELETE CASCADE ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=10 **DEFAULT CHARSET**=latin1 **COMMENT**=**'This will store all of the current food stocks in the house. This table will need to be optimised for frequent read/writes'**

### Recipe

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| recipe\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| recipe\_name | Varchar | 40 | Not-null |  | Unique |  | Recipe Name |
| description | Varchar | 2000 | Not-null |  |  |  | Recipe description |

**CREATE TABLE** `recipe` (  
 **`recipe\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`recipe\_name` varchar**(40) **NOT NULL**,  
 **`description` varchar**(767) **NOT NULL**,  
 **PRIMARY KEY** (**`recipe\_id`**),  
 **UNIQUE KEY** `recipe\_description\_uindex` (**`description`**)  
) **ENGINE**=InnoDB **AUTO\_INCREMENT**=4 **DEFAULT CHARSET**=latin1

### Recipe Ingredient

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| recipe\_ingredient\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| number\_of\_units | Float | (6, 3) | Not-null |  | Unsigned |  | Number of units |
| food\_type | Integer | 11 | Not-null |  |  | FK | Food Type |
| recipe | Integer | 11 | Not-null |  |  | FK | Recipe |

**CREATE TABLE** `recipe\_ingredient` (  
 **`recipe\_ingredient\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`number\_of\_units` float**(6,3) **unsigned NOT NULL**,  
 **`food\_type` int**(11) **NOT NULL**,  
 **`recipe` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`recipe\_ingredient\_id`**),  
 **KEY** `recipe\_ingedient\_food\_type\_food\_type\_id\_fk` (**`food\_type`**),  
 **KEY** `recipe\_ingredient\_recipe\_recipe\_id\_fk` (**`recipe`**),  
 **CONSTRAINT** `recipe\_ingedient\_food\_type\_food\_type\_id\_fk` **FOREIGN KEY** (**`food\_type`**) **REFERENCES** `food\_type` (**`food\_type\_id`**) **ON UPDATE CASCADE**,  
 **CONSTRAINT** `recipe\_ingredient\_recipe\_recipe\_id\_fk` **FOREIGN KEY** (**`recipe`**) **REFERENCES** `recipe` (**`recipe\_id`**) **ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=13 **DEFAULT CHARSET**=latin1

### Food Transaction

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| food\_transaction\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| transaction\_time | Timestamp | 4 | Not-null | NOW() |  |  | Timestamp |
| user | Integer | 11 | Not-null |  |  | FK | User |
| supplier | Integer | 11 | Not-null |  |  | FK | Supplier |

**CREATE TABLE** `food\_transaction` (  
 **`food\_transaction\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`transaction\_time` timestamp NOT NULL DEFAULT** *CURRENT\_TIMESTAMP*,  
 **`user` int**(11) **NOT NULL**,  
 **`supplier` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`food\_transaction\_id`**),  
 **KEY** `food\_transaction\_user\_user\_id\_fk` (**`user`**),  
 **KEY** `food\_transaction\_suppliers\_supplier\_fk` (**`supplier`**),  
 **CONSTRAINT** `food\_transaction\_suppliers\_supplier\_fk` **FOREIGN KEY** (**`supplier`**) **REFERENCES** `supplier` (**`supplier\_id`**) **ON UPDATE CASCADE**,  
 **CONSTRAINT** `food\_transaction\_user\_user\_id\_fk` **FOREIGN KEY** (**`user`**) **REFERENCES** `user` (**`user\_id`**) **ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=5 **DEFAULT CHARSET**=latin1 **COMMENT**=**'This will be a list of items purchased together, in a single transaction.'**

### Food Purchase

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| food\_purchase\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| standard\_units | Integer | 11 | Not-null | 1 | Unsigned |  | Number of units |
| cost | Float | (5,2) | Not-null |  | Unsigned |  | Cost |
| food\_type | Integer | 11 | Not-null |  |  | FK | Food type |
| food\_transaction | Integer | 11 | Not-null |  |  | FK | Transaction |

**CREATE TABLE** `food\_purchase` (  
 **`food\_purchase\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 `number\_of\_standard\_units` **int**(11) **unsigned NOT NULL DEFAULT '1'**,  
 **`cost` float**(5,2) **unsigned NOT NULL**,  
 **`food\_type` int**(11) **NOT NULL**,  
 **`food\_transaction` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`food\_purchase\_id`**),  
 **KEY** `food\_purchase\_food\_type\_food\_type\_id\_fk` (**`food\_type`**),  
 **KEY** `food\_purchase\_transaction\_\_fk` (**`food\_transaction`**),  
 **CONSTRAINT** `food\_purchase\_food\_type\_food\_type\_id\_fk` **FOREIGN KEY** (**`food\_type`**) **REFERENCES** `food\_type` (**`food\_type\_id`**) **ON UPDATE CASCADE**,  
 **CONSTRAINT** `food\_purchase\_transaction\_\_fk` **FOREIGN KEY** (**`food\_transaction`**) **REFERENCES** `food\_transaction` (**`food\_transaction\_id`**) **ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=11 **DEFAULT CHARSET**=latin1 **COMMENT**=**'This will be a purchase of a single item.'**

### Income

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| income\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| title | Enum | 5 | Not-null | wages |  |  | Title |
| amount | Float | (5,2) | Not-null |  | Unsigned |  | Amount |
| time\_entered | Timestamp | 4 | Not-null | NOW() |  |  | Time entered |
| due\_date | Date | 3 | Not-null |  |  |  | Due date |
| user | Integer | 11 | Not-null |  |  | FK | User |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Title Enum** | |  |  |  |  |
| **Values** | wages | child benefit | dividends | freelance | other |

**CREATE TABLE** `income` (  
 **`income\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`title` enum**(**'wages'**,**'child benefit'**,**'dividends'**,**'freelance'**,**'other'**) **NOT NULL DEFAULT 'wages'**,  
 **`amount` float**(5,2) **unsigned NOT NULL**,  
 **`time\_entered` timestamp NOT NULL DEFAULT** *CURRENT\_TIMESTAMP*,  
 **`due\_date` date NOT NULL**,  
 **`user` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`income\_id`**),  
 **KEY** `income\_user\_\_\_fk` (**`user`**),  
 **CONSTRAINT** `income\_user\_\_\_fk` **FOREIGN KEY** (**`user`**) **REFERENCES** `user` (**`user\_id`**) **ON DELETE CASCADE ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=18 **DEFAULT CHARSET**=latin1 **COMMENT**=**'This is the table to track the incomes that will be stored in the database'**

### Recurring Income

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| recurring\_income\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| interval\_in\_days | Integer | 11 | Not-null | 7 | Unsigned |  | Interval |
| income | Integer | 11 | Not-null |  |  | FK | Income |

**CREATE TABLE** `recurring\_income` (  
 **`recurring\_income\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`interval\_in\_days` int**(11) **unsigned NOT NULL DEFAULT '7'**,  
 **`income` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`recurring\_income\_id`**),  
 **KEY** `recurring\_income\_income\_income\_id\_fk` (**`income`**),  
 **CONSTRAINT** `recurring\_income\_income\_income\_id\_fk` **FOREIGN KEY** (**`income`**) **REFERENCES** `income` (**`income\_id`**) **ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=9 **DEFAULT CHARSET**=latin1

### Unplanned Expenditure

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| expenditure\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| title | Varchar | 20 | Not-null | miscellaneous |  |  | Title |
| time\_entered | Timestamp | 4 | Not-null | NOW() |  |  | Time entered |
| amount | Float | (5,2) | Not-null |  | Unsigned |  | Amount |
| user | Integer | 11 | Not-null |  |  | FK | User |
| supplier | Integer | 11 | Not-null |  |  | FK | Supplier |

**CREATE TABLE** `unplanned\_expenditure` (  
 **`expenditure\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`title` varchar**(20) **NOT NULL DEFAULT 'miscellaneous'**,  
 **`time\_entered` timestamp NOT NULL DEFAULT** *CURRENT\_TIMESTAMP*,  
 **`amount` float unsigned NOT NULL**,  
 **`user` int**(11) **NOT NULL**,  
 **`supplier` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`expenditure\_id`**),  
 **KEY** `expenditure\_suppliers\_supplier\_fk` (**`supplier`**),  
 **CONSTRAINT** `expenditure\_suppliers\_supplier\_fk` **FOREIGN KEY** (**`supplier`**) **REFERENCES** `supplier` (**`supplier\_id`**) **ON UPDATE CASCADE**,  
 **CONSTRAINT** `expenditure\_user\_\_\_fk` **FOREIGN KEY** (**`expenditure\_id`**) **REFERENCES** `user` (**`user\_id`**) **ON DELETE CASCADE ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=5 **DEFAULT CHARSET**=latin1

### Planned Expenditure

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| expenditure\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| due\_date | Date | 3 | Not-null |  |  |  | Due Date |
| unplanned\_id | Integer | 11 | Not-null |  |  | FK | Unplanned record |

**CREATE TABLE** `planned\_expenditure` (  
 **`expenditure\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`due\_date` date NOT NULL**,  
 **`unplanned\_id` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`expenditure\_id`**),  
 **KEY** `planned\_expenditure\_unplanned\_expenditure\_expenditure\_id\_fk` (**`unplanned\_id`**),  
 **CONSTRAINT** `planned\_expenditure\_unplanned\_expenditure\_expenditure\_id\_fk` **FOREIGN KEY** (**`unplanned\_id`**) **REFERENCES** `unplanned\_expenditure` (**`expenditure\_id`**) **ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=3 **DEFAULT CHARSET**=latin1

### Recurring Expenditure

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Size** | **Null/Not-null** | **Default** | **Constraints** | **Index** | **Description** |
| expenditure\_id | Integer | 11 | Not-null |  | Auto\_increment | PK | Primary key |
| interval | Integer | 11 | Not-null | 30 | Unsigned |  | Interval |
| planned\_id | Integer | 11 | Not-null |  |  | FK | Planned record |

**CREATE TABLE** `recurring\_expenditure` (  
 **`recurring\_expenditure\_id` int**(11) **NOT NULL AUTO\_INCREMENT**,  
 **`interval\_days` int**(10) **unsigned NOT NULL DEFAULT '30'**,  
 **`planned\_id` int**(11) **NOT NULL**,  
 **PRIMARY KEY** (**`recurring\_expenditure\_id`**),  
 **KEY** `recurring\_planned\_expenditure\_\_fk` (**`planned\_id`**),  
 **CONSTRAINT** `recurring\_planned\_expenditure\_\_fk` **FOREIGN KEY** (**`planned\_id`**) **REFERENCES** `planned\_expenditure` (**`expenditure\_id`**) **ON UPDATE CASCADE**) **ENGINE**=InnoDB **AUTO\_INCREMENT**=3 **DEFAULT CHARSET**=latin1

# Data Entry

This section contains the full set of SQL Insert statements, in order of required execution, required to populate the database with test data.

**INSERT INTO** advanced\_databases\_project.user\_group (**group\_name**) **VALUES** (**'Power'**);  
**INSERT INTO** advanced\_databases\_project.user\_group (**group\_name**) **VALUES** (**'Wemyss'**);

**INSERT INTO** advanced\_databases\_project.user (**email\_address**, **first\_name**, **surname**, user\_group\_id) **VALUES** (**'joewemyss@gmail.com'**, **'joe'**, **'wemyss'**, 1);  
**INSERT INTO** advanced\_databases\_project.user (**email\_address**, **first\_name**, **surname**, user\_group\_id) **VALUES** (**'marywemyss@gmail.com'**, **'mary'**, **'wemyss'**, 1);  
**INSERT INTO** advanced\_databases\_project.user (**email\_address**, **first\_name**, **surname**, user\_group\_id) **VALUES** (**'michellepower@gmail.com'**, **'michelle'**, **'power'**, 2);  
**INSERT INTO** advanced\_databases\_project.user (**email\_address**, **first\_name**, **surname**, user\_group\_id) **VALUES** (**'johnpower@gmail.com'**, **'john'**, **'power'**, 2);

**INSERT INTO** advanced\_databases\_project.supplier (**supplier\_name**, **supplier\_type**, **supplier\_description**) **VALUES** (**'aldi'**, **'grocery'**, **'shop for groceries'**);  
**INSERT INTO** advanced\_databases\_project.supplier (**supplier\_name**, **supplier\_type**, **supplier\_description**) **VALUES** (**'tesco'**, **'grocery'**, **'shop for groceries'**);  
**INSERT INTO** advanced\_databases\_project.supplier (**supplier\_name**, **supplier\_type**, **supplier\_description**) **VALUES** (**'pre pay power'**, **'utilities'**, **'electricity provider'**);  
**INSERT INTO** advanced\_databases\_project.supplier (**supplier\_name**, **supplier\_type**, **supplier\_description**) **VALUES** (**'virgin media'**, **'utilities'**, **'internet provider'**);  
**INSERT INTO** advanced\_databases\_project.supplier (**supplier\_name**, **supplier\_type**, **supplier\_description**) **VALUES** (**'netflix'**, **'entertainment'**, **'entertainment provider'**);  
**INSERT INTO** advanced\_databases\_project.supplier (**supplier\_name**, **supplier\_type**, **supplier\_description**) **VALUES** (**'jack and jones'**, **'clothing'**, **'clothing provider'**);

**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'wages'**, 500, **'2017-02-26 20:26:06'**, **'2017-04-01'**, 1);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'dividends'**, 1250.34, **'2017-02-26 20:26:07'**, **'2017-03-28'**, 1);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'wages'**, 300, **'2017-02-26 20:26:07'**, **'2017-04-03'**, 2);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'child benefit'**, 160, **'2017-02-26 20:26:07'**, **'2017-03-28'**, 2);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'wages'**, 700, **'2017-02-26 20:26:07'**, **'2017-03-27'**, 3);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'freelance'**, 250.34, **'2017-02-26 20:26:07'**, **'2017-04-05'**, 3);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'wages'**, 500, **'2017-02-26 20:26:07'**, **'2017-04-03'**, 4);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'child benefit'**, 160, **'2017-02-26 20:26:07'**, **'2017-03-28'**, 4);  
**INSERT INTO** advanced\_databases\_project.income (**title**, **amount**, **time\_entered**, **due\_date**, **user**) **VALUES** (**'wages'**, 1260, **'2017-02-26 22:17:14'**, **'2017-04-28'**, 1);

**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (7, 9);  
**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (365, 10);  
**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (7, 11);  
**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (7, 12);  
**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (30, 13);  
**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (7, 15);  
**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (30, 16);  
**INSERT INTO** advanced\_databases\_project.recurring\_income (**interval\_in\_days**, **income**) **VALUES** (30, 17);

**INSERT INTO** advanced\_databases\_project.unplanned\_expenditure (**title**, **time\_entered**, **amount**, **user**, **supplier**) **VALUES** (**'Prepay power'**, **'2017-02-26 20:42:57'**, 20, 1, 3);  
**INSERT INTO** advanced\_databases\_project.unplanned\_expenditure (**title**, **time\_entered**, **amount**, **user**, **supplier**) **VALUES** (**'internet bill'**, **'2017-02-26 20:42:57'**, 75, 2, 4);  
**INSERT INTO** advanced\_databases\_project.unplanned\_expenditure (**title**, **time\_entered**, **amount**, **user**, **supplier**) **VALUES** (**'new clothes'**, **'2017-02-26 20:42:57'**, 100, 3, 6);  
**INSERT INTO** advanced\_databases\_project.unplanned\_expenditure (**title**, **time\_entered**, **amount**, **user**, **supplier**) **VALUES** (**'netflix'**, **'2017-02-26 20:42:57'**, 9, 4, 5);

**INSERT INTO** advanced\_databases\_project.planned\_expenditure (**due\_date**, **unplanned\_id**) **VALUES** (**'2017-03-02'**, 2);  
**INSERT INTO** advanced\_databases\_project.planned\_expenditure (**due\_date**, **unplanned\_id**) **VALUES** (**'2017-03-02'**, 4);

**INSERT INTO** advanced\_databases\_project.recurring\_expenditure (**interval\_days**, **planned\_id**) **VALUES** (30, 1);  
**INSERT INTO** advanced\_databases\_project.recurring\_expenditure (**interval\_days**, **planned\_id**) **VALUES** (30, 2);

**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'cereals'**, **'cupboard'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'chilled vegetables'**, **'fridge'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'cold meats'**, **'fridge'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'dairy'**, **'fridge'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'dry goods'**, **'cupboard'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'frozen desserts'**, **'freezer'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'fruits'**, **'shelf'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'hot drinks'**, **'cupboard'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'raw meats'**, **'fridge'**);  
**INSERT INTO** advanced\_databases\_project.food\_categories (**category\_name**, **category\_location**) **VALUES** (**'vegetables'**, **'cupboard'**);

**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'milk'**, 250, **'milliliter'**, 5);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'coffee'**, 100, **'gram'**, 8);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'ham'**, 1, **'unit'**, 3);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'carrots'**, 1, **'unit'**, 7);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'potatoes'**, 500, **'gram'**, 7);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'spring onions'**, 1, **'unit'**, 6);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'vienetta'**, 1, **'unit'**, 2);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'frosties'**, 250, **'gram'**, 1);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'cornflakes'**, 250, **'gram'**, 1);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'butter'**, 225, **'gram'**, 5);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'bread'**, 1, **'unit'**, 9);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'cheese'**, 100, **'gram'**, 5);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'steak'**, 1, **'unit'**, 10);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'mushrooms'**, 50, **'gram'**, 6);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'onions'**, 1, **'unit'**, 7);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'chicken breast'**, 1, **'unit'**, 10);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'flour'**, 500, **'gram'**, 9);  
**INSERT INTO** advanced\_databases\_project.food\_type (food\_type\_name, **standard\_measure**, **unit\_of\_measurement**, **food\_type\_category**) **VALUES** (**'pastry cases'**, 1, **'unit'**, 5);

**INSERT INTO** advanced\_databases\_project.recipe (**recipe\_name**, **description**) **VALUES** (**'ham and cheese sandwich'**, **'ham and cheese sandwich'**);  
**INSERT INTO** advanced\_databases\_project.recipe (**recipe\_name**, **description**) **VALUES** (**'steak dinner'**, **'steak dinner'**);  
**INSERT INTO** advanced\_databases\_project.recipe (**recipe\_name**, **description**) **VALUES** (**'vol au vents'**, **'vol au vents'**);

**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (0.1, 20, 1);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (0.01, 19, 1);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (0.2, 21, 1);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (2, 22, 2);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (0.1, 14, 2);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (0.33, 24, 2);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (1, 23, 2);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (1, 27, 3);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (3, 25, 3);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (0.01, 26, 3);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (1, 23, 3);  
**INSERT INTO** advanced\_databases\_project.recipe\_ingredient (**number\_of\_units**, **recipe\_food\_type**, **recipe**) **VALUES** (0.01, 19, 3);

**INSERT INTO** advanced\_databases\_project.food\_transaction (**transaction\_time**, **user**, **supplier**) **VALUES** (**'2017-02-27 17:02:23'**, 1, 1);  
**INSERT INTO** advanced\_databases\_project.food\_transaction (**transaction\_time**, **user**, **supplier**) **VALUES** (**'2017-02-27 17:02:23'**, 2, 2);  
**INSERT INTO** advanced\_databases\_project.food\_transaction (**transaction\_time**, **user**, **supplier**) **VALUES** (**'2017-02-27 17:02:23'**, 3, 2);  
**INSERT INTO** advanced\_databases\_project.food\_transaction (**transaction\_time**, **user**, **supplier**) **VALUES** (**'2017-02-27 17:02:23'**, 4, 1);

**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (2, 2.19, 19, 1);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (2, 1.49, 10, 1);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (1, 2, 12, 1);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (1, 0.99, 15, 1);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (1, 0.87, 13, 2);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (2, 0.99, 14, 2);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (1, 0.99, 10, 3);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (2, 2.2, 17, 3);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (3, 2, 10, 4);  
**INSERT INTO** advanced\_databases\_project.food\_purchase (**number\_of\_standard\_units**, **cost**, **food\_type**, **food\_transaction**) **VALUES** (1, 2.19, 11, 4);

# Views

For the purposes of demonstration, there will be two logical views. One view will represent the details pertaining to food while the other will focus on the fiscal aspect of the database. These views are representative of how the database would be partitioned in a distributed database landscape.

It is worth noting that in a database this size, using these views would cause significant performance overhead, as all the fields required by users of that view would need to be left joined to ensure that all fields are available for any queries that would be performed on the view. Since views are not persisted, this means that the database must effectively pull in the entire result set for the view every time a query is made on that view.

There would also be a third logical partition. This partition would be used for data replication. The third partition would be built using a NoSQL technology, such as MongoDB. This partition would sit on an external server so that data gathered outside the internal network where the main database resides could be replicated on said database. An example of this would be the Firebase Real-Time Database, which provides an API for listening for changes in the external database.

## Food View

This view will contain the details pertaining to food. This view is designed to be used by application systems that only need access to the food section of the database, such as recipe planners. The SQL code to create the view is as follows:

**CREATE VIEW** food\_view **AS  
 SELECT** food\_type.**food\_type\_name**, food\_type.**standard\_measure**, food\_type.**unit\_of\_measurement**, food\_categories.**category\_name**,  
 food\_categories.**category\_location**, recipe.**recipe\_name**, recipe\_ingredient.**number\_of\_units**,  
 current\_food\_stocks.**current\_standard\_units  
 FROM** food\_categories  
 **LEFT JOIN** food\_type **ON** food\_categories.**category\_id** = food\_type.**food\_type\_category  
 LEFT JOIN** current\_food\_stocks **ON** food\_type.**food\_type\_id** = current\_food\_stocks.**food\_type  
 LEFT JOIN** recipe\_ingredient **ON** food\_type.**food\_type\_id** = recipe\_ingredient.**recipe\_food\_type  
 LEFT JOIN** recipe **ON** recipe\_ingredient.**recipe** = recipe.**recipe\_id**;

## Food Transaction View

This view will not be queried directly, but will act as a go-between for the food and transaction views. This view will gather all of the food purchases, group them by user and then calculate the total spent by each user. This data can then be queried by either the Food View or the Transaction View.

**CREATE VIEW** food\_transaction\_view **AS  
 SELECT user**.**user\_group\_id AS 'User Group'**, **user**.**user\_id AS 'User ID'**, *CONCAT*(**user**.**first\_name**, **' '**, **user**.**surname**) **AS 'Username'**,  
 *SUM*(food\_purchase.**cost**) **AS 'Total Food Cost'  
 FROM user  
 JOIN** food\_transaction **ON user**.**user\_id** = food\_transaction.**user  
 JOIN** food\_purchase **ON** food\_transaction.**food\_transaction\_id** = food\_purchase.**food\_transaction GROUP BY user**.**user\_id**;

## Transaction View

This view will contain all the details pertaining to transactions. This view will be used primarily by financial tracker applications that do not need access to the details about food. The SQL code to create this view is below.

**CREATE VIEW** transaction\_view **AS  
 SELECT** `Username`, `Total Food Cost`, user\_group.**group\_name**,  
 income.**title AS 'Income Title'**, income.**amount AS 'Income Amount'**, income.**due\_date AS 'Income Due'**,  
 recurring\_income.**interval\_in\_days AS 'Income Interval'**, unplanned\_expenditure.**title as 'Expenditure Title'**,  
 unplanned\_expenditure.**amount AS 'Expenditure Amount'**, planned\_expenditure.**due\_date AS 'Expenditure Due'**,  
 recurring\_expenditure.**interval\_days AS 'Expenditure Interval'**, supplier.**supplier\_name**, supplier.**supplier\_type**, supplier.**supplier\_description  
 FROM** food\_transaction\_view  
 **LEFT JOIN** user\_group **ON** food\_transaction\_view.`User Group` = user\_group.**user\_group\_id  
 LEFT JOIN** food\_transaction **ON** food\_transaction\_view.`User ID` = food\_transaction.**user  
 LEFT JOIN** food\_purchase **ON** food\_transaction.**food\_transaction\_id** = food\_purchase.**food\_transaction  
 LEFT JOIN** income **ON** food\_transaction\_view.`User ID` = income.**user  
 LEFT JOIN** recurring\_income **ON** income.**income\_id** = recurring\_income.**income  
 LEFT JOIN** unplanned\_expenditure **ON** food\_transaction\_view.`User ID` = unplanned\_expenditure.**expenditure\_id  
 LEFT JOIN** planned\_expenditure **ON** unplanned\_expenditure.**expenditure\_id** = planned\_expenditure.**unplanned\_id  
 LEFT JOIN** recurring\_expenditure **ON** planned\_expenditure.**expenditure\_id** = recurring\_expenditure.**planned\_id  
 LEFT JOIN** supplier **ON** food\_transaction.**supplier** = supplier.**supplier\_id**;

# Commonly used Queries

## Cross-View

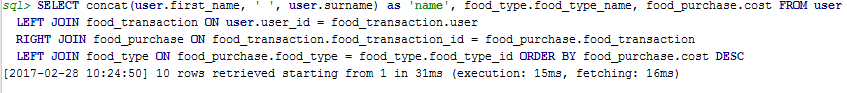
The queries listed below are queries that will need access to both views to run. As such, these queries are run on the entire database, rather than on a single view. There are also some of the more complex queries, which I did not have time to coerce into using views.

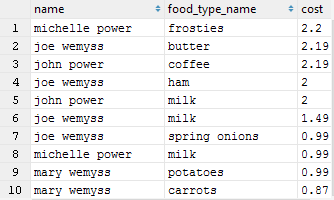
### Purchases by User

This query will return the details of all purchases made. The result set will contain the users name, the name of the food type, and the cost of the purchase. This query is not likely to be used often, since it will return every purchase in the purchase table. This is a table that will possibly grow quite large, so a O(n) data retrieval would not be desirable.

This query belongs in the cross view section, as it is required that both the name of the food, and the name of the person who bought it be included in the result set, neither of which are available in both views.

**SELECT** *concat*(**user**.**first\_name**, **' '**, **user**.**surname**) **as 'name'**, food\_type.**food\_type\_name**, food\_purchase.**cost FROM user  
 LEFT JOIN** food\_transaction **ON user**.**user\_id** = food\_transaction.**user  
 RIGHT JOIN** food\_purchase **ON** food\_transaction.**food\_transaction\_id** = food\_purchase.**food\_transaction  
 LEFT JOIN** food\_type **ON** food\_purchase.**food\_type** = food\_type.**food\_type\_id ORDER BY** food\_purchase.**cost DESC**;



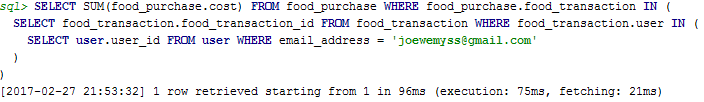


### Select total cost of all food transactions by a single user.

This would be a very commonly used query. This query will need to work across three tables, food\_purchase, food\_transaction and user. At first I wrote this query using sub queries and it looked like this:

**SELECT** *SUM*(food\_purchase.**cost**) **FROM** food\_purchase **WHERE** food\_purchase.**food\_transaction IN** (  
 **SELECT** food\_transaction.**food\_transaction\_id FROM** food\_transaction **WHERE** food\_transaction.**user IN** (  
 **SELECT user**.**user\_id FROM user WHERE email\_address** = **'joewemyss@gmail.com'** )  
);

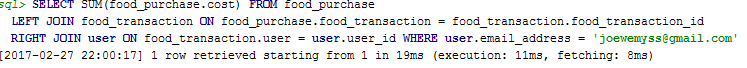
When I tested this, I found that it had a somewhat high response time, especially considering the small amount of data. The output of the query can be seen below.



Seeing this I tried to re-write my query to make use of joins instead. The new query looked like this:

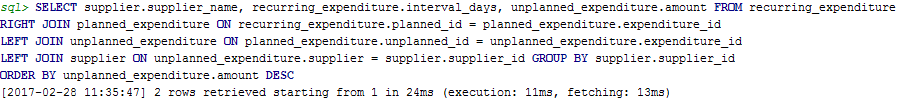
**SELECT** *SUM*(food\_purchase.**cost**) **FROM** food\_purchase  
 **LEFT JOIN** food\_transaction **ON** food\_purchase.**food\_transaction** = food\_transaction.**food\_transaction\_id  
 RIGHT JOIN user ON** food\_transaction.**user** = **user**.**user\_id WHERE user**.**email\_address** = **'joewemyss@gmail.com'**;

The new query showed a substantial performance increase on the previous method.



### Total Recurring Expenditure Grouped by Supplier

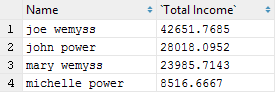
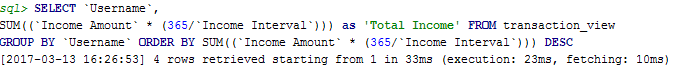
**SELECT** supplier.**supplier\_name**, recurring\_expenditure.**interval\_days**, unplanned\_expenditure.**amount FROM** recurring\_expenditure  
**RIGHT JOIN** planned\_expenditure **ON** recurring\_expenditure.**planned\_id** = planned\_expenditure.**expenditure\_id  
LEFT JOIN** unplanned\_expenditure **ON** planned\_expenditure.**unplanned\_id** = unplanned\_expenditure.**expenditure\_id  
LEFT JOIN** supplier **ON** unplanned\_expenditure.**supplier** = supplier.**supplier\_id GROUP BY** supplier.**supplier\_id  
ORDER BY** unplanned\_expenditure.**amount DESC**;





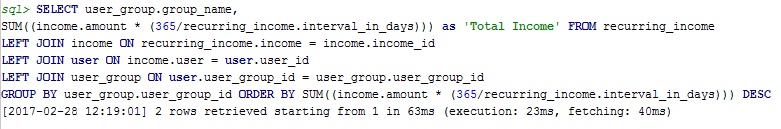
### Total Annual Income per user

**SELECT** `Username`,  
*SUM*((`Income Amount` \* (365/`Income Interval`))) **as 'Total Income' FROM** transaction\_view  
**GROUP BY** `Username` **ORDER BY** *SUM*((`Income Amount` \* (365/`Income Interval`))) **DESC**;



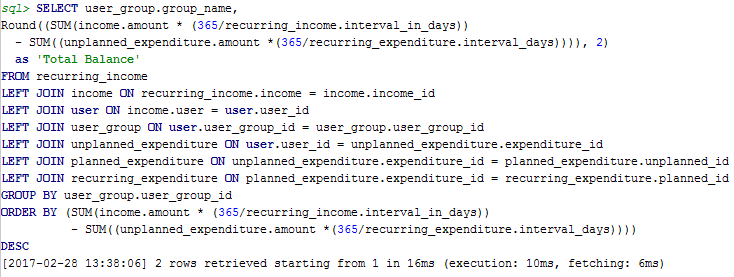
### Total Annual Income per User Group

**SELECT** user\_group.**group\_name**,  
*SUM*((income.**amount** \* (365/recurring\_income.**interval\_in\_days**))) **as 'Total Income' FROM** recurring\_income  
**LEFT JOIN** income **ON** recurring\_income.**income** = income.**income\_id  
LEFT JOIN user ON** income.**user** = **user**.**user\_id  
LEFT JOIN** user\_group **ON user**.**user\_group\_id** = user\_group.**user\_group\_id  
GROUP BY** user\_group.**user\_group\_id ORDER BY** *SUM*((income.**amount** \* (365/recurring\_income.**interval\_in\_days**))) **DESC**;



### Total Annual Balance per User Group

**SELECT** user\_group.**group\_name**,  
*Round*((*SUM*(income.**amount** \* (365/recurring\_income.**interval\_in\_days**))  
 - *SUM*((unplanned\_expenditure.**amount** \*(365/recurring\_expenditure.**interval\_days**)))), 2)  
 **as 'Total Balance'  
FROM** recurring\_income  
**LEFT JOIN** income **ON** recurring\_income.**income** = income.**income\_id  
LEFT JOIN user ON** income.**user** = **user**.**user\_id  
LEFT JOIN** user\_group **ON user**.**user\_group\_id** = user\_group.**user\_group\_id  
LEFT JOIN** unplanned\_expenditure **ON user**.**user\_id** = unplanned\_expenditure.**expenditure\_id  
LEFT JOIN** planned\_expenditure **ON** unplanned\_expenditure.**expenditure\_id** = planned\_expenditure.**unplanned\_id  
LEFT JOIN** recurring\_expenditure **ON** planned\_expenditure.**expenditure\_id** = recurring\_expenditure.**planned\_id  
GROUP BY** user\_group.**user\_group\_id  
ORDER BY** (*SUM*(income.**amount** \* (365/recurring\_income.**interval\_in\_days**))  
 - *SUM*((unplanned\_expenditure.**amount** \*(365/recurring\_expenditure.**interval\_days**))))  
**DESC**;

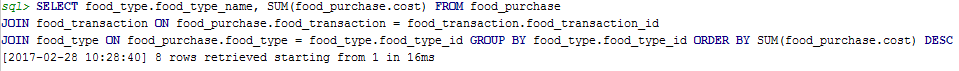


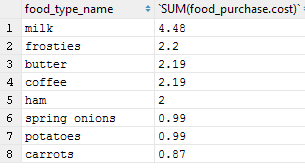
### Total Spent per Food Item

This query will show the total spent, grouped by food type. This takes into account all of the purchases and will group them by the food item.

Again, this belongs in this section as the food names are not available from the transaction view.

**SELECT** food\_type.**food\_type\_name**, *SUM*(food\_purchase.**cost**) **FROM** food\_purchase  
**JOIN** food\_transaction **ON** food\_purchase.**food\_transaction** = food\_transaction.**food\_transaction\_id  
JOIN** food\_type **ON** food\_purchase.**food\_type** = food\_type.**food\_type\_id GROUP BY** food\_type.**food\_type\_id ORDER BY** *SUM*(food\_purchase.**cost**) **DESC**;





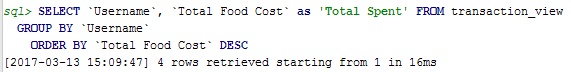
## Transaction View

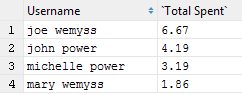
These queries would be performed on the Transaction partition. These are all of the queries that relate to fiscal matters.

### Food Spending by User

These queries would be used to correlate all the food transactions by user. This query falls under the Transactions partition because it is totally concerned with how much money was spent on food, rather than the quantities purchased.

**SELECT** `Username`, `Total Food Cost` **as 'Total Spent' FROM** transaction\_view  
 **GROUP BY** `Username`  
 **ORDER BY** `Total Food Cost` **DESC**;

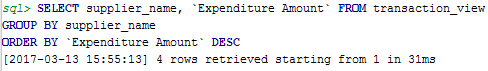


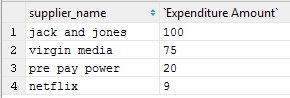


### Total Non-Food Expenditure per Supplier

This query shows the total non-food expenditure, grouped by supplier.

**SELECT supplier\_name**, `Expenditure Amount` **FROM** transaction\_view  
**GROUP BY supplier\_name  
ORDER BY** `Expenditure Amount` **DESC**;





### Total Food Expenditure per Supplier

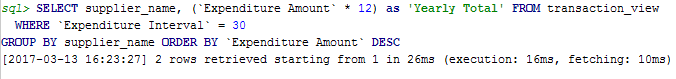
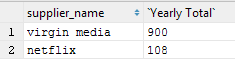
**SELECT** `Food Supplier Name`, *SUM*(**DISTINCT** `Food Cost`) **FROM** transaction\_view  
**Group BY** `Food Supplier Name` **ORDER BY** *SUM*(**DISTINCT**`Food Cost`) **DESC**;





### Total Annual cost of all monthly bills

**SELECT supplier\_name**, (`Expenditure Amount` \* 12) **as 'Yearly Total' FROM** transaction\_view  
 **WHERE** `Expenditure Interval` = 30  
**GROUP BY supplier\_name ORDER BY** `Expenditure Amount` **DESC**;

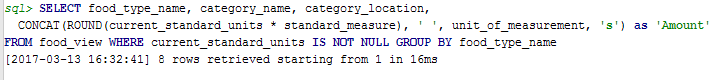
 

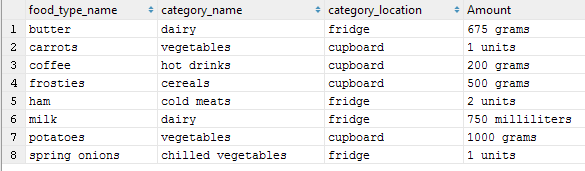
## Food View

These queries would be performed on the Food partition. These are the queries that relate to food quantities in the house.

### Current Food Stocks

**SELECT food\_type\_name**, **category\_name**, **category\_location**, *CONCAT*(*ROUND*(**current\_standard\_units** \* **standard\_measure**), **' '**, **unit\_of\_measurement**, **'s'**) **as 'Amount'  
FROM** food\_view **WHERE current\_standard\_units IS NOT NULL GROUP BY food\_type\_name**;





### Food stocks required for a recipe

**SELECT food\_type\_name**, *CONCAT*((**standard\_measure** \* food\_view.**number\_of\_units**), **' '**, **unit\_of\_measurement**, **'s'**)  
**FROM** food\_view **WHERE recipe\_name**=**'Ham and Cheese Sandwich'**;



