A Food Traceability Database Model with Base Parameters and Algorithms

1 author:

Cagla Ediz
Sakarya University
13 PUBLICATIONS **2** CITATIONS

A Food Traceability Database Model with Base Parameters and Algorithms

Araştırma Makalesi/Research Article

Çağla EDİZ

Department of Management Information Systems, Sakarya University, Sakarya, Turkey
cediz@sakarya.edu.tr

Abstract— This study is offering a common food traceability algorithm and a sample database model usable for food related companies. Monitoring food items and related sub-items is important in order to recall them when a food problem occurs and to investigate the source of the problem. If food companies have a common standard structure for traceability, it will be easier to share and transmit these records. In addition, food workers, supervisors and consumers will be able to monitor food more safely and comfortably. For these reasons, a sample structure, considering basic parameters and methods used in food traceability studies, is tried to be established within this study. For this purpose, the basic parameters and relationships required to ensure food traceability in food traceability studies are investigated by content analysis and an entity relationship diagram is created using these parameters. Then an algorithm is prepared to combine the records of food items systematically. Afterwards, the validity of the database is examined through the data available in a food company. In this way, this study aims to make forward and backward food traceability in a simple manner using recursive queries and to provide infrastructure for future standard applications.

Temel Parametreleri ve Algoritmalarıyla Bir Gıda İzlenebilirliği Veritabanı Modeli

Özet— Bu çalışma, gıda ile ilgili çalışan şirketlerde ortak kullanılabilecek bir gıda izlenebilirliği algoritması ve bu kapsamda hazırlanmış örnek bir veritabanı sunmaktadır. Gıda kalemlerini ve ilgili alt kalemleri izlemek, gıda ile ilgili bir problem yaşandığında onları geri çağırabilmek ve yaşanan problemin kaynağını tespit edebilmek için önemlidir. Gıda şirketleri izlenebilirlik için ortak bir standart yapıya sahip olursa, bu kayıtların paylaşımı ve iletimi daha kolay olacaktır. Ayrıca, gıda işçileri, denetçiler ve tüketiciler gıdaları daha güvenli ve rahat bir şekilde izleyebileceklerdir. Bu nedenlerden dolayı, bu çalışmada gıda izlenebilirliğinde kullanılan temel parametreler ve yöntemler dikkate alınarak örnek bir yapı oluşturulmaya çalışıldı. Bu amaçla, gıda izlenebilirliği makalelerinde kullanılan temel parametreler içerik analiziyle araştırılmış ve bu parametreler kullanılarak varlık ilişki diyagramı oluşturulmuştur. Ardından, gıda izlenebilirlik kayıtlarını sistematik olarak birleştirebilmek için bir algoritma hazırlanmıştır. Daha sonra, bir gıda şirketinde mevcut veriler üzerinden veritabanının geçerliliği incelenmiştir. Bu şekilde, bu araştırma çalışması özyinelemeli sorguları kullanarak basit ve kolay bir şekilde gıda izlenebilirliği sağlamayı ve gelecekteki standart uygulamalar için altyapı sağlamayı amaçlamaktadır.

1. INTRODUCTION

Traceability is defined as "the ability to trace the history, application or location of that which is under consideration" in ISO 9001:2000 standards. Olsen and Borit [1] evaluated existing traceability definitions and made a new traceability definition like "the ability to access any or all information relating to that which is under consideration, throughout its entire life cycle, by means of recorded identifications". Traceability can be achieved by tracing (traceback) or by tracking (traceforward). Tracing is the detection of the source of food in the supply chain; on the other hand, tracking is the following of food's route in supply chain [2]. Ensuring food traceability has many benefits [3-5]. First of all, thanks to traceability, people reach food safely. Then traceability provides a cause and effect relation for food recalling and so ensures that the risky products are easily removed from the market and maintains its brand image. In addition traceability also avoids counterfeiting and provides feedback on product quality, distribution, improvement of conditions and easy access to information during inspections.

ISO 22005 Food Traceability Standard expects each firm to know what they get and what they send according to the principle "one-up and one-down". For this purpose in supply management system, each individual firm records entering and leaving products in food supply chain. Thus, when needed, they can combine and present that knowledge to the authorities [6]. European Commission says that, when a risk is identified by national authorities or food businesses, they must find its source to isolate the problems and must prevent delivering of the contaminated products [5]. So keeping records of food in order to find the source of the problem and take the necessary precautions are important for each firm.

1.1. Reasons For This Study

The systematic literature review indicates that even in scientific articles there are lot of confusions and inconsistencies about traceability [1]. Wognum et. al. (2011) state this as follows, "Today's legislation provides no clear rules for the required performance of traceability systems. Moreover, current QMSs (like ISO), focus at in-company traceability but not full supply-chain traceability" [7]. Although, the EU Regulation 178 /2002 which took effect on 1st January 2005, expects for efficient traceability in the Food Industry, it doesn't offer a specific methodology for the food business operators. Thus food business operators are free to choose the mechanism to ensure effective traceability method for their products [8, 9]. The current methods used in traceability systems are very ineffective to provide a healthy working traceability system. The main reason for this is the lack of standardized communication between different systems which are causing the lack of traceability in the food industry [10]. Traceability can only be achieved if the actors in the supply chain are able to access and process information by

providing interoperate and transparency [11,12]. Folinas and et al. (2006) described the main features of a good traceability system. According to them, filtering and sorting information should be done by the system. In addition the system should be compatible with current technologies, international codification standard EAN-UCC and internet standards [8]. Although there are a lot of controls for food, generally systems are similar and relational databases are used to store data [13]. So investigating the data models in literature can be helpfull to create a common algorithm and data traceability structure for food traceability.

In the next section, main parameters and algorithms of food traceability are searched. In the third section, the validity of the main algorithm is investigated by using a dairy company data as a case study. After discussion and conclusion sections, ""Appendix" section is added to show a sample database prepared in MSSQL platform.

2. MAIN PARAMETERS AND ALGORITHMS OF FOOD TRACEABILITY

21. Selecting Main Parameters

Many scientists have worked to define necessary structure of traceability systems. In one of these studies, Moe (1998) divides structure of traceability systems into two parts as product and activities [3]. Afterwards, he divides product into two parts as product type and product quantity. He also divides activity into two parts as activity type and activity time. In another study, Folinas et al. (2006) propose an XML based technology to model of traceability data. They reached six elements for this model. These are data of the products, node elements, descriptions, usage, properties and information about the data measurements [8]. Regattieri et al. (2007) also define traceability system in their work. On the other hand, they split their model into four categories like: product identification, data to trace, product routing and traceability tools [14]. Different from these papers, a "data model" was offered by using business actors (egg producer, breeder and grower) in 2003 [15, 16]. This data model includes relations between lots, operations and operation controls. Another traceability model belongs to Bechini et al. (2008). In this model, "Lot" and "Activity" entities modelled by "TraceableEntity" which is an abstract class and they are also in relation with "Site" and "Responsible Actor". Because traceability is evaluated by quality features, "Lot" and "Activity" entities are also in relation with quality entities [17].

On the other hand, Bevilacqua et al. (2009) developed a computerised system for managing product traceability by business process reengineering. The data used in the system start from the production of seed and cover all stages of cultivation, processing and marketing of the product[18].

Table 1. Data parameters in food traceability literature

Table 1.Data parameters in food traceability literature	Item	Item Type	Quantity	Activity	Activity Type	Activity Time	Quality	Lot	Place	Equipments	Firm	Responsible
Moe, 1998 [3]	√	√	√	√	√	√						
Van Dorp, 2003 [15]	√			√					√			
Bechini et al., 2008 [17]				√				√	√			√
Thakur and Hurbugh, 2009 [19]	√		√		√	√	√	√	√	√	√	√
Khabbazi et al., 2010 [20]				√		√	√					
Vukatana et al., 2016 [21]	√							√	√		√	√
Wang and Yue, 2017 [24]	√	√	√			√	√					
Guirado-Clavijo, 2019 [25]	√				√	√	√		√			√
Shahid et al., 2020 [28]	√		√			√		√	√		√	√
Corolla et al., 2020 [23]	√				√			√	√			√

Thakur and Hurburgh (2009) worked about grain supply chain traceability. Their model shows which information related with grain lots must be recorded and which information must be transferred to the next actor [19].

Khabbazi et al. (2010) used once more UML diagrams for traceability in production process. In this data model, based on lot details; dynamic data of qualities and purchase activities are processed [20]. Again in 2011, a relational database management system which stores all necessary information related to the grain lots in order to enable product traceability was proposed by Thakur et al. (2011). This system is executable to trace back and track forward [10]. Different from the others, Storoy et al. (2013), split the food traceability framework in to six components. They are principle of unique identifications, documentation of transformations of units, generic language for electronic exchange of information, sector-specific language for electronic information exchange, generic guidelines for implementation of traceability and sector-specific guidelines for implementation of traceability [2].

In an interview with winemakers in Albanian, Vukatana et al. (2016) understood that data required by ISO to ensure product traceability is kept as only hard copy. Based on this, they provided a data model for them to develop an agile traceability system [21].

Suppliers follow, control, plan and optimize business processes via Internet. In this context, Verdouw et al. (2016) analyse the concept of virtual food supply chains and propose an architecture to implement information systems via internet [22]. On the other hand, addition to internet, Industry 4.0 technologies are also usefull for food traceability. Corolla et al. (2020), in their case study with olive producers, showed that by the use of Industry 4.0 technologies in food traceability, the big data obtained along the supply chain can be transformed into useful information for modern consumers [23]. Wang and Yue also used technology to track food in 2017. In this study, they aimed to create a warning system that prevents possible problems by evaluating data obtained by the association rule method, which is one of the methods of

machine learning [24]. Yet in another study based on IoT technologies, Guirado-Clavijo et al. (2019) created a data model that transfers data to the cloud for greenhouse production using information Technologies [25].

One of the most popular topics in the food supply chain lately is block chain technology [26-28]. The use of blockchain technology in food supply chain does not aim to ensure data model of food activities throughout entire life of food chain. Instead, the purpose of using blockchain technology is to ensure the reliability of food data transfer between the actors involved in food traceability.

The basic parameters in Table 1 are determined by examining the scientific articles containing data models, frameworks and supply chain data flows related to food traceability. In order to minimize the probability of recalls, quality controls should be included in traceability systems as seen in Table 1 [29]. However, because it is focused on the main entities in this work, quality records, item types and equipments are not included to prepare entity-relation diagrams. These excluded entities should be linked with the main entities after preparing these.

2.2. Entity-Relation Diagram

Each material has a volume in space and needs time to change its location. In addition, in an instant time, only one material can be located in an identical location. For this reason, the most basic parameters to monitor an object are space and datetime data of objects. These parameters are also important for the food traceability. Location and datetime data are needed to recall food, especially when a problem occurs [30]. On the other hand, products or inputs (like fertilizer, chemical spreys) change properties of food by processes and compose food [31]. Therefore products, inputs and processes are in the center of food traceability. In this study, products and inputs are called as items, also processes are called as activities.

Evaluating the above parameters, the main parameters affecting food health can be divided in item, activity and

place. There is a tracing relationship between the item, place and activity entities as seen in Figure 1. The attributes of this relationship are "LotID", "TagID", "Quantity" and "Unit". The properties of food change with activities. An activity starts at a spesific time and continues up to the next activity.
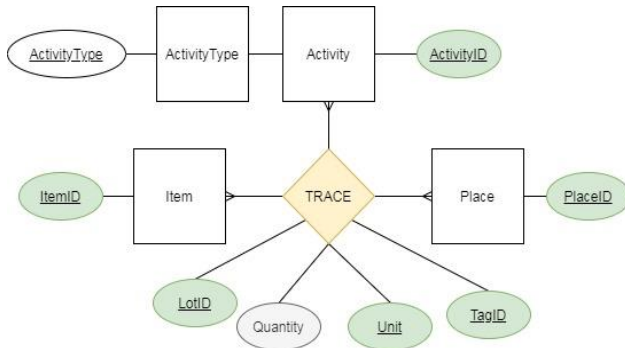

Figure 1. Entity relationship diagram for food traceability

Movements that affect food traceability parameters are called activities. "ActivityID" must be unique in order to avoid errors in the food traceability system. So "ActivityID" can be generated using "DateTime" and "PlaceId".

Items represent food products or inputs added into food. The product activity change items according to prepared Bills of Materials (BOM). On the other hand, "Place" defines where the food is. It has attributes effecting food like temperature, location, containment. The defined "Place" concept can be a fixed place like a warehouse or can be a mobile place like a tank of a truck. So "Place" is not geographic location. But "Place" has a geographical location attribute depending on time. Geographical locations are beyond the scope of this study.

The main considerations in the concept of place can be listed as follows:

•   The place information used in monitoring of a food item should always be at the same hierarchical level. For example, if the places for the same food items are defined at the warehouse level, the shelf numbers in the warehouse must not be used as place. if there is a change of the place level, changing should be done for all food items with the same item number.

•   The selected hierarchical level for place should be sustainable.

•   If food is in two predefined places at the same time, the more detailed place is selected. For example, during transportation of food, if the transport vehicle enters the warehouse, the place of food should be recorded as transport vehicle.

With today's technology, monitoring of every object in terms of instant time and space is possible but too difficult,

so more applicable monitoring systems are preferred to achieve this. In these applications, both monitored objects and places of these objects are grouped into certain criterias and monitoring is done in this way. The concept of "Trace Resource Unit (TRU)" emerges as a result of combining the objects needed for monitoring according to certain parameters and is used for lot numbers mostly. Lot defines the food produced or packaged under the same conditions [32]. On the other hand, TRU is defined in many articles as " that which is under consideration" and TRU can be different objects like trade unit (e.g. a box, a bottle), logistic unit (e.g. a pallet, a container) or production unit (e.g. lot) [33]. Because each lot is formed by an activity, the "ActivityID" can be used as "LotID". The International System of Units" were used e.g. kilogram, litre and quantity can be expressed in real numbers.

Tags are physical labels and attach to monitored food. Tags can be in different types like RFID tags, barcode tags. Using tags make easier to record activities in interface programs [34]. Monitored object's properties change via activities. If the tags are prepared very easily, the traceability tags can change after each activity as it's supposed to be. However, many firm unable to implement these applications yet and traceability cannot be provided completely. Olsen and Borit (2018) gave an example for this situation [33]. In this example, red and green trucks are used to transport food from the production area to the storage area. Red truck failure caused the products to be exposed to high temperatures. However, since traceability tags are not changed after this transport activity, it will never be known which of these products carried to the same location are exposed to high temperatures. In future, automatically read tags will be used for food traceability to avoid many physical operations. Thanks to IoT, huge number of tags can be used to trace activities in future. Today, "TagID" value can be a constant string like "X" if there is no any tag. The records related with "TagID"'s can be used with the following rules:

- Each tag matches with one or more "ItemID", "LotID", "PlaceID" , "Unit" and "Quantity" values.
- Among these records, "LotID" and "PlaceID" must be same.

Using above entity relationship diagram, the data model in Figure 2 was formed for activities and tracing. The "FoodTrace" table contains the basic components of food traceability. These are "ItemID", "LotID", "TagID", "PlaceID", "Unit" , "ActivityID" and "Quantity". On the other hand, "Activity" table includes more detailed parameters such as time, firm and location. In "FoodTrace" table, combination of "ItemID", "LotID", "TagID", "PlaceID", "Unit" , "ActivityID" must be defined as primary key.
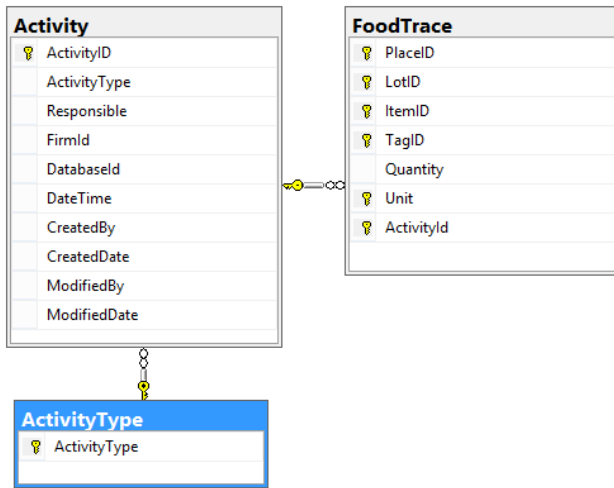
Figure 2. Data model for basic food traceability

2.3. Main Algorithm of Food Traceability

TRU are frequently used for lots or batches [33]. On the other hand, to query traceability of food, we need another concept, which is called in this study as Recorded Resource Unit (RRU). RRU includes "LotID", "ItemID", "TagID", "PlaceID" and "Unit".

The relations of the traceability records have two types in this study. The first one is the relation among activities and the second is the relation among RRU.

The relation among activities can be explained as follows. The attributes of food (ItemID, LotID, PlaceID, Quantity, TagID and Unit) used for food traceability vary by using activities. For that reason, to follow changings of attributes, one or more records are added to the "FoodTrace" table for each activity. Records with negative quantities are inputs of their activity; records with positive quantities are outputs of their activity. So "ActivityID" provides the relationship between the records used as inputs in the activity and the records generated as the result of the activity.
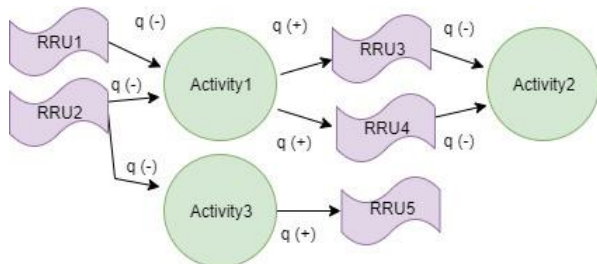

Figure 3. Algorithms of "FoodTrace" recording

The relation among RRUs can be explained as follows. Each record which is input of an activity with negative quantity, is also output of previous activity (except for some type activities) and has positive quantity value in previous activity record. All output values in the previous activity record are equal to the input record in the new activity except "Quantity" and "ActivityID" fields. This algorithm binds records entering the new activity to the

records coming from the previous activity. This algorithm is shown in Figure 3.

2.4. Existence Control Algorithm

For each record with negative quantity, it is controlled whether there is required quantity for entered record or not. It is not allowed to record if there is not required quantity.

If π_{Quantity} (σ (@LotID, @ItemID , @TagID @PlaceID, @Unit, @Quantity, @ActivityID)) < 0 $\Rightarrow$

@Count $\leftarrow \pi_{\text{SUM(Quantity)}}$ ($\sigma_{\text{LotID=@LotID and ItemID=@ItemID and TagID=@TagID and PlaceID=@PlaceID and Unit=@Unit}}$ (FoodTrace));

@Quantity+ @Count > 0

2.5. Update and Delete Triggers for FoodTrace Table

For a record to be deleted or to be modified, the resulting materials of this record's activity (RRU) must not have been used by another activity. That is, only the last activity records can be deleted or updated. If it is necessary to make update or delete, it is also necessary to delete all the activity records linked this activity, one by one starting from the last record. In order to perform this control, it is necessary to create triggers depending on the update and delete queries. In this trigger @Activity (ActivityID) table is used to reach next activities.

@Activity $\leftarrow \pi_{\text{ActivityId}}$ ($\sigma_{\text{ItemID=@ItemID and Unit=@Unit and LotID=@LotID and TagtID=@TagID and PlaceID=@PlaceID and Quantity<0}}$ (FoodTrace))

If $\pi_{\text{count(*)}}$(@Activity) $\Rightarrow$ It is not allowed to update or delete

2.6. Searching of Food Places

A stored procedure was prepared for recalling of products with problems. To identify a product, lot number and item number of that product are used. In this stored procedure, created for the identification of food places, product locations are found by using item (@ItemID) and lot(@LotID) information.

To find the place of the food:

$\pi_{\text{SUM(Quantity), ItemID, LotID, TagID,Unit , PlaceID}}$ (ItemID, LotID, TagID, Unit, PlaceID $G_{\text{SUM(Quantity)}}$ $\sigma_{\text{SUM(Quantity)>0, LotID=@LotID and ItemID=@ItemID}}$ (FoodTrace)

If the searched food's last activity is "Transfer", the food's place knowledge should be requested from other firm which transformation occurs to.

2.7. Algorithms for Traceback and Traceforward with Recursive Queries

"ActvityID" and "RRU" provide link among food records. In this systematic, consumed RRU records in activity are with negative quantities and produced RRU records in activity are positive quantities. Because each RRU produces after an activity, we can trace the food by using activities. For this reason, with traceback, activities in which monitored items have positive quantities are

searched and also, with traceforward, activities in which monitored items have negative quantities are searched as seen in Figure 4.

$$@food \leftarrow @placeid, @lotid, @itemid, @unit$$

$$@activity \leftarrow @activity - \sigma(@activity)$$

$$@activity \leftarrow \pi_{ActivityID} \, \sigma_{XXXX} \, (@activity \bowtie @food) - \pi_{ActivityID} \, (@activitytrace)$$

$$\pi_{count(ActivityID)} \, (@activitytrace) > 0$$

$$@activitytrace \leftarrow @activitytrace \cup \sigma(@activity)$$

$$@food \leftarrow @food - \sigma(@food)$$

$$@food \leftarrow \pi_{PlaceID, LotID, ItemID, Unit} \, \sigma_{YYYY} \, (@activity \bowtie FoodTrace)$$

$$\pi_{ItemID, LotID, PlaceID, Unit, Quantity, ActivityId} \, \sigma(@activitytrace \bowtie FoodTrace)$$

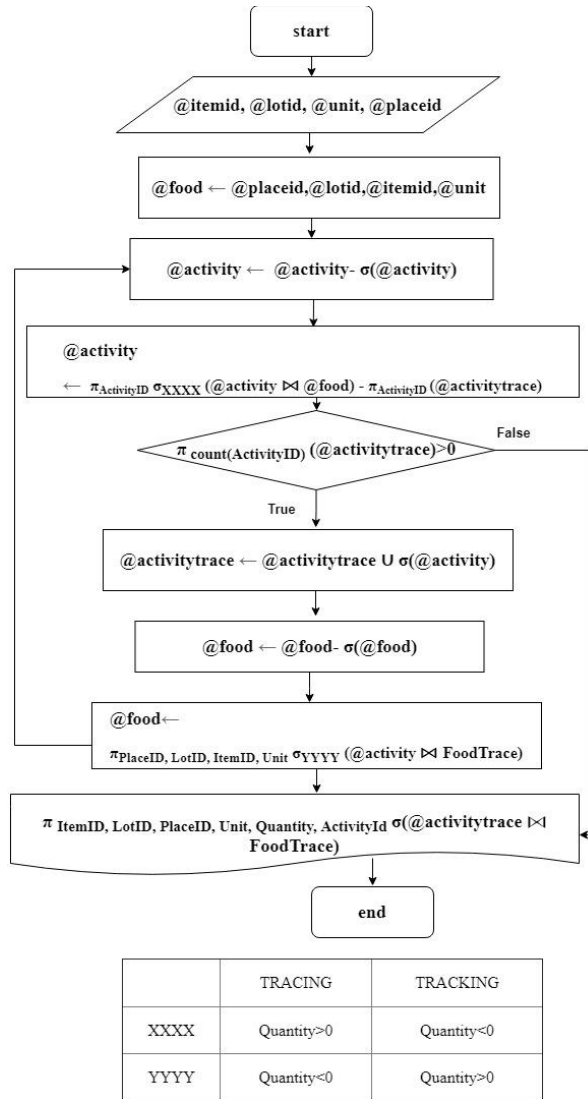|	TRACING	TRACKING
XXXX	Quantity>0	Quantity<0
YYYY	Quantity<0	Quantity>0

Figure 4. Flowchart of tracing and tracking

When product information is transferred to a different database, the product's tracing records are added to the new database. The links of the products with other products represent a graph diagram. For this reason, although some items in the data transferred from another server have different activities afterwards, only the transferred items' activities can be monitored after transfer. For other activities related to these items, information can be obtained from the server where they are transferred from. Every company is responsible for their own activities.

3.   CASE STUDY

In a dairy company, inputs, processes and outputs are investigated to check prepared algorithm applicability. So, first of all, activities are searched in the dairy factory.

3.1. Activity Types in Dairy Factory

In this study, to select main activities in studied dairy factory, document analyses and interviews with workers were done. The activities for the products in the dairy factory are named as "Stock", "Raw", "Feed", "Milk", "Consume", "Move", "Destruct", "Transfer in", "Transfer to", "Recycle", "Split", "ItemChange", "LotChange", "UnitChange", "TagChange" and "Join".

"Stock" is used for first records of stocks. Actually, it doesn't sustain traceability. But, at the beginning of recording, this activity is used to record items.

"Raw" is the activity to record materials which have no information about previous activities like fishing or obtaining water from drinking fountain.

"Feed" is used to record feeding.

"Milk" is the activity to record milking process.

"Move" is used for purchases, sales, all kinds of internal and external shipments.

"Consume" activity is used to record items sent to the last consumer.

"Destruct" activity is used for registration of the items turned into garbage for different reasons.

"Recycle" is used to record reused items whose traceability records are lost such as the reuse of bottles returned as deposit in production.

"Split" is used to obtain items with known traceability records, such as separating the cheese from the damaged package and re-recording.

"Produce" is used to record item numbers changing according to BOM.
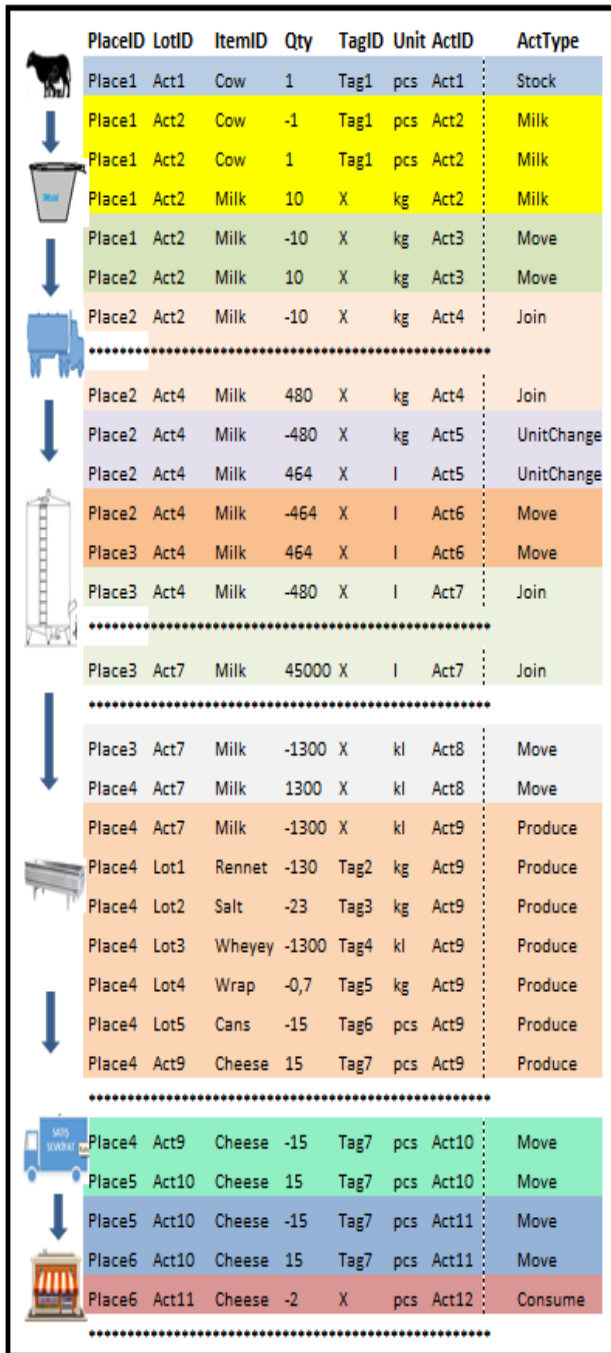
"Join" activity is used to combine food items having the same "ItemID" and different "LotID" such as mixing milks having different "LotID" in a tank.

If the monitoring records of a product are transferred to another server for some reasons such as sales or distribution, the activity information of this product end with the "Transfer to" activity. Before this activity "Move" activity must be done not to lose tracking knowledge. If the monitoring records of a product are transferred from another server, again "Transfer to" activity is used.

Firms can change item numbers or lot numbers using "ItemChange" or "LotChange" activities because they want to use their own definitions. "UnitChange" activity is used to change product units and so quantity. Sometimes,

because of some destructions in tags, there can be needed to use **"TagChange"** ativity.

3.2.   White Cheese Production Records as a Case Study

	PlaceID	LotID	ItemID	Qty	TagID	Unit	ActID	ActType
	Place1	Act1	Cow	1	Tag1	pcs	Act1	Stock
	Place1	Act2	Cow	-1	Tag1	pcs	Act2	Milk
	Place1	Act2	Cow	1	Tag1	pcs	Act2	Milk
	Place1	Act2	Milk	10	X	kg	Act2	Milk
	Place1	Act2	Milk	-10	X	kg	Act3	Move
	Place2	Act2	Milk	10	X	kg	Act3	Move
	Place2	Act2	Milk	-10	X	kg	Act4	Join
	Place2	Act4	Milk	480	X	kg	Act4	Join
	Place2	Act4	Milk	-480	X	kg	Act5	UnitChange
	Place2	Act4	Milk	464	X	l	Act5	UnitChange
	Place2	Act4	Milk	-464	X	l	Act6	Move
	Place3	Act4	Milk	464	X	l	Act6	Move
	Place3	Act4	Milk	-480	X	l	Act7	Join
	Place3	Act7	Milk	45000	X	l	Act7	Join
	Place3	Act7	Milk	-1300	X	kl	Act8	Move
	Place4	Act7	Milk	1300	X	kl	Act8	Move
	Place4	Act7	Milk	-1300	X	kl	Act9	Produce
	Place4	Lot1	Rennet	-130	Tag2	kg	Act9	Produce
	Place4	Lot2	Salt	-23	Tag3	kg	Act9	Produce
	Place4	Lot3	Wheyey	-1300	Tag4	kl	Act9	Produce
	Place4	Lot4	Wrap	-0,7	Tag5	kg	Act9	Produce
	Place4	Lot5	Cans	-15	Tag6	pcs	Act9	Produce
	Place4	Act9	Cheese	15	Tag7	pcs	Act9	Produce
	Place4	Act9	Cheese	-15	Tag7	pcs	Act10	Move
	Place5	Act10	Cheese	15	Tag7	pcs	Act10	Move
	Place5	Act10	Cheese	-15	Tag7	pcs	Act11	Move
	Place6	Act10	Cheese	15	Tag7	pcs	Act11	Move
	Place6	Act11	Cheese	-2	X	pcs	Act12	Consume

Figure 5. A traceability study using a dairy factory's data

White cheese production in the dairy factory is choosen as a case study. White cheese production is seen partially as an example in Figure 5. In this sample, recording cow for the first time, milking the cow, transporting and mixing milk by transportation tankers, mixing milks in milk boilers, producing white cheese and selling activities are shown. The values are recorded as simple words to be understood easily.

The studied algorithm allows to monitor the basic data in a chained manner depending on activities. In some cases the same item both enters and exits an activity. For example, the milked cow is both an input and output of the activity of milking.

3.3. RRU Changes in Activities

Activities can create new records, end the records, change the properties of the records according to the activity type with changing the number of records or without changing them (Figure 6).
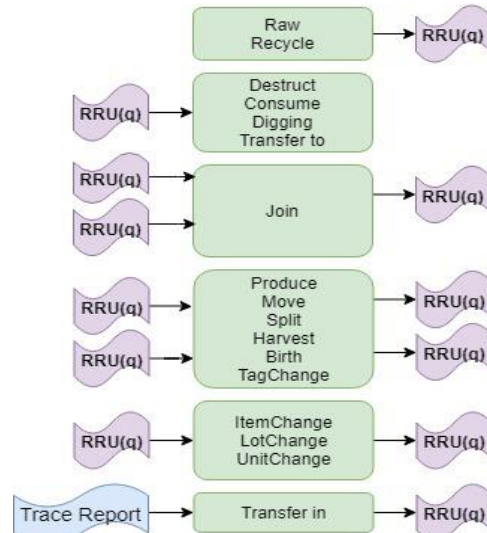

Figure 6. Transformations of RRUs within some activities

4. DISCUSSION

Although similar structures and relational databases are used as food traceability tools, there is no a common food traceability model in literature. This study was put forward to fullfill this need. One of the distinguishing feature of this study is that it is intended to use the algorithm in all food businesses, not for a private company. For this reason, only the main parameters are focused. Entity Relation Model is used to determine the associations of basic parameters. Besides, food traceabilities are reported at the data model levels generally in literature. In this study, an algorithm is prepared to associate the basic parameters and the necessary rules for the sustainable usage of this model in a holistic approach are explained. Since the algorithm prepared in the study is based on very basic concepts, it can be adapted to every activity. In fact, similar algorithms with the presented one, are used in many ERP programs for inventory tracking or inventory costs. As an example of this structure, "InventTrans" table in Microsoft Dynamics, which is used in calculating inventory costs, can be shown. In this study, an algorithm similar to algorithms that connects the records in these tables, is designed for

traceability and the basic rules required for sustainability are explained.

The algorithms needed to perform forward and backward food traceability using the parameters are prepared in a simple manner using the recursive method. In this way, an attempt was made to establish an infrastructure for the food traceability system to be used in the future. Tables, triggers and procedures prepared in MSSQL Server are shared in the "Appendix" in order to understand of the study more easily.

5. CONCLUSION

Using a common food traceability model for all food items will greatly facilitate sustainability of food traceability. For this reason, this study deals with food traceability with basic parameters and algorithms for a common database model. Thus tracing and tracking can be done using the same recursive queries in different type of companies. When there is a problem with food, executing tracing query in database helps to find the source of the problem. In addition, when a problem encountered, all items related to problem can be called by using tracking queries. Besides, using a standard common structure in different food related companies will facilitate transmission of data among companies and helps inspection activities. Food traceability should not only monitor the place, quantity and activities of food, but also monitor whether food's and activities' exhibit expected attributes. That is, food traceability is not limited only to the parameters mentioned in this study. However, the first step to ensure a universal food traceability should establish a standard structure for the main parameters. In future, this framework of standardization should be expanded and the parameters such as quality, equipment and location attributes can be added to this structure.

To avoid possible errors in recording, there must be controls dependent on activity types. For example, "Raw" activities have input records only. These rules should be added as triggers in the database system. Besides, each company should create additional controls according to their needs. For instance, product trees can be used to execute query for the accuracy of records consumed and produced in "Produce" activity. Moreover, conversion coefficient tables can be used to convert units in "UnitChange" activity.

Companies use different ERP systems. For this reason, the food traceability system used in each company differs from each other. However, in order to achieve a common system, companies can prepare their traceability database as a data warehouse associated with their own information systems. Although proposed systematic recording is

difficult for many companies for the time being, this is not true for the near future. The world has been entering Industry 4.0. In the near future, these operations can be performed easily using sensors, IOT, RFID, intelligent softwares, cloud and big data systems. RFID numbers ("TagID") will be associated with "RRU" and "Quantity". So activities will be read from sensors and recorded by intelligent softwares automatically using a common model.

REFERENCES

[1]　P. Olsen, M. Borit, "How to define traceability", *Trends in food science & technology*, 29 (2), 142-150, 2013.

[2]　J. Storøy, M. Thakur, P. Olsen, "The TraceFood Framework–Principles and Guidelines for İmplementing Traceability In Food Value Chains", *Journal Of Food Engineering*, 115(1), 41-48, 2013.

[3]　T. Moe, "Perspectives on Traceability in Food Manufacture", *Trends in Food Science & Technology*, 9(5), May 1998, 211-214, 1998.

[4]　Z. Cebeci, "Gıda İzlenebilirliğinde Bilgi Teknolojileri", **Ulusal Tarım Kurultayı,** 15-17 Kasım 2006, Çukurova Üniversitesi, Adana, 189-195, 2006.

[5]　Internet: Europen Commission "Food Traceability" , Health and Consumer Protection, June 2007, http://ec.europa.eu/food/safety/docs/gfl_req_factsheet_traceability_2007_en.pdf., 8.10.2018.

[6]　J. Hu, X. Zhang, L. M. Moga, M. Neculita, "Modeling and Implementation of the Vegetable Supply Chain Traceability System", *Food Control*, 30 (1), 341-353, 2013.

[7]　P. N. Wognum, H. Bremmers, J. H. Trienekens, J. G. van der Vorst, J. M. Bloemhof, "Systems For Sustainability and Transparency Of Food Supply Chains–Current Status and Challenges", *Advanced Engineering Informatics*, 25(1), 65-76, 2011.

[8]　D. Folinas, I. Manikas, B. Manos, "Traceability Data Management for Food Chains", *British Food Journal*, 108(8), 622-633, 2006.

[9]　J. Wang, "Methods and Techniques of Supply Chain Management", *Information Technologies*, IGI Global, United States of America, 2012.

[10]　M. Thakur, B. J. Martens, C. R. Hurburgh, "Data Modeling To Facilitate İnternal Traceability at A Grain Elevator", *Computers and Electronics in Agriculture*, 75(2), 327-336, 2011.

[11]　A. Tursi, M. Dassisti, H. Panetto, "Products İnformation İnteroperability İn Manufacturing Systems", **In Ottavo Convegno AITeM (Associazione Italiana Tecnologia Meccanica),** 10, September, 2007.

[12]　H. Ringsberg, "Perspectives On Food Traceability: A Systematic Literature Review", *Supply Chain Management: An International Journal*, 19(5/6), 558-576, 2014.

[13]　T. Pizzuti, G. Mirabelli, M. A. Sanz-Bobi, F. Goméz-Gonzaléz, "Food Track & Trace Ontology For Helping The Food Traceability Control", *Journal of Food Engineering*, 120, 17-30, 2014.

[14]　A. Regattieri, M. Gamberi, R. Manzini, "Traceability of Food Products: General Framework and Experimental Evidence", *Journal of Food Engineering*, 81(2), 347-356, 2007.

[15] C.A. Van Dorp, "A Treaceability Application Based On Gozinto Graphs", **EFITA 2003 Conference**, 5-9 July 2003, Debrecen, Hungary, 280 -285, 2003.

[16] M. H. Jansen-Vullers, C. A van Dorp, A. J. Beulens, "Managing Traceability Information in Manufacture", *International Journal of Information Management*, 23(5), 395-413, 2003.

[17] A. Bechini, M. G. Cimino, F. Marcelloni, A. Tomasi, "Patterns and Technologies for Enabling Supply Chain Traceability Through Collaborative e-business", *Information and Software Technology*, 50 (4), 342-359, 2008.

[18] M. Bevilacqua, F. E Ciarapica, G. Giacchetta, "Business Process Reengineering of A Supply Chain and A Traceability System: A Case Study", *Journal of Food Engineering*, 93(1), 13-22, 2009.

[19] M. Thakur, C. R. Hurburgh, "Framework for Implementing Traceability System in the Bulk Grain Supply Chain", *Journal of Food Engineering*, 95 (4), 617-626, 2009.

[20] M. R. Khabbazi, M. Ismail, N. Ismail, S. A. Mousavi, "Modeling of Traceability Information System for Material Flow Control Data", *Australian Journal of Basic and Applied Sciences*, 4(2), 208-216, 2010.

[21] K Vukatana, K. Sevrani, E. Hoxha, (2016). "Wine Traceability: A Data Model And Prototype İn Albanian Context", *Foods*, 5(1), 11, 2016.

[22] C. N Verdouw, J.Wolfert, A. J. M. Beulens, A. Rialland, "Virtualization Of Food Supply Chains With The Internet Of Things", *Journal of Food Engineering*, 176, 128–136, 2015.

[23] A. Corallo, M. E. Latino, M. Menegoli, "Agriculture 4.0: How Use Traceability Data to Tell Food Product to the Consumers", **9th International Conference on Industrial Technology and Management (ICITM)**, 197-201, IEEE, 2020.

[24] J. Wang, H. Yue, "Food Safety Pre-Warning System Based On Data Mining For A Sustainable Food Supply Chain", *Food Control*, 73, 223-229, 2017.

[25] R. Guirado-Clavijo, J. A. Sanchez-Molina, H. Wang, F. Bienvenido, "Conceptual Data Model For Iot İn A Chain-İntegrated Greenhouse Production: Case Of The Tomato Production İn Almeria (Spain)", *IFAC-PapersOnLine*, 51(17), 102-107, 2018.

[26] J. F. Galvez, J. C. Mejuto, J. Simal-Gandara, "Future Challenges On The Use Of Blockchain For Food Traceability Analysis", *TrAC Trends in Analytical Chemistry*, 107, 222-232, 2018.

[27] A. Kamilaris, A. Fonts, F. X. Prenafeta-Boldú, "The Rise Of Blockchain Technology İn Agriculture And Food Supply Chains", *Trends in Food Science & Technology*, 91, 640-652, 2019.

[28] A. Shahid, A. Almogren, N. Javaid, F. Al-Zahrani, M. Zuair, M. Alam, "Blockchain-Based Agri-Food Supply Chain: A Complete Solution", *IEEE Access*, 8, 69230-69243, 2020.

[29] M. Mattevi, J.A. Jones, "Traceability İn The Food Supply Chain: Awareness And Attitudes Of UK Small And Medium-Sized Enterprises", *Food Control*, 64, 120-127, 2016.

[30] D. Montet, G. Dey, "History of food traceability", *In Food Traceability and Authenticity*, 1-30, CRC press, 2017.

[31] L. U. Opara, "Traceability in Agriculture and Food Supply Chain: A Review Of Basic Concepts, Technological Implications, and Future Prospects", *Journal of Food Agriculture and Environment*, 1, 101-106, (2003).

[32] Internet: Ministry Of Food, Agriculture And Livestock, http://www.resmigazete.gov.tr/eskiler/2012/01/20120104-11.htm 8.10.2018.

[33] P. Olsen and M. Borit, "The Components of A Food Traceability System", *Trends in Food Science & Technology*, 77, 2018, 143-149, 2018.

[34] E. Meydanoğlu, "RFID Sistemleri ve Veri Güvenliği", *Bilişim Teknolojileri Dergisi,* 1(3), 2008.

APPENDIX

```
create database [FoodTraceDB]
go
use FoodTraceDB
go
create table [dbo].[ActivityType]
    (
            [ActivityType] [varchar](12) not null primary key
            )
go
create table [dbo].[Activity](
        [ActivityID] [varchar](50) Primary key,
        [ActivityType] [varchar](12) Not null references
FoodTraceDB.dbo.ActivityType(ActivityType) ,
        [Responsible] [varchar](11) not null,
        [FirmId] [varchar](9) not null,
        [DatabaseId] [varchar](14) not null,
    [DateTime] [smalldatetime] null,
        [CreatedBy] [varchar](11) not null,
        [CreatedDate] [smalldatetime] null DEFAULT (getdate()),
        [ModifiedBy] [varchar](11) null,
        [ModifiedDate] [smalldatetime] null
        )
go
create table FoodTrace
(
        [PlaceID] [varchar](13) not null,
        [LotID] [varchar](50) not null,
        [ItemID] [varchar](13) NOT null,
         [TagID] [varchar](8) not null,
        [Quantity] [real] not null,
        [Unit] [varchar](10) not null,
        [ActivityId] [varchar](50) references
dbo.Activity(ActivityID)
        Constraint PK_FoodTrace Primary
Key(PlaceID,ItemID,LotID,TagID,Unit, ActivityId))


use FoodTraceDB
go
create          trigger        [dbo].[InsertRecord]            on
[FoodTraceDB].[dbo].[FoodTrace]
instead of insert
as
 begin
declare @PlaceID varchar(13);
declare @LotID varchar(50);
declare @ItemID varchar(13);
declare @TagID varchar(8);
declare @Quantity real;
declare @Count int;
declare @PreviousActivityID varchar(50);
declare @PreviousActivityType varchar(12);
declare @Unit varchar(10);
declare @ActivityID varchar(50);
declare @ActivityType varchar(12);
declare @counter int;
declare @say int;
```

```
select @PlaceID = i.PlaceID  from inserted i;
select @LotID = i.LotID  from inserted i;
select @ItemID = i.ItemID  from inserted i;
select @TagID = i.TagID  from inserted i;
select @Quantity = i.Quantity  from inserted i;
select @Unit = i.Unit  from inserted i;
select @ActivityID = i.ActivityID  from inserted i;

declare @food table([PlaceID] [varchar](13) not null,
        [LotID] [varchar](50) not null,
        [ItemID] [varchar](13) NOT null,
        [Unit] [varchar](10)  not null)

begin try

if (@Quantity<0)

begin
 --Control for previous activitiy/activities
 --find previous activitiy/activities
declare @activity table([ActivityId] [varchar](50) )
insert into @food values(@PlaceID,@LotID,@ItemID,@unit)
insert into @activity select ft.ActivityId from FoodTrace ft  join @food f
on
 f.ItemID=ft.ItemID and f.LotID=ft.LotID and f.PlaceID=ft.PlaceID and
f.Unit=ft.Unit where quantity>0

 --select @aa=ActivityId from @activity
--declare control_activities cursor --for select ActivityId from @activity
declare @PreviousActivityIds cursor
set @PreviousActivityIds=cursor for select ActivityId from @activity
open @PreviousActivityIds
fetch next from @PreviousActivityIds into @PreviousActivityId
while @@FETCH_STATUS=0
--fetch next from control_activities into @PreviousActivityId

begin
select @PreviousActivityType=ActivityType  from dbo.Activity  where
Activity.ActivityID=@PreviousActivityID

---join
 if (@PreviousActivityType='Join')
 begin
 select @count=sum(Quantity) from FoodTrace where ItemID=@ItemID
and Unit=@Unit and ActivityId=@PreviousActivityID
 if (@count<>0)
 raiserror('join sum quantity must be 0 in previous activity',11,1);
 end

---move
 if (@PreviousActivityType='Move')
 begin
 select    @count=sum(Quantity)    from    dbo.FoodTrace    where
ItemID=@ItemID               and               Unit=@Unit               and
ActivityId=@PreviousActivityID
 if (convert(integer,@count)<>0)
raiserror('move sum quantity must be 0 in previous activity',11,1);
end

---ItemChange
 if (@PreviousActivityType='ItemChange')
 begin
 select     @count=sum(Quantity)     from     dbo.FoodTrace     where
LotID=@LotID    and    Unit=@Unit    and    PlaceID=@PlaceID    and
ActivityId=@PreviousActivityID
 if (convert(integer,@count)<>0)
raiserror(' sum quantity must be 0 in previous ItemChangeactivity',11,1);
 end
---LotChange
 if (@PreviousActivityType='LotChange')
 begin
 select     @count=sum(Quantity)     from     dbo.FoodTrace     where
ItemID=@ItemID    and    Unit=@Unit    and    PlaceID=@PlaceID    and
ActivityId=@PreviousActivityID
 if (convert(integer,@count)<>0)
raiserror('sum quantity must be 0 in previous LotChange activity',11,1);
 end

---UnitChange
 if (@PreviousActivityType='UnitChange')
 begin
 select     @count=count(ItemID)     from     dbo.FoodTrace     where
ItemID=@ItemID and LotID=@LotID and PlaceID=@PlaceID and
Quantity<0 and ActivityId=@PreviousActivityID
 if (convert(integer,@count)=0)
raiserror('UnitChange activity count with negative quantities must not be
0 in previous activity',11,1);
 select     @count=count(ItemID)     from     dbo.FoodTrace     where
ItemID=@ItemID and LotID=@LotID and PlaceID=@PlaceID and
Quantity>0 and ActivityId=@PreviousActivityID
 if (convert(integer,@count)=0)
raiserror('UnitChange activity count with positive quantities must not be
0 in previous activity',11,1);
 end

---Produce
 if (@PreviousActivityType='Produce')
 begin
 select     @count=count(ItemID)     from     dbo.FoodTrace     where
PlaceID=@PlaceID            and            Quantity<0            and
ActivityId=@PreviousActivityID
 if (convert(integer,@count)=0)
raiserror('Produce activity count with negative quantities must not be 0 in
previous activity',11,1);
 select     @count=count(ItemID)     from     dbo.FoodTrace     where
PlaceID=@PlaceID            and            Quantity>0            and
ActivityId=@PreviousActivityID
 if (convert(integer,@count)=0)
raiserror('Produce activity count with positive quantities must not be 0 in
previous activity',11,1);
 end

fetch next from @PreviousActivityIds into @PreviousActivityId

end
close @PreviousActivityIds
end

--Existence
  select  @Count=sum(FoodTrace.Quantity) FROM FoodTrace where
LotID=@LotID   and  ItemID=@ItemID and  PlaceID=@PlaceID and
Unit=@Unit;
  if (@Quantity+@Count<0) raiserror( 'there is no enough product',11,1)

--Raw, Consume, Destruct and Recycle must be one record
select  @ActivityType=dbo.Activity.ActivityType  from  dbo.Activity
where dbo.Activity.ActivityID=@ActivityID;

if  (@ActivityType='Raw'   or   @ActivityType='Consume'   or
@ActivityType='Recycle' or @ActivityType='Destruct')
Begin

if  ((@ActivityType='Raw'   or   @ActivityType='Recycle')   and
@Quantity<0)
 raiserror('raw activity must have positive quantity',11,1);
if  ((@ActivityType='Destruct'  or  @ActivityType='Consume')  and
@Quantity>0)
raiserror('raw activity must have negative quantity',11,1);
end

Insert  into  dbo.FoodTrace  values (@PlaceID,  @LotID,  @ItemID,
@TagID, @Quantity, @Unit,@ActivityID)

end try

begin catch
select error_message()
goto son
end catch
son:
end

use FoodTraceDB
go
```

```
create trigger UpdateControl ON  [FoodTraceDB].[dbo].[FoodTrace]
for update
AS
begin
declare @PlaceID varchar(13);
declare @LotID varchar(50);
declare @ItemID varchar(13);
declare @TagID varchar(8);
declare @Quantity real;
declare @Count int;
declare @Unit varchar(10);
declare @ActivityID varchar(50);
declare @ActivityType varchar(12);
declare @say int;
select @PlaceID = i.PlaceID  from inserted i;
select @LotID = i.LotID  from inserted i;
select @ItemID = i.ItemID  from inserted i;
select @TagID = i.TagID  from inserted i;
select @Quantity = i.Quantity  from inserted i;
select @Unit = i.Unit  from inserted i;
select @ActivityID = i.ActivityID  from inserted i;

declare @food table([PlaceID] [varchar](13) not null,
          [LotID] [varchar](50) not null,
          [ItemID] [varchar](13) NOT  null,
          [Unit] [varchar](10)  not null)

declare @activity table([ActivityId] [varchar](50) )

insert into @food values(@PlaceID,@LotID,@ItemID,@Unit)

insert into @activity select ft.ActivityId from FoodTrace ft  join @food f
on
 f.ItemID=ft.ItemID and f.LotID=ft.LotID and f.PlaceID=ft.PlaceID and
f.Unit=ft.Unit where quantity<0

--select *  from @activity
select @say=count(*) from @activity

begin try

if (@say>0)
raiserror('There must not any activity after this activity',11,1);

Update  dbo.FoodTrace  set  PlaceID=@PlaceID,  LotID=@LotID,
ItemID=@ItemID, TagID=@TagID, Quantity=@Quantity, Unit=@Unit
where FoodTrace.ActivityID=@ActivityID
end try

begin catch
select error_message()
goto son
end catch
son:
end

use FoodTraceDB
go
create              trigger              [dbo].[DeleteControl]          ON
[FoodTraceDB].[dbo].[FoodTrace]
for delete
AS
 begin
declare @PlaceID varchar(13);
declare @LotID varchar(50);
declare @ItemID varchar(13);
declare @TagID varchar(8);
declare @Quantity real;
declare @Count int;
declare @Unit varchar(10);
declare @ActivityID varchar(50);
declare @ActivityType varchar(12);
declare @say int;
select @PlaceID = i.PlaceID  from inserted i;
select @LotID = i.LotID  from inserted i;
select @ItemID = i.ItemID  from inserted i;
select @TagID = i.TagID  from inserted i;
```

```
select @Quantity = i.Quantity  from inserted i;
select @Unit = i.Unit  from inserted i;
select @ActivityID = i.ActivityID  from inserted i;
declare @food table([PlaceID] [varchar](13) ,
          [LotID] [varchar](50) ,
          [ItemID] [varchar](13) ,
          [Unit] [varchar](10) )
declare @activity table([ActivityId] [varchar](50) )
insert into @food values(@PlaceID,@LotID,@ItemID,@Unit)
insert into @activity select ft.ActivityId from FoodTrace ft  join @food f
on
 f.ItemID=ft.ItemID and f.LotID=ft.LotID and f.PlaceID=ft.PlaceID and
f.Unit=ft.Unit where quantity<0
select @say=count(*) from @activity

begin try
if (@say>0)
raiserror('There must not any activity after this activity',11,1);
Delete dbo.FoodTrace where ActivityID=@ActivityID
end try
begin catch
select error_message()
goto son
end catch
son:
end

use FoodTraceDB
go
create procedure [dbo].[Lookup] @LotID [varchar](50), @ItemID
[varchar](13)  as
begin
select FoodTrace.PlaceID, FoodTrace.LotID, FoodTrace.ItemID,
FoodTrace.Unit, sum(FoodTrace.Quantity) as sum
FROM FoodTrace where LotID=@LotID  and ItemID=@ItemID
GROUP BY FoodTrace.PlaceID, FoodTrace.LotID,
FoodTrace.ItemID,FoodTrace.Unit having sum(FoodTrace.Quantity)>0
end

use FoodTraceDB
go
create procedure [dbo].[Tracing]
 @itemid [varchar](13), @lotid [varchar](50),  @placeid [varchar](13),
@unit [varchar](10)
as
begin
declare @food table([PlaceID] [varchar](13) not null,
          [LotID] [varchar](50) not null,
          [ItemID] [varchar](13) NOT  null,
          [Unit] [varchar](10)  not null)

declare @activity table([ActivityId] [varchar](50) )
declare @activitytrace table([ActivityId] [varchar](50) )
declare @say int

insert into @food values(@placeid,@lotid,@itemid,@unit)
--select * from @food
basla:
delete @activity
insert into @activity select ft.ActivityId from FoodTrace ft  join @food f
on
 f.ItemID=ft.ItemID and f.LotID=ft.LotID and f.PlaceID=ft.PlaceID and
f.Unit=ft.Unit where quantity>0
except
select * from @activitytrace
select @say=count(*) from @activity

if (@say>0)
begin
insert into @activitytrace select * from @activity
delete  @food
insert into @food select PlaceID, LotID,ItemID,Unit from FoodTrace ft
 join @activity a on a.ActivityId=ft.ActivityId where ft.Quantity<0
goto basla
end
```

```sql
select  f.ItemID, f.LotID, f.PlaceID, f.Unit,
f.Quantity,f.ActivityId,aa.ActivityType from FoodTrace f join
@activitytrace a on f.ActivityId=a.ActivityId join Activity aa on
aa.ActivityId=a.ActivityId
end


use FoodTraceDB
go
create procedure [dbo].[Tracking]
 @itemid [varchar](13), @lotid [varchar](50), @placeid [varchar](13),
@unit [varchar](10)
as

begin
declare @food table([PlaceID] [varchar](13) not null,
            [LotID] [varchar](50) not null,
            [ItemID] [varchar](13) NOT null,
            [Unit] [varchar](10)  not null)

declare @activity table([ActivityId] [varchar](50) )
declare @activitytrace table([ActivityId] [varchar](50) )
declare @say int

insert into @food values(@placeid,@lotid,@itemid,@unit)
--select * from @food
basla:
delete @activity
insert into @activity select ft.ActivityId from FoodTrace ft  join @food f
on
 f.ItemID=ft.ItemID and f.LotID=ft.LotID and f.PlaceID=ft.PlaceID and
f.Unit=ft.Unit where quantity<0
except
select * from @activitytrace

--select *  from @activity
select @say=count(*) from @activity

if (@say>0)
begin
insert into @activitytrace select * from @activity
delete  @food
insert into @food select PlaceID, LotID,ItemID,Unit from FoodTrace ft
 join @activity a on a.ActivityId=ft.ActivityId where ft.Quantity>0
goto basla
end

select  f.ItemID, f.LotID, f.PlaceID, f.Unit, f.Quantity,f.ActivityId from
FoodTrace f join @activitytrace a on f.ActivityId=a.ActivityId
end


use FoodTraceDB
go
insert [dbo].[ActivityType] ([ActivityType]) values (N'Consume')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Destruct')
insert [dbo].[ActivityType] ([ActivityType]) values (N'ItemChange')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Join')
insert [dbo].[ActivityType] ([ActivityType]) values (N'LotChange')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Move')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Produce')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Raw')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Recycle')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Transfer')
insert [dbo].[ActivityType] ([ActivityType]) values (N'Split')
insert [dbo].[ActivityType] ([ActivityType]) values (N'UnitChange')

--Lot Activities added not to get error
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act1', N'Raw',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act10', N'Move',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act11', N'Consume',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act14', N'Join',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act2', N'Move',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act3', N'Produce',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act4', N'UnitChange',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act5', N'Move',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act6', N'Join',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act7', N'Move',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act8', N'Produce',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Act9', N'Move',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Lot', N'Raw',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Lot1', N'Raw',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
```

```
[ModifiedBy], [ModifiedDate]) values (N'Lot2', N'Raw',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Lot3', N'Raw',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Lot4', N'Raw',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20918983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
insert [dbo].[Activity] ([ActivityID], [ActivityType], [Responsible],
[FirmId], [DatabaseId], [DateTime], [CreatedBy], [CreatedDate],
[ModifiedBy], [ModifiedDate]) values (N'Lot5', N'Raw',
N'20908983144', N'1533154', N'1522466', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), N'20908983144', CAST(N'2018-03-07
14:24:00' AS SmallDateTime), null, null)
--Lot Activities added not to get error
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place1', N'Act1', N'Milk',
N'X',   10, N'kg', N'Act1')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place1', N'Act1', N'Milk',
N'X',   -10, N'kg', N'Act2')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place2', N'Act1', N'Milk',
N'X',   10, N'kg', N'Act2')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place2', N'Act1', N'Milk',
N'X',   -10, N'kg', N'Act3')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place2', N'Act3', N'Milk',
N'X',   480, N'kg', N'Act3')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place2', N'Act3', N'Milk',
N'X',   -480, N'kg', N'Act4')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place2', N'Act3', N'Milk',
N'X',   464, N'l', N'Act4')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place2', N'Act3', N'Milk',
N'X',   -464, N'l', N'Act5')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place3', N'Act3', N'Milk',
N'X',   464, N'l', N'Act5')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place3', N'Lot', N'Milk', N'X',
44536, N'l', N'Lot')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot1', N'Rennet',
N'Tag1',  130, N'kg', N'Lot1')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot2', N'Salt',
N'Tag2',  23, N'kg', N'Lot2')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot3', N'Wheyey',
N'Tag3',  1300, N'l', N'Lot3')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot4', N'Wrap',
N'Tag4',  1, N'kg', N'Lot4')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot5', N'Cans',
N'Tag5',  15, N'pcs', N'Lot5')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place3', N'Act3', N'Milk',
N'X',   -464, N'l', N'Act6')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place3', N'Act6', N'Milk',
N'X',   45000, N'l', N'Act6')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place3', N'Lot', N'Milk', N'X',
-44536, N'l', N'Act6')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place3', N'Act6', N'Milk',
N'X',   -1300, N'l', N'Act7')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Act6', N'Milk',
N'X',   1300, N'l', N'Act7')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot5', N'Cans',
N'Tag5',  -15, N'pcs', N'Act8')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Act6', N'Milk',
N'X',   -1300, N'l', N'Act8')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot1', N'Rennet',
N'Tag1',   -130, N'kg', N'Act8')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot2', N'Salt',
N'Tag2',   -23, N'kg', N'Act8')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot3', N'Wheyey',
N'Tag3',   -1300, N'l', N'Act8')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Lot4', N'Wrap',
N'Tag4',   -1, N'kg', N'Act8')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Act8', N'Cheese',
N'Tag6',  15, N'pcs', N'Act8')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place4', N'Act8', N'Cheese',
N'Tag6',   -15, N'pcs', N'Act9')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place5', N'Act8', N'Cheese',
N'Tag6',  15, N'pcs', N'Act9')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place5', N'Act8', N'Cheese',
N'Tag6',   -15, N'pcs', N'Act10')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place6', N'Act8', N'Cheese',
N'Tag6',  15, N'pcs', N'Act10')
insert [dbo].[FoodTrace] ([PlaceID], [LotID], [ItemID], [TagID],
[Quantity], [Unit], [ActivityId]) values (N'Place6', N'Act8', N'Cheese',
N'Tag6',   -2, N'pcs', N'Act11')
```