Ultimate Adventures Management System

(*MSSQL + MongoDB Implementation*)

Created by: Andrew Lane

04/04/16

CS3550

[C:\Users\a500806\Downloads\github_logo.png](https://github.com/theandrewlane/UAMS)

[Visit the Repository](https://github.com/theandrewlane/UAMS)

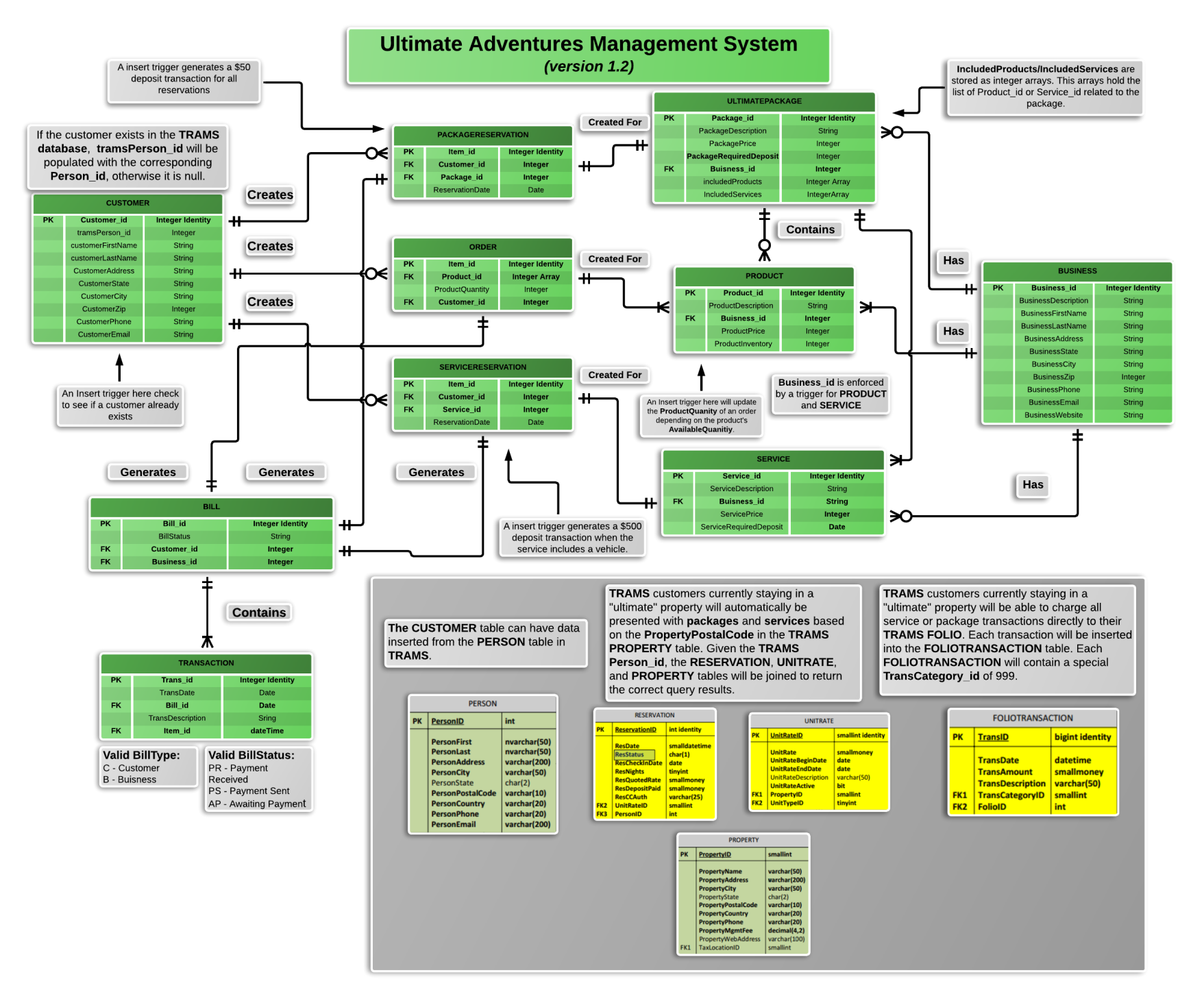
Part 1: Project Overview

Ultimate Adventures is every vacationer’s go-to platform for crafting their dream adventure. UA’s team of expert adventurers scoured the planet for the best businesses in the industry. We’ve grown personal relationships with these “ultimate” businesses, and developed thousands of partnerships worldwide. We’re rapidly growing, and adding new partnerships every week! We leverage these “ultimate” partnerships to package all kinds of goods/services together. All of our packages come with our low price/no hassle “ultimate” guarantee. Our customers have the option to pick their adventurous goods/services al-la-carte too!

Customers choose UA because they know all of our goods/services are “ultimate approved”, and we don’t have any hidden fees. We take the headache out of adventure booking because we’ve streamlined the entire process – Each good/service you want is offered through a different business? No problem. There’s only one payment, and we take care of everything else. The activities you want require you to sign disclosures/agreements? No problem. We’ve digitized everything and allow you to sign all documents electronically. If you happen to be vacationing at one of our “ultimate” vacation properties, it’s even easier! Each property location is associated with a variety of “ultimate” packages – because you’re staying at an “ultimate” property we give you an “ultimate” discount, simplify the booking process, and also provide the option of billing your vacation property directly.

**Overview of the UAMS Data Tier:**

The core functionality of UAMS is to provide customers the ability to search/book various “ultimate” goods and services (from a variety of partnered businesses). UAMS can store customer information, produce detailed financial statements, and distribute customer payments across all involved “ultimate” businesses. UAMS has partnered with the Timeshare Reservation Accounting Management System (TRAMS), to simplify the booking process for customers staying at an “ultimate” property. Leveraging TRAMS, UAMS can autofill customer information; automatically show goods/services affiliated with the “ultimate” property, and also bill the property directly. If a customer stays at “ultimate” properties frequently, UAMS will give the customer a bigger discount.

**Part 2: Data Specifications Overview**

**Part 2: Data Specifications Overview (*continued*)**

**Business Rules Satisfied by ERD and Normal Constraints:**

* All primary keys have identity values
* Tax is built in to every service, ultimate package, and product.

**Business Rules Satisfied by Triggers (all defined in MongoDB):**

1. A $50 deposit is required for all reservations.

* **Trigger Name: tr \_addDepositForReservation**
* **Trigger Type:** AFTER trigger
* **Trigger Description:** A $50 deposit is required at checkout for **each** **different** vehicle (quad, jet ski, snow mobile, etc.). If a TRANSACTION is created from a UTLIMATEPACKAGE entry, this trigger will call the **addVehicleDeposit** stored procedure for each vehicle rental.

1. A $500 deposit is required at checkout for all vehicles (quad, jet ski, snow mobile, etc.)

* **Trigger Name: tr\_addDepositForVehicle**
* **Trigger Type:** AFTER trigger
* **Trigger Description:** A $500 deposit is required at checkout for all vehicles (quad, jet ski, snow mobile, etc.). If a TRANSACTION is created from a SERVICERESERVATION, this trigger will call the addVehicleDeposit stored procedure which will add a new deposit transaction to the BILL.

3. The product quantity of an order cannot be greater than the product inventory.

* **Trigger Name: tr \_checkStock**
* **Trigger Type:** INSERT trigger
* **Trigger Description:** When an order is created, the ***ProductQuantity*** is checked against PRODUCT’s ***ProductInventory***. If ***ProductQuantity***is less than ***AvailableQuantity****,* we call the **updateOrderQuantity** stored procedure which inserts the ***ProductQuantity***value as the ***ProductInventory***value.

4. A customer cannot have multiple CustomerID

* **Trigger Name: tr \_checkExistingCustomer**
* **Trigger Type:** INSERT trigger
* **Trigger Description:** When a new customer is added, we double check to see if a customer with that **CustomerEmail** already exists, if so throw an error.

**Business Rules Satisfied by Stored Procedures/User Defined Functions (FUNCTIONS/UDFS are the same in MongoDB):**

1. Businesses can update product inventory, and placed orders can update product inventory.

* **Function Name:** [***updateInventory()***](https://github.com/theandrewlane/UAMS/blob/master/scripts/update-inventory.js)
* **Function Description:** This function is designed to be called after an order has placed, or when a business has restocked – It will update a product’s inventory amount accordingly.
* **Input Parameter 1: *ProductId*** (integer) – This is the corresponding **ProductID**, which will be updated.
* **Input Parameter 2: *Inventory*** (integer) – This is a positive or negative integer. The value of this parameter will be added/subtracted from ***productInventory***.

2. Customers vacationing at an “ultimate” property, receive a 5% discount. Customers vacationing at an “ultimate” property who have booked with UA more than twice receive a 10% discount.

* **Function Name:** [***addUltimateDiscount()***](https://github.com/theandrewlane/UAMS/blob/master/scripts/add-ultimate-discount.js)
* **Function Description:** This function is designed to be called if [***getTramsVacations****()*](https://github.com/theandrewlane/UAMS/blob/master/scripts/get-trams-vacations.js)returns an integer greater than zero. If the return value of [***getTramsVacations****()*](https://github.com/theandrewlane/UAMS/blob/master/scripts/get-trams-vacations.js)is less than 2, reduce bill amount by 5%. If the return value is greater than 2, reduce the bill amount by 10% *(called via trigger when transaction is added to bill).*
* **Input Parameter:** **Vacations** (integer) – The amount of TRAMS vacations
* **Returns:** Boolean on successful bill update.

**3.** Customers vacationing at an “ultimate” property, receive a 5% discount. Customers vacationing at an “ultimate” property who have booked with UA more than twice receive a 10% discount.

* **Function Name:** [***getTramsVacations****()*](https://github.com/theandrewlane/UAMS/blob/master/scripts/get-trams-vacations.js)
* **Function Description:** This function queries **TRAMS** – Given the ***PersonID***, this function will count the occurrences in the reservation table. This function also relies on the helper function [***customerExistsInTrams()***](https://github.com/theandrewlane/UAMS/blob/master/scripts/customer-exists-in-trams.js)***.*** If this function doesn’t return a ***PersonID***,[***getTramsVacations****()*](https://github.com/theandrewlane/UAMS/blob/master/scripts/get-trams-vacations.js)will log an error.
* **Input Parameter:** ***PersonID*** *(integer)* – The TRAMS ***PersonID***
* **Returns:** An Integer representation which reflects the Customer’s total amount of TRAMS vacations.

**4.** Customers can view bills related to them.

* **Function Name:** [***produceCustomerBill****()*](https://github.com/theandrewlane/UAMS/blob/master/scripts/produce-customer-bill.js)
* **Function Description:** Given a customer’s name *<first last>*, produce a bill (*stdout*) listing the customer details, transaction count, transaction details, and the total amount due.
* **Input Parameter:** ***customerName*** *(String)* – The customer’s name *<first last>*
* **Returns:** The customer’s bill, via console output.

**5.** Customers that exist in the TRAMS database can opt to have their customer information copied over to the Customer Table.

* **Function Name:** [***InsertTRAMSPersonData()***](https://github.com/theandrewlane/UAMS/blob/master/scripts/insert-trams-person-data.js)
* **Function Description**: Given a ***PersonId*** from TRAMS, add all related data to **Customer**.
* **Input Parameter:** **TRAMSPersonID** (*integer*)
* **Returns:** Inserts a new customer

**6.** If the customer wants to have the charge sent to their **TRAMS** Folio.

* **Function Name:** [***sendTransactionToTRAMS()***](https://github.com/theandrewlane/UAMS/blob/master/scripts/send-transaction-to-TRAMS.js)
* **Function Description:** If a customer exists in TRAMS and has a FOLIO, create a new FOLIOTransaction for that customer containing all the details from the transaction. Remove the transaction from the UAMS database.
* **Input Parameter:** ***transID*** (*integer*), ***customerID*** (integer)
* **Returns:** The **TRAMS** ***TransID*** and the **TRAMS** ***FolioID***, along with a success message *(stdout)*.

**7.** The system can create transactions when necessary

* **Function Name:** [***createBillTransaction()***](https://github.com/theandrewlane/UAMS/blob/master/scripts/create-bill-transaction.js)
* **Function Description:** Given a BillID, create a transaction. The timestamp for this transaction will be generated when it is created.
* **Input Parameter:** ***BillID*** (integer), ***transactionDescription*** (string), ***transactionAmount*** *(integer)*,
* **Optional Parameter: *BillDescription*** (*string*)
* **Returns:** Adds a new transaction to the specified bill

**8. Check whether a customer exists in TRAMS (this is a helper function)**

* **Function Name:** [***customerExistsInTrams*()**](https://github.com/theandrewlane/UAMS/blob/master/scripts/customer-exists-in-trams.js)
* **Function Description:** Given a customer’s name <first last> check whether or not they exist in the **TRAMS** **Person** table.
* **Input Parameter:** ***customerName*** (String) – The customer’s name *<first last>*
* **Returns:** The TRAMS PersonID if the customer exists in the database, otherwise this will return null;