Ad Flyer Page Creation

Daniel Cona, Nathan Cummings, Donte Black, Jacob Meyer, Carl Dunn

Group #2

Table of Contents

Introduction

This document summarizes the requirements, analysis, and design of a database system designed to support the needs of a small chain of grocery stores that wants to publish Sunday Ad Flyers.

Project Specification

Customer Name: ICN

Summary of Need: ICN runs a small chain of grocery stores and wants to improve the workflow and reporting associated with their Sunday Ad Flyers.

Motivation or Reason for Need: The current process is not efficient, and does not provide the ability to generate business intelligence reporting.

Nature of Business: The purpose of the flyer is to improve sales and revenue, improve customer loyalty, and help to plan inventory.

Workflow: The Ad Flyer is a physical and digital piece of media which is created weekly. Each Sunday a new Ad Flyer is placed in-stores and mailed to residents within a certain radius of each store. Each Ad Flyer has a name and unique alphanumeric code to identify the Ad Flyer. It also has a start and end date for when the sales in the Ad Flyer are good for. An Ad Flyer also has a type such as weekly or 3-day Ad Flyer. Typically, this is one week, Sunday to Saturday, however they do run specials such as on Black Friday which do not run for a whole week. As well as longer running Ad Flyers for newly opened stores.

Each Ad Flyer contains some number of pages. On an average week ICN produces Ad Flyers with 6 pages, but this can go up and down depending on the offers for that week. Each page has a height and width for printing and an order to ensure the pages show correctly in the final product. ICN has a planning team which determines how much space each department of the store gets to show of their offers. The planning team does this by creating boxes or sections on a page in a grid-like layout. Each box is assigned a name to be easily identifiable and a department. This lets the department know which box is theirs. The boxes or sections are of varying size and must be tracked for reporting purposes.

ICN creates offers which represent the 'on sale' items customers will see in the store. For example, "$1.00 off Potato Chips". Each offer has a name and a unique numeric identifier. It has a status for tracking purposes, and a type. It also holds the discount type and discount amount. For example, it could be $1 off where dollar is the type, 1 is the amount or it could be 50% off where percent is the type and 50 is the amount. Each offer also has a list of items for which the discount is applied to. Each item has a unique numeric identifier, a name, a cost, a retail, and a supplier. The 'on sale' price should also be tracked in the system for each item.

Statement of Customer-Perceived Requirements: The current Ad Flyer process will be implemented in a database system which will store Ad Flyer data allow the creation of reports to help ICM manage and plan their business.

Statement of Objectives

- Implement the Ad Flyer workflow in a database system
- Create reports based on queries to provide insight to the Ad Flyer process.

Requirements

1	Users shall create Ad Flyers
2	A user must associate an Ad Flyer with one or more stores
3	A user must select a start date for the Ad Flyer
4	A user must enter an end date for the Ad Flyer
5	A user shall create one or more pages within an Ad Flyer
6	A user must select a page number for each page in an Ad Flyer
7	A user shall be able to enter a page height and a page width
8	When a page is created, the page height shall be set to 11 inches, and the width shall be set to 8.5 inches by default.
9	The planning team shall create and name boxes or sections of varying sizes on pages
10	The planning team shall assign boxes or sections on pages to departments
11	A user must be able to select a department from a constrained list of departments.
12	A user must be able to enter the name, status, offer department, discount type, and discount amount to represent an Offer.
13	A user shall be able to select 'dollar' or 'percent' for the discount type
14	If the discount type is dollar, the discount shall be calculated as the discount amount in dollars.
15	If the discount type is percent, the discount shall be calculated as the discount amount as a percentage.
16	The list of offer statuses shall be a constrained list of 'Draft, Pending Approval, Approved, and Rejected.
17	When an offer is created, the status shall be set to 'Draft' by default
18	A user must be able to associate a list of items to include in the offer. Each item in the offer list has a unique numeric identifier, a name, a cost, a retailer, and a supplier.
19	A user must be able to assign one offer to a box or section on a page
20	The user who created the offer must be tracked for reporting
21	The date the offer was created must be tracked for reporting
22	The on sale price shall be calculated as the item cost – discount amount.
23	The on sale price shall be tracked in the system for each item.
24	The margin shall be calculated as the on sale price – item cost
25	The User Id in offer is a separate table that contains the name and Id of the User who created the offer
Report 1	Offers Report: List all offers which are approved and start within the next 3 weeks. Show the offer name, number of items in the offer, discount type, discount amount, the user who created the offer, when the offer was created, and the offer department
Report 2	Box or Section Report: For a specific Ad Flyer, list each box with its name and the name of the offer, if it has one, that is tied to the box or section
Report 3	Item Report: For a specific Ad Flyer, list all the items inside of the Ad Flyer. Including the offer number and name, item number, description, cost, 'on-sale' price, and margin.
Report 4	Ad Flyer Department Page Report: For a specific Ad Flyer, show the percentage of area each department has for the Ad Flyer. Calculate the area of each of the boxes or sections a department has, and divide it by the total area of the pages in the Ad flyer
Report 5	Late Offers Report: List all offers that are not approved within 6 weeks. Show the offer name, number of items in the offer, discount type, discount amount, offer department, the user who created the offer, and when the offer was created.
Report 6	Department Discount Report: For an ad flyer, list each department with content in the Ad Flyer,

	and for each department list each offer name, the total cost of each offer, the discount for each offer, and the margin for each offer.
Report 7	Ad Flyer Department Trend Page Report: For all Ad Flyers over the past year, show the percentage of area each department has for the Ad Flyers. Calculate the area of each of the boxes or sections a department has, and divide it by the total area of the pages in the Ad flyer.
Report 8	Department Discount Trend Report: For all Ad Flyers over the past year, list each department with content in the Ad Flyers, and for each department list the number of offers, the total cost the offers, the discount for the offers, and the margin for the offers, grouped by month.
Report 9	List all Ad Flyers that are scheduled to start in the next 2 weeks that have offers that are not approved. For each Ad Flyer with not approved offers, list the Ad Flyer, the Ad Flyer Start Date, for each not approved offer in the Ad Flyer, list the offer name, number of items in the offer, discount type, discount amount, offer department, the user who created the offer, and when the offer was created.

Design

Schema

ITEM (Name: STRING, Cost: INT, Supplier: STRING, Retailer: STRING, Item_ID: STRING references LISTOF, Sale: INT)
Key: {Item_ID}

OFFER (Name: STRING, Status: STRING, Type: STRING, Date: DATE, Dept: STRING, Amount: STRING, User_ID: STRING, Offer_ID: STRING)
Key: {Offer_ID}

BOX (Size: INT, Row: INT, Column: INT, Dept: STRING, Box_ID: STRING)
Key: {Box_ID}

PAGE (Height: INT, Width: INT, Total_Col: INT, Total_Row: INT, Num: INT, Page_ID: STRING)
Key: {Ad_ID}

ADFLYER (Name: STRING, SDate: DATE, EDate: DATE, NumDays: INT, Ad_ID: STRING)
Key: {Ad_ID}

ListOf (Item_ID: STRING references ITEM, Offer_ID: STRING references OFFER)
Key: {Item_ID, Offer_ID}

HasOne (Box_ID: STRING references BOX, Offer_ID: STRING references OFFER)
Key: {Box_ID, Offer_ID}

Contains (Box_ID: STRING references BOX, Page_ID: STRING references PAGE)
Key: { Box_ID, Page_ID }

Has (Ad_ID: STRING references ADFLYER, Page_ID: STRING references PAGE)
Key: {Ad_ID, Page_ID}

Data Dictionary

Table: Item

Field Name	Datatype	Description	Example
Name	String	Name of the item	Banana
Cost	Int	Price of the item	2
Supplier	String	The supplier of the item	John’s Fruit Farm
Retailer	String	The people who will sell the item	Wegman’s
Item_ID	STRING	Unique alphanumeric Id of the item	12532
Sale	String	Description of the sale	None

Table: Ad_Flyer

Field Name	Datatype	Description	Example
AD_ID	STRING	Unique alphanumeric Id of the flyer	19g
Name	String	Name of the flyer	Black Friday
SDate	Date	This is the first date the flyer will be valid	5/7/21
EDate	Date	This is the last date the flyer will be valid	6/7/21
NumDays	Int	The number of the days that the flyer is public for	30

Table: Offer

Field Name	Datatype	Description	Example
Offer_ID	STRING	Unique alphanumeric Id of the offer	5425646
Name	String	Name of the offer	Chips
Dept	String	The name of the department that made the offer	Whole Foods
Type	String	Type of offer (whether amount/percent off)	percent
Status	String	Whether sale is active or inactive	Pending
Amount	Int	Amount of the offer	15
User_ID	Int	User_Id of whom created the offer	John324
Date	Date	Date of the offer	6/3/21

Table: Page

Field Name	Datatype	Description	Example
Page_ID	STRING	Alphanumeric Id of the page	G583
Num	Int	Number of the page on the flyer	12
Height	Int	Height of the page	12
Width	Int	Width of the page	6
Total_Col	Int	Total amount of columns on the page	3
Total_Row	Int	Total amount of rows on the page	3

Table: Box

Field Name	Datatype	Description	Example
Name	String	Name of the box, references CONTAINS	Top left
Dept	String	Department that made the offer	Sea Food
Row	Int	What row it is in	2
Box_ID	STRING	Unique alphanumeric Id of the box	546hj

Relation: LISTOF

Field Name	Datatype	Description	Example
Offer_ID	STRING	References from Offer(Id)	567g3567
Item_ID	STRING	References from Item(Id)	5257s4

Relation: HAS

Field Name	Datatype	Description	Example
Ad_ID	STRING	References from AdFlyer(Id)	5622h46
Page_ID	STRING	References from Page(Id)	7611c345

Relation: CONATINS

Field Name	Datatype	Description	Example
Page_ID	STRING	References from Page(Id)	998w334
Box_ID	STRING	References from Box(Id)	098t2134

Relation: HASONE

Field Name	Datatype	Description	Example
Offer_ID	STRING	References from Offer(Id)	1273q56
Box_ID	STRING	References from Box(Id)	235z74

Constraints

Table: Item

Field Name	Constraint
Name	Not Null
Cost	Between $0 - $1000000
Supplier	Not Null
Retailer	
Item_Id	Primary Key
Sale	Not Null

Table: Ad_Flyer

Field Name	Constraint
AD_ID	Primary Key
Name	Not Null
SDate	Not Null
EDate	Not Null
NumDays	<365

Table: Offer

Field Name	Constraint
Offer_ID	Primary Key
Name	Primary Key
Dept	Not Null
Type	“Percent” or “Amount”
Status	“Draft” or “Pending” or “Approved” or “Denied”
Amount	Cannot be more then 20% off of the price
User_ID	Unique

Date	Not Null

Table: Page

Field Name	Constraint
Ad_ID	Primary Key
Num	Int AND less then 16 pages
Height	Table Check
Width	Table Check
Total_Col	Int
Total_Row	Int

Table: Box

Field Name	Constraint
Name	Primary Key
Dept	Not Null
Size	At least 3”
Box_ID	Primary Key

Relation: LISTOF

Field Name	Constraint
Offer_ID	Primary Key
Item_ID	Primary Key

Relation: HAS

Field Name	Constraint
Ad_ID	Primary Key

Page_ID	Primary Key

Relation: HASONE

Field Name	Constraint
Offer_ID	Primary Key
Box_ID	Primary Key

Relation: CONTAINS

Field Name	Constraint
Page_ID	Primary Key
Box_ID	Primary Key

Queries

The following queries will satisfy the project requirements:

1) Create an Ad Flyer using attributes as per Ad_Flyer schema

2) Create a Page for the ad flyer as per the Page schema

3) Create boxes on the page as per the Box schema

4) Create an offer for each box as per the Offer schema

5) Give an offer an item as per the Item schema

6) Assign a start and end date to each ad flyer

7) Assign the dimensions for each page

8) A department will be assigned for each offer created

9) A newly created offer will start in the draft phase

10) The price of an item will be the cost minus the discount

11) Display the approved offers that start soon

12) Display the offer attributes for the approved offers: name, number of items, type, amount, user id, offer id, department, date

13) Display the boxes that belong to a page with the; row, column, size, offer id, box id, and department

14) Display the items apart of an ad flyer with their attributes; name, cost, supplier, retailer, sale type, and item id

15) Divide the size of a box by the height and width to get the total rows and columns, each box takes up one of those rows and columns

16) Display the offers not approved within a certain time frame with their attributes; name, number of items, type, amount, department, user id, date

17) Display the departments associated with a box id, for those departments list the offers name, total cost, discount, and margin

18) Display, over the past year, the percentage of area each department by taking the area of that departments boxes and dividing it by the total area of the ad flyers pages

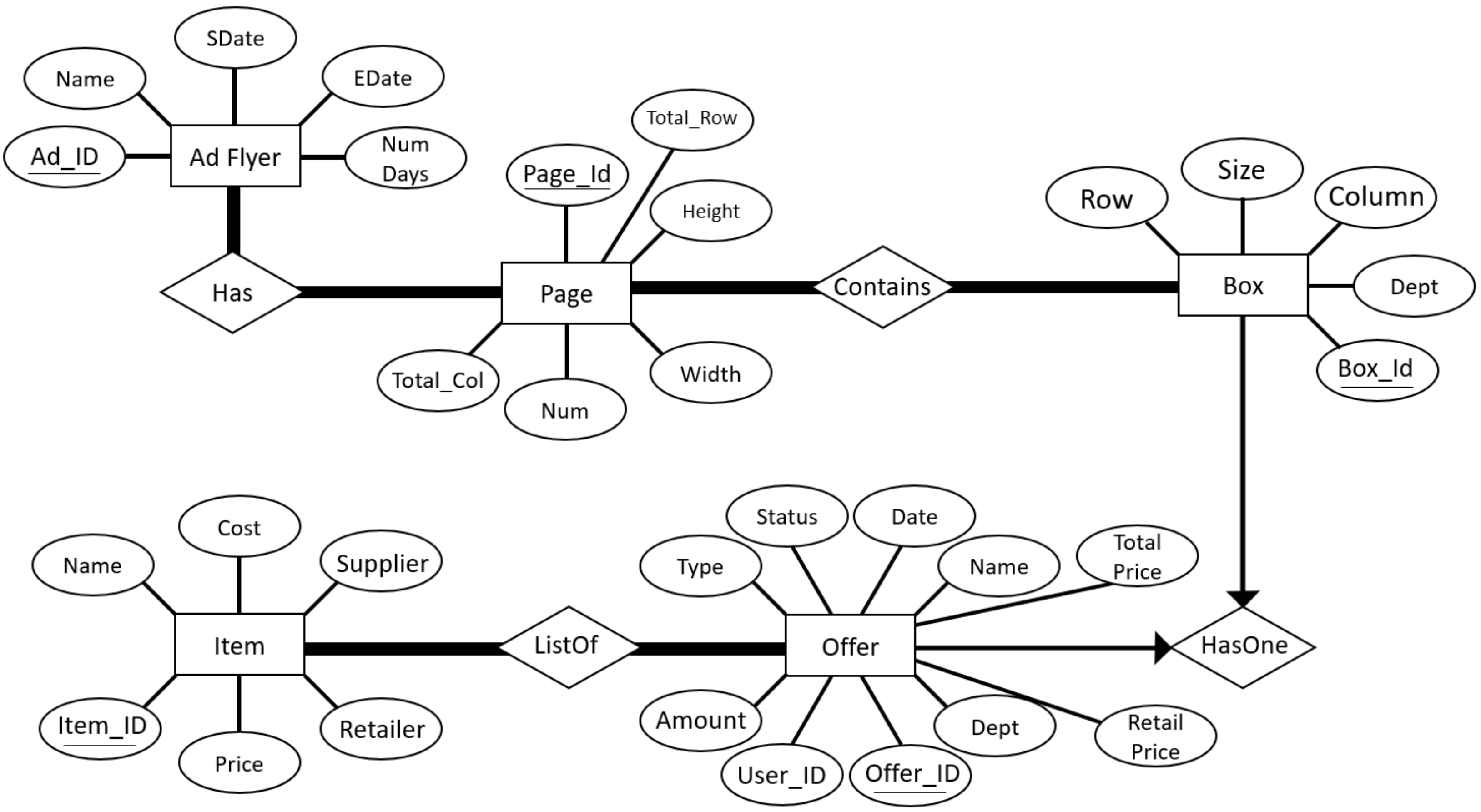
19) Display, over the past year, a department with offers in the ad flyer and the number of offers, total cost of the offers, discount on the offers, margin for the offers, grouped in months

20) Display the ad flyers whom start date is within two weeks and has pending offers, list their start date, ad flyer is belongs to, as well as, the pending offers name, number of items in it, type, amount, department, user id, date

Requirements Matrix

Report	1	2	3	4	5	6	7	8
Req 1	X	X	X	X		X	X	X
Req 2	X	X	X	X		X	X	X
Req 3	X	X	X	X		X	X	X
Req 4	X	X	X	X		X	X	X
Req 5	X	X	X	X		X	X	X
Req 6		X	X	X		X	X	X
Req 7		X		X			X	
Req 8		X		X			X	
Req 9		X	X	X		X	X	X
Req 10		X	X	X		X	X	X
Req 11		X	X	X	X	X	X	X
Req 12	X	X	X		X	X		X
Req 13	X		X			X		X
Req 14			X			X		X
Req 15			X			X		X
Req 16	X					X		X
Req 17	X				X			
Req 18		X	X		X	X		X
Req 19		X	X			X		X
Req 20	X				X			
Req 21	X				X			
Req 22			X			X		X
Req 23			X			X		X
Req 24			X			X		X
Req 25	X				X			

Entity-Relationship Diagram



Test Plan

Test Case	Expected Result	Testing
Create Ad Flyer Record. Do not associate a store	Error	R2, R3, R4
Associate Ad Flyer with Store	No Error	R4
Create an ad Flyer Page Do not enter a page number	Error	R5, R6 ,R7 ,R8
Enter a page number	No Error	R6
Validate that the default page size is 8.5x11	No Error	R8
Update the page size	No Error	R7
Create and name boxes or sections of varying sizes on the page.	No Error	R9
Associate a box or section on a page with an invalid department	Error	R10, R11
Associate a box or section on a page with an valid department	No Error	R10, R11
Create an offer with an invalid discount type	Error	R12, R13
Create an offer with a valid discount type Validate that the default status is set to 'Draft'	No Error	R12, R13, R17
Try to update an offer with an invalid status	Error	R16
Associate an invalid list of items with the offer.	Error	R18
Associate a valid list of items with the offer.	No Error	R18
Associate one offer to a box or section on a page	No Error	R19
Associate more than one offer to a section or box on a page.	Error	R19
Run offers report with offers that are not approved	Error, no records returned	Report 1
Run offers report with offers that are approved but do not start within 3 weeks	Error, no records returned	Report 1
Run offers report with offers that are approved and that start within 3 weeks	No Error	Report 1
Run Box or Section Report with a Flyer that has no pages	Error, no records returned	Report 2
Run Box or Section Report with a Flyer that has pages but no offers	No Error, records returned but no offers displayed	Report 2
Run Box or Section Report with a flyer that has pages and that has offers	No error	Report 2
Run Item Report with a Flyer that has no items	Error, no records returned	Report 3
Run Item Report with a Flyer that has items	No Error	Report 3

Run Ad Flyer Department Report	No Error – validate that the calculations are correct	Report 4
Run Late Offers Report with percent discount type	No Error – validate that the discount is calculated correctly	Report 5, R20, R21, R22, R23, R15
Run Late Offers Report with dollar discount type	No Error – validate that the discount is calculated correctly	Report 5, R20, R21, R22, R23, R14
Run Late Offers Report with all offers within the next 6 weeks approved	Error – no records returned	Report 5
Run Department Discount Report	No Error	Report 6, R24
Run Department Discount Report with an invalid Ad Flyer ID	Error	Report 6
Run Ad Flyer Department Trend Page Report with less than a year’s data	No Error – only the months with data available will be listed.	Report 7
Run Ad Flyer Department Trend Page Report with more than a year’s data	No Error – only the latest 12 months of data will be listed.	Report 7
Run Department Discount Trend Report with less than a year’s data	No Error – only the months with data available will be listed.	Report 8
Run Department Discount Trend Report with more than a year’s data	No Error – only the latest 12 months of data will be listed	Report 8

Implementation

For our implementation we used the administrative UI PhpMyAdmin. Here we added all of our tables, constrains, and triggers. We also tested our test plan to make sure it works correctly. We also get screenshots of the tables that it created.

Queries in SQL

The SQL queries along with the constraints were tested to ensure that they worked and fit the needs of the project statement. AdFlyer has a constraint to prevent the number of days a flyer is good or from going over a year.

```
CREATE TABLE AdFlyer (

Ad_ID VARCHAR (128),

Name VARCHAR (128) NOT NULL,

SDate DATE NOT NULL,

EDate DATE NOT NULL,

NumDays INT,

PRIMARY KEY (Ad_ID),

CHECK (Numdays < 365) )
```

```
CREATE TABLE Page (

Page_ID VARCHAR (128),

Num INT NOT NULL,

Height DECIMAL (10,2) NOT NULL,

Width DECIMAL (10,2) NOT NULL,

Total_Row INT NOT NULL,

Total_Column INT NOT NULL,

PRIMARY KEY (Page_ID),

CHECK (Num < 16) )
```

A constraint was added to the Box table to ensure that the size of a box is at least three inches.

```
CREATE TABLE Box (

Box_ID VARCHAR (128),

Dept VARCHAR (128) NOT NULL,

BoxRow INT NOT NULL,

BoxColumn INT NOT NULL,

Size INT NOT NULL,

PRIMARY KEY (Box_ID),

CHECK (Size > 3) )
```

The ENUM declaration for Status and Type can be seen as well as a constraint to prevent a discount from going over 20%.

```
CREATE TABLE Offer (

Offer_ID VARCHAR (128),

User_ID VARCHAR (128) NOT NULL UNIQUE,

Name VARCHAR (128) NOT NULL,

Dept VARCHAR (128) NOT NULL,

Status ENUM ('Draft', 'Pending', 'Approval', 'Approved',
'Rejected') NOT NULL,

Type ENUM ('Dollar', 'Percent') NOT NULL,

Amount INT NOT NULL,

Date DATE NOT NULL,

PRIMARY KEY (Offer_ID),

CHECK (Type = ‘Percent’ AND Amount < 20) )
```

```
CREATE TABLE OfferSalePrice (

Offer_ID VARCHAR (128),
SalePrice DECIMAL (10, 2),

RetailPrice DECIMAL (10, 2),

Margin DECIMAL (10, 2),

Cost DECIMAL (10, 2),

PRIMARY KEY (Offer_ID),

FOREIGN KEY (Offer_ID) REFERENCES Offer (Offer_ID) )
```

The constraints in the Item table prevent an item from having no cost and too high of a cost.

```
CREATE TABLE Item (

Item_ID VARCHAR (128),

Name VARCHAR (128) NOT NULL,

Price DECIMAL (10, 2) NOT NULL,

Cost DECIMAL (10, 2) NOT NULL,

Supplier VARCHAR (128) NOT NULL,

Retailer VARCHAR (128) NOT NULL,

PRIMARY KEY (Item_ID),

CHECK (Price > 0 AND Price < 1000000),

CHECK (Cost > 0 AND Cost < 1000000) )
```

```
CREATE TABLE FlyerHasPage (

Ad_ID VARCHAR (128),

Page_ID VARCHAR (128),

PRIMARY KEY (Ad_ID, Page_ID),

FOREIGN KEY (Ad_ID) REFERENCES AdFlyer (Ad_ID),

FOREIGN KEY (Page_ID) REFERENCES Page (Page_ID) )
```

```
CREATE TABLE PageContainsBoxes (

Page_ID VARCHAR (128),

Box_ID VARCHAR (128),

PRIMARY KEY (Page_ID, Box_ID),

FOREIGN KEY (Page_ID) REFERENCES Page (Page_ID),

FOREIGN KEY (Box_ID) REFERENCES Box (Box_ID) )


CREATE TABLE BoxHasOneOffer (

Offer_ID VARCHAR (128),

Box_ID VARCHAR (128),

PRIMARY KEY (Offer_ID, Box_ID),

FOREIGN KEY (Offer_ID) REFERENCES Offer (Offer_ID),

FOREIGN KEY (Box_ID) REFERENCES Box (Box_ID) )


CREATE TABLE ListOf (

Offer_ID VARCHAR (128),

Item_ID VARCHAR (128),

PRIMARY KEY (Offer_ID, Item_ID),

FOREIGN KEY (Offer_ID) REFERENCES Offer (Offer_ID),

FOREIGN KEY (Item_ID) REFERENCES Item (Item_ID) )
```

Triggers

A number of useful triggers were written in the course of the implementation, some for relatively trivial tasks, others to enable crucial functionality.

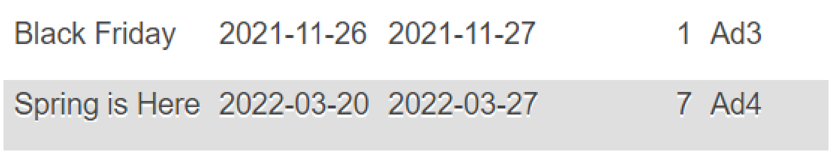
The big problem that needed to be solved was the calculation of the discount price. When a change occurs to the Offer or ListOf table a trigger updates the pricings.
Here is an example of such a trigger written for inserts in the Offer and ListOf tables:

```
After Update trigger on Offer

Call SetOfferSalePrice(New.Offer_ID)


After Insert trigger on ListOf

After Update trigger in ListOf

After Delete trigger on ListOf

Call SetOfferSalePrice(New.Offer_ID)


Stored Procedure SetOfferSalePrice(O_ID as parameter)


BEGIN

SET @RP:= (Select sum(RetailPrice) from Items where Item_ID in

(Select ListOf.Item_ID

from  ListOf where ListOf.Offer_ID = O_ID));


SET @C:= (Select sum(Cost) from Items where Item_ID in

(Select ListOf.Item_ID

from  ListOf where ListOf.Offer_ID = O_ID));


SET @M:= @RP - @C;


SET @OfferType = (Select Offer.Type from Offer where Offer.Offer_ID = O_ID);
SET @OfferAmount = (Select Offer.Amount from Offer where Offer.Offer_ID = O_ID);


if ( @OfferType = 'Dollar' )  THEN

        set @SP:= @RP - @OfferAmount;

ELSE

        set @SP:= (@RP * (1 - @OfferAmount));

END IF;



UPDATE OfferSalePrice set RetailPrice = @RP, SalePrice = @SP, Margin = @M, Cost = @C
where OfferSalePrice.Offer_ID = O_ID;

END
```

Test Data

The following set of test data was used to test insertion, updating, and deletion of data in the database, as well as the behavior of constraints:

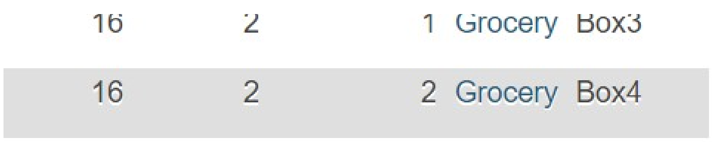
Ad Flyer



Black Friday	2021-11-26	2021-11-27	1	Ad3
Spring is Here	2022-03-20	2022-03-27	7	Ad4

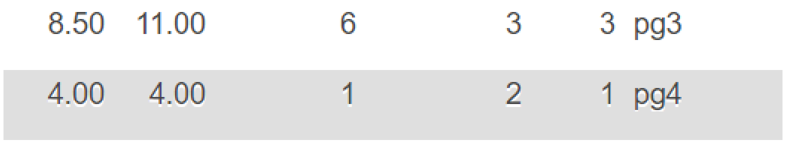
Box

Box



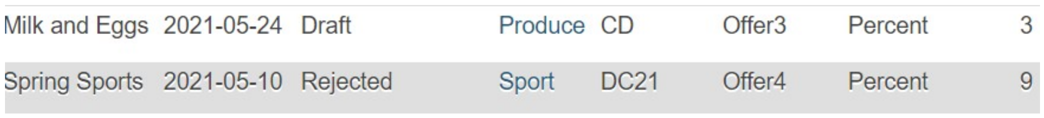
16	2	1	Grocery	Box3
16	2	2	Grocery	Box4

Page



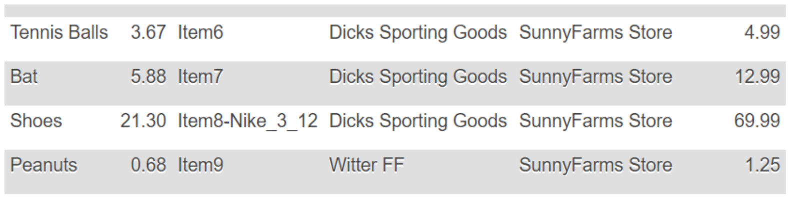
8.50	11.00	6	3	3	pg3
4.00	4.00	1	2	1	pg4

Offer



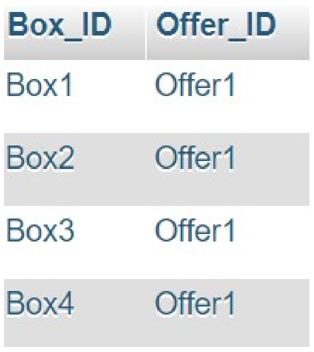
Milk and Eggs	2021-05-24	Draft	Produce	CD	Offer3	Percent	3
Spring Sports	2021-05-10	Rejected	Sport	DC21	Offer4	Percent	9

Item



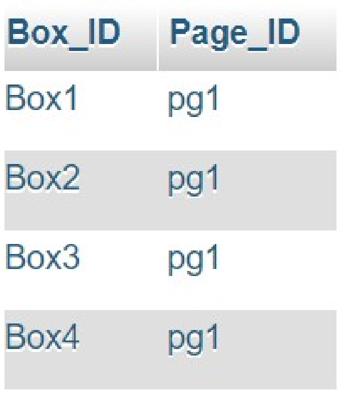
Tennis Balls	3.67	Item6	Dicks Sporting Goods	SunnyFarms Store	4.99
Bat	5.88	Item7	Dicks Sporting Goods	SunnyFarms Store	12.99
Shoes	21.30	Item8-Nike_3_12	Dicks Sporting Goods	SunnyFarms Store	69.99
Peanuts	0.68	Item9	Witter FF	SunnyFarms Store	1.25

HasOne



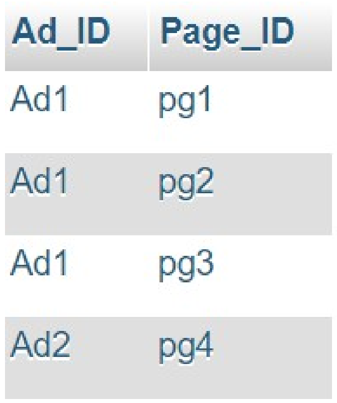
Box_ID	Offer_ID
Box1	Offer1
Box2	Offer1
Box3	Offer1
Box4	Offer1

Contains



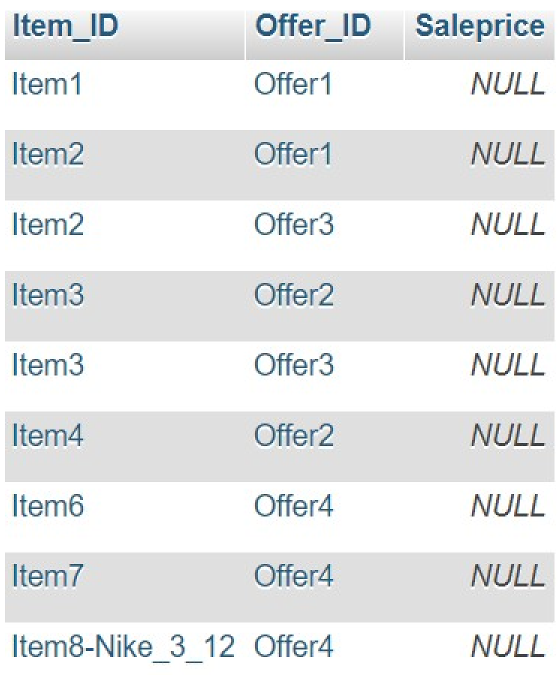
Box_ID	Page_ID
Box1	pg1
Box2	pg1
Box3	pg1
Box4	pg1

Has



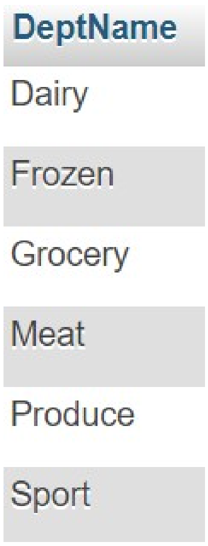
Ad_ID	Page_ID
Ad1	pg1
Ad1	pg2
Ad1	pg3
Ad2	pg4

ListOf



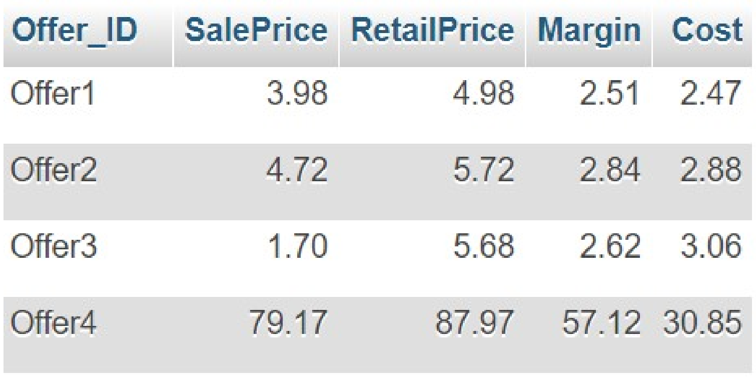
Item_ID	Offer_ID	Saleprice
Item1	Offer1	*NULL*
Item2	Offer1	*NULL*
Item2	Offer3	*NULL*
Item3	Offer2	*NULL*
Item3	Offer3	*NULL*
Item4	Offer2	*NULL*
Item6	Offer4	*NULL*
Item7	Offer4	*NULL*
Item8-Nike_3_12	Offer4	*NULL*

Department



DeptName
Dairy
Frozen
Grocery
Meat
Produce
Sport

OfferSalesPrice



Offer_ID	SalePrice	RetailPrice	Margin	Cost
Offer1	3.98	4.98	2.51	2.47
Offer2	4.72	5.72	2.84	2.88
Offer3	1.70	5.68	2.62	3.06
Offer4	79.17	87.97	57.12	30.85

Testing Procedure

The actual results shown in the testing procedure were from the first pass of testing, from which implementation problems were discovered and corrected.

Individual Steps Performed	Expected Result	Actual Results	Req. #
Validate that the default page size is 8.5x11: Enter a blank page and see result: INSERT INTO \`spr21_cis422_cdunn5\`.\`Page\` (\`Page_ID\`) VALUES ('TestPage')	No error	No error: default was set to 8.5x11	8
Update page size: UPDATE \`spr21_cis422_cdunn5\`.\`Page\` SET \`Height\` = '10.00', \`Width\` = '13.50' WHERE \`Page\`.\`Page_ID\` = 'TestPage'	No error	Successful	7
Associate a box or section on a page with an invalid department INSERT INTO \`spr21_cis422_cdunn5\`.\`Box\` (\`Dept\`, \`Box_ID\`) VALUES ('Home', 'TestBox')	Error	Error, you cannot have an invalid department as it is a forgien key	10, 11
Run Box or Section Report with a Flyer that has no pages SELECT Ad_ID FROM FlyerHasPages WHERE Page_ID IS NULL	Error	No error but returns nothing	2

Create an offer with a valid discount type Validate that the default status is set to 'Draft' INSERT INTO `spr21_cis422_cdunn5`.`Offer` (`Name`, `Offer_ID`, `Type`,`Amount`) VALUES ('TestOffer', 'OfferTest', 'Dollar', 0)	No error	The default value is indeed Draft R12, R13, R17

Associate an invalid list of items with the offer
 	
INSERT INTO
`spr21_cis422_cdunn5`.`ListOf`
(`Item_ID`, `Offer_ID`)
VALUES
('InvalidItem', 'OfferTest')
	Error	Error, item does not exist 1 8

Run Late Offers Report with percent discount type SELECT SalePrice, RetailPrice, Name FROM OfferSalePrice OS, Offer O where O.Type = 'Percent' AND O.Offer_ID = OS.Offer_ID	No Error – validate that the discount is calculated correctly	No Error, successful R 5
Run Department Discount Report SELECT Dept, SalePrice, Margin, Cost FROM OfferSalePrice OS, Offer O where O.Offer_ID = OS.Offer_ID	No error	Returns the report; Department, Sale Price, Total cost, and 2 4, R 6

Create and name boxes or sections of varying sizes on the page INSERT INTO \`spr21_cis422_cdunn5\`.\`Box\` (\`BoxSize\`, \`BoxRow\`, \`Boxcolumn\`, \`Dept\`, \`Box_ID\`) VALUES ('9', '1', '3', 'Meat', 'Box5');	no error	Successfully created box with varying size	9
Create report 1; List all offers which are approved and start within the next 3 weeks. Show the offer name, number of items in the offer, discount type, discount amount, the user who created the offer, when the offer was created, and the offer department Select Offer.Offer_ID, Adflyer.Sdate, FlyerHasPages.Ad_ID, PageContainsBoxes.Page_ID, BoxHasOneOffer.Box_ID, Offer.Name, Offer.Type, Offer.Amount,Offer.Offerdate, Offer.Dept, count(ListOf.Item_ID) from Offer, BoxHasOneOffer, PageContainsBoxes, FlyerHasPages, Adflyer, ListOf where BoxHasOneOffer.Offer_ID = Offer.Offer_ID and PageContainsBoxes.Box_ID = BoxHasOneOffer.Box_ID and FlyerHasPages.Page_ID = PageContainsBoxes.Page_ID and Adflyer.Ad_ID = FlyerHasPages.Ad_ID and ListOf.Offer_ID = Offer.Offer_ID and DATEDIFF(Adflyer.Sdate,NOW()) <= 21 and Offer.Status = 'Approved' order by Offer_ID	No error	Creation of the report successful	Report 1
Report 2; For a specific Ad Flyer, list each box with its name and the name of the offer, if it has one, that is tied to the box or section	No error	Creation of the report successful	Report 2

select FlyerHasPages.Ad_ID, PageContainsBoxes.Box_ID, BoxHasOneOffer.Offer_ID from FlyerHasPages, PageContainsBoxes, BoxHasOneOffer where FlyerHasPages.Page_ID = PageContainsBoxes.Page_ID and PageContainsBoxes.Box_ID = BoxHasOneOffer.Box_ID		

Try to update an offer with an invalid status
 	
UPDATE
`Offer`
SET
`Status` = 'Invalid'
WHERE
Offer_ID = 'Offer1'
	Error	No error but the attribute was blank, not Null. Need to add a constraint for this. 16

Appendix

Ad Flyer Page Creation

Customer Name: ICN
Summary of Need: ICN runs a small chain of grocery stores and is looking for a database solution to streamline their workflow. The exact type of workflow they are trying to improve is the creation of their Sunday Ad Flyers.

Motivation or Reason for Need: ICN wants to improve the efficiency and effectiveness of their Ad Flyers. There are errors in the Ad Flyers about what is on sale and not on sale and the current process is complex and difficult to follow.

Nature of Business: Creating a better workflow for creating Ad Flyer Pages will make the Ad Flyer more effective and more accurate. This will increase sales of the on sale items and increase revenue for the grocery stores. It will also allow the employees of ICN who must create the Ad Flyers easier and will allow them to streamline what they do so they are free to do more with their time.

Present System: The Ad Flyer is a physical and digital piece of media which is created weekly. Each Sunday a new Ad Flyer is placed in-stores and mailed to residents within a certain radius of each store. Each Ad Flyer has a name and unique alphanumeric code to identify the Ad Flyer. It also has a start and end date for when the sales in the Ad Flyer are good for. An Ad Flyer also has a type such as weekly or 3-day Ad Flyer. Typically this is one week, Sunday to Saturday, however they do run specials such as on Black Friday which do not run for a whole week. As well as longer running Ad Flyers for newly opened stores. Each Ad Flyer contains some number of pages. On an average week ICN produces Ad Flyers with 6 pages, but this can go up and down depending on the offers for that week. Each page has a height and width for printing and an order to ensure the pages show correctly in the final product. ICN has a planning team which determines how much space each department of the store gets to show of their offers. The planning team does this by creating boxes or sections on a page in a grid-like layout. Each box is assigned a name to be easily identifiable and a department. This lets the department know which box is theirs. The boxes or sections are of varying size and must be tracked for reporting purposes. ICN creates offers which represent the ‘on sale’ items customers will see in the store. For example “$1.00 off Potato Chips”. Each offer has a name and a unique numeric identifier. It has a status for tracking purposes, and a type. It also holds the discount type and discount amount. For example it could be $1 off where dollar is the type, 1 is the amount or it could be 50% off where percent is the type and 50 is the amount. Each offer also has a list of items for which the discount is applied to. Each item has a unique numeric identifier, a name, a cost, a retail, and a supplier. The ‘on sale’ price should also be tracked in the system for each item.

Statement of Customer-Perceived Requirements: ICN would like the new system to do what the employees for the old system but also create reports off the data it gathers.

1. Offers Report
    a. List all offers which are approved and start within the next 3 weeks. Show Offer Name, Number of items in the offer, discount type, discount amount, the user who created the offer, when the offer was created, the offer type
2. Box or Section Report
    a. For a specific Ad Flyer, list each box with its name and the name of the offer, if it has one, that is tied to the box or section
3. Item Report

a. For a specific Ad Flyer, list all the items inside of the Ad Flyer. Including the offer number and name, item number, description, cost, ‘on-sale’ price, and margin.

i. Margin = ‘on-sale’ - cost

4. Ad Flyper Department Page Report

a. For a specific Ad Flyer, show the percentage of area each department has for the Ad Flyer. Calculate the area of each of the boxes or sections a department has, and divide it by the total area of the pages in the Ad flyer