Database Design & Management with Microsoft Access and Microsoft SQL Server

*Note: All Data has been edited, randomized, or made up for the Security of the firm

3/21/2020

I work for the Pricing Team for T-Mobile specializing in business intelligence (automation and reporting). There is a gap in reporting and automating those reports via SQL + R because the team is receiving data from disparate sources. The team receives data from Salesforce regarding deals that receive special approvals and discounts. Everything the sales team does is recorded in Salesforce. On the other hand, all the deals won (measured by activations) and all corresponding information related to those activations are inputted into a RDMS, Teradata. Finally, the team manually inputs the credits and subsidies per deal via an excel sheet – this is the tracker management delivers on a weekly basis. Daily, a member of the team is filling out this excel document reporting the deal, credits agreed to, subsidy amount agreed to, etc. This process alone is a full-time job because we are approving 1,000 + credits/subsidies/quotes a quarter. The main problem is combining all of these different sources into one source so an accurate report can be created.

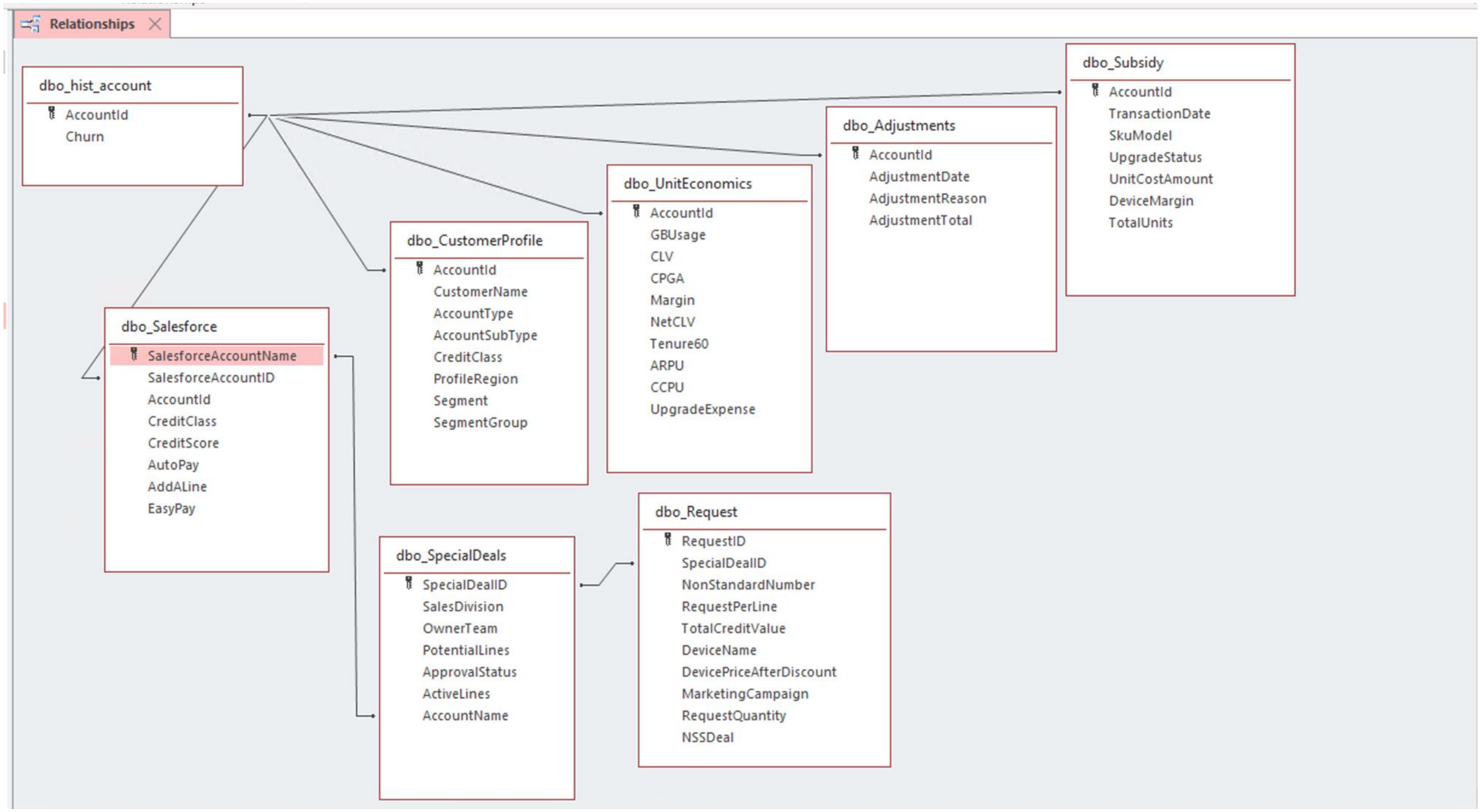
The objective is to create a monthly report that can answer these critical questions for senior management. In order to do this, my objective is to create a cohesive and unified database that can track all of this information. I want all Teradata information, Salesforce information, and Excel data inputted into one database that can be queried.

This database is able to answer the following data questions:

1) What is the average churn rate across the government sector? **50.00%**
2) Of those accounts that churn, what is the average subsidy? **$120.00**
3) Of those accounts that churn, what is the average credit received? **$1,072**
4) Of those accounts that churn, what proportion of them are NSS Deals? **40%**
5) What is the difference in the average Net CLV of those accounts that churn from the average NET CLV of those accounts that do not churn? **56**
6) What is the difference in the average Tenure of those accounts that churn from the average Tenure of those accounts that do not churn? **-1**

Lucas Daniel Zarzeczny
T-MOBILE

/*-- FULL DATABASE STRUCTURE IN MS ACCESS --*/



dbo_hist_account
- AccountId
- Churn

dbo_Salesforce
- SalesforceAccountName
- SalesforceAccountID
- AccountId
- CreditClass
- CreditScore
- AutoPay
- AddALine
- EasyPay

dbo_CustomerProfile
- AccountId
- CustomerName
- AccountType
- AccountSubType
- CreditClass
- ProfileRegion
- Segment
- SegmentGroup

dbo_UnitEconomics
- AccountId
- GBUsage
- CLV
- CPGA
- Margin
- NetCLV
- Tenure60
- ARPU
- CCPU
- UpgradeExpense

dbo_Adjustments
- AccountId
- AdjustmentDate
- AdjustmentReason
- AdjustmentTotal

dbo_Subsidy
- AccountId
- TransactionDate
- SkuModel
- UpgradeStatus
- UnitCostAmount
- DeviceMargin
- TotalUnits

dbo_SpecialDeals
- SpecialDealID
- SalesDivision
- OwnerTeam
- PotentialLines
- ApprovalStatus
- ActiveLines
- AccountName

dbo_Request
- RequestID
- SpecialDealID
- NonStandardNumber
- RequestPerLine
- TotalCreditValue
- DeviceName
- DevicePriceAfterDiscount
- MarketingCampaign
- RequestQuantity
- NSSDeal

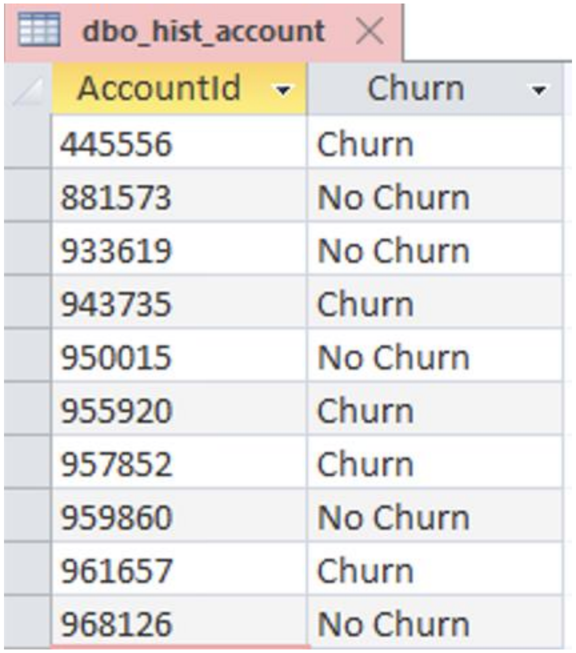
```
/*------------------------------------------------------------------------------------------------------------ DROPPING STORED PROCEDURES ------------------------------------------------------------------------------------------------------------*/
        -- Drop Stored Procedure AddUnitEconomics
        IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddUnitEconomics' AND type = 'P')
        BEGIN
            DROP PROCEDURE AddUnitEconomics
        END
        GO

        -- Drop Stored Procedure AddAdjustments
        IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddAdjustments' AND type = 'P')
        BEGIN
            DROP PROCEDURE AddAdjustments
        END
        GO

        -- Drop Stored Procedure AddSubsidy
        IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddSubsidy' AND type = 'P')
        BEGIN
            DROP PROCEDURE AddSubsidy
        END
        GO

        -- Drop Stored Procedure AddCustomerProfile
        IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddCustomerProfile' AND type = 'P')
        BEGIN
            DROP PROCEDURE AddCustomerProfile
        END
        GO

        -- Drop Stored Procedure AddSpecialDeals
        IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddSpecialDeals' AND type = 'P')
        BEGIN
            DROP PROCEDURE AddSpecialDeals
        END
        GO

        -- Drop Stored Procedure AddRequest
        IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddRequest' AND type = 'P')
        BEGIN
            DROP PROCEDURE AddRequest
        END
        GO

/*---------------------------------------------------------------------------------------------------------------------DROPPING VIEWS ------------------------------------------------------------------------------------------------------------------*/

        --Drop AvgTenureNoChurn View
        IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AvgTenureNoChurn')
```

```
BEGIN
    DROP VIEW AvgTenureNoChurn
END
GO

--Drop AvgTenureChurn View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AvgTenureChurn')
BEGIN
    DROP VIEW AvgTenureChurn
END
GO

--Drop ChurnSubsidy View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'ChurnSubsidy')
BEGIN
    DROP VIEW ChurnSubsidy
END
GO

--Drop ChurnAdjustment View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'ChurnAdjustment')
BEGIN
    DROP VIEW ChurnAdjustment
END
GO

--Drop NetCLVNoChurn View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'NetCLVNoChurn')
BEGIN
    DROP VIEW NetCLVNoChurn
END
GO


--Drop NetCLVChurn View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'NetCLVChurn')
BEGIN
    DROP VIEW NetCLVChurn
END
GO

--Drop AccountsChurned View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'ChurnSubsidy')
BEGIN
    DROP VIEW ChurnSubsidy
END
GO
```

```sql
    --Drop AccountsChurned View
    IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AccountsChurned')
    BEGIN
        DROP VIEW AccountsChurned
    END
    GO

    -- Drop AccountsNoChurn View

    IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AccountsNoChurn')
    BEGIN
        DROP VIEW AccountsNoChurn
    END
    GO

/*------------------------------------------------------------------------------------------------------------------------------------ DROPPING TABLES------------------------------------------------------------------------------------------------------------------------------------*/

    -- Dropping Table Request
    IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Request')
    BEGIN
        DROP TABLE Request
    END
    GO

    -- Dropping Table SpecialDeals
    IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'SpecialDeals')
    BEGIN
        DROP TABLE SpecialDeals
    END
    GO


    -- Dropping Table Salesforce
    IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Salesforce')
    BEGIN
        DROP TABLE Salesforce
    END
    GO


    -- Dropping Table CustomerProfile
    IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'CustomerProfile')
    BEGIN
        DROP TABLE CustomerProfile
    END
    GO
```

```
-- Dropping Table Subsidy
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Subsidy')
BEGIN
    DROP TABLE Subsidy
END
GO

-- Dropping Table Adjustments
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Adjustments')
BEGIN
    DROP TABLE Adjustments
END
GO

-- Dropping Table UnitEconomics
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'UnitEconomics')
BEGIN
    DROP TABLE UnitEconomics
END
GO

-- Dropping Table hist_account
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'hist_account')
BEGIN
    DROP TABLE hist_account
END
GO
```

```
/*------------------------------------------------------------------------------------------------------------------------ CREATING TABLES, CRUD COMMANDS, CREATING STORED PROCEDURES ----------------------------------------------------------------------------------------*/

        -- Creating the hist_account Table
        CREATE TABLE hist_account (
            -- Columns for the hist_account Table
            AccountId char(6) not null,
            Churn varchar(10) not null,
            -- Constraints on the hist_account Table
            CONSTRAINT PK_hist_account PRIMARY KEY (AccountID),
            CONSTRAINT U1_hist_account UNIQUE(AccountID)
        )
        -- End Creating the hist_account Table

        -- Insert Values into the hist_account Table

        INSERT INTO hist_account(AccountId, Churn)
            VALUES
                ('445556','Churn'),
                ('957852','Churn'),
                ('933619','No Churn'),
                ('968126','No Churn'),
                ('881573','No Churn'),
                ('959860','No Churn'),
                ('950015','No Churn'),
                ('943735','Churn'),
                ('961657','Churn'),
                ('955920','Churn')

        -- View all the values in the hist_account Table



SELECT * FROM hist_account
```



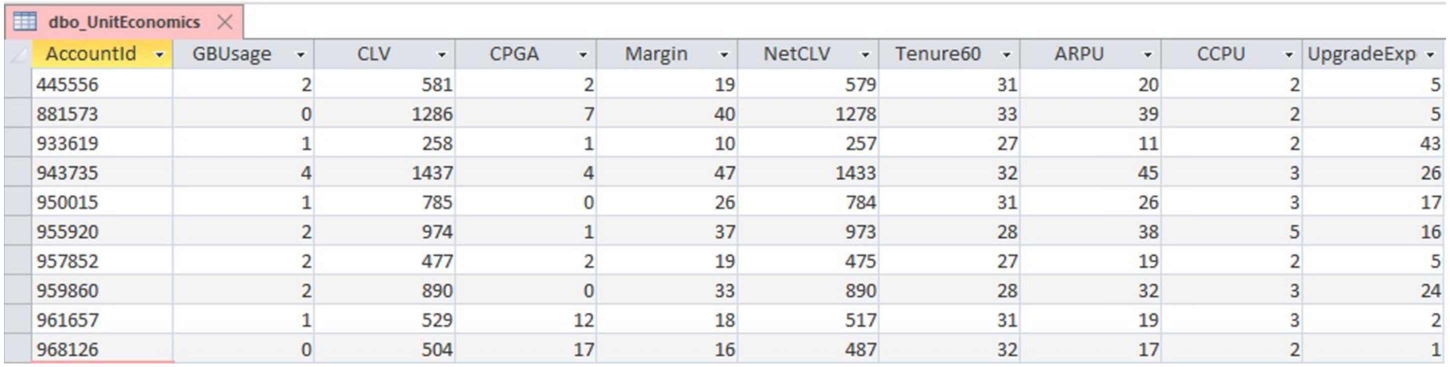
dbo_hist_account

AccountId	Churn
445556	Churn
881573	No Churn
933619	No Churn
943735	Churn
950015	No Churn
955920	Churn
957852	Churn
959860	No Churn
961657	Churn
968126	No Churn

```
-- Creating the UnitEconomics Table
CREATE TABLE UnitEconomics (
    -- Columns for the UnitEconomics Table
    AccountId char(6) not null,
    GBUsage int not null,
    CLV int not null,
    CPGA int not null,
    Margin int not null,
    NetCLV int not null,
    Tenure60 int not null,
    ARPU int not null,
    CCPU int not null,
    UpgradeExpense int not null,
    -- Constraints on the UnitEconomics Table
    CONSTRAINT PK_UnitEconomics PRIMARY KEY(AccountId),
    CONSTRAINT U1_UnitEconomics UNIQUE(AccountId),
    CONSTRAINT FK1_UnitEconomics FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the UnitEconomics Table
```

```sql
-- Insert values into UnitEconomics Table

INSERT INTO UnitEconomics(AccountId,GBUsage,CLV,CPGA,Margin,NetCLV,Tenure60,ARPU,CCPU,UpgradeExpense)
    VALUES
        ('445556',2,581,2,19,579,31,20,2,5),
        ('957852',2,477,2,19,475,27,19,2,5),
        ('933619',1,258,1,10,257,27,11,2,43),
        ('968126',0,504,17,16,487,32,17,2,1),
        ('881573',0,1286,7,40,1278,33,39,2,5),
        ('959860',2,890,0,33,890,28,32,3,24),
        ('950015',1,785,0,26,784,31,26,3,17),
        ('943735',4,1437,4,47,1433,32,45,3,26),
        ('961657',1,529,12,18,517,31,19,3,2),
        ('955920',2,974,1,37,973,28,38,5,16)

/*Creating a procedure for UnitEconomics. We can only add data into the table if the AccountID specified by the user
is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
If it is, the user cannot add data to the UnitEconomics Table.
*/
GO
CREATE PROCEDURE AddUnitEconomics(
    @AccountId char(6),
    @GBUsage int,
    @CLV int,
    @CPGA int,
    @Margin int,
    @NetCLV int,
    @Tenure60 int,
    @ARPU int,
    @CCPU int,
    @UpgradeExpense int)
AS
BEGIN

    -- We need the AccountID from the hist_account table
    -- First, declare a variable to hold the ID
    DECLARE @GetAccountID int

    -- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
    SELECT @GetAccountID = AccountId FROM hist_account
    WHERE hist_account.AccountId IS NOT NULL
    AND hist_account.AccountId = @AccountId

    --Now we can add the row using an insert statement
    INSERT INTO UnitEconomics(AccountId,GBUsage,CLV,CPGA,Margin,NetCLV,Tenure60,ARPU,CCPU,UpgradeExpense)
    VALUES (@GetAccountID,@GBUsage, @CLV, @CPGA, @Margin, @NetCLV, @Tenure60, @ARPU, @CCPU,@UpgradeExpense)
```

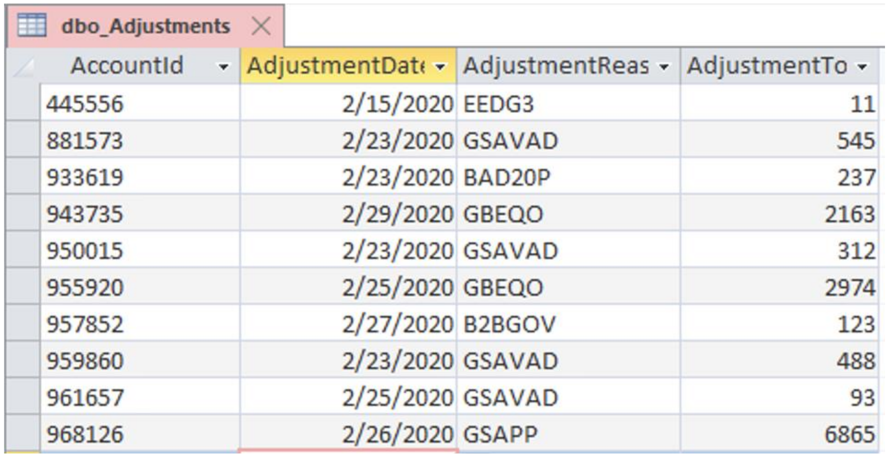
```
        --Now return the @@identity so the calling code knows where
        -- the data ended up
        RETURN SCOPE_IDENTITY()
    END
    GO

    --End of creating the stored procedure

    -- View all the values in the UnitEconomics Table

SELECT * FROM UnitEconomics
```



AccountId	GBUsage	CLV	CPGA	Margin	NetCLV	Tenure60	ARPU	CCPU	UpgradeExp
445556	2	581	2	19	579	31	20	2	5
881573	0	1286	7	40	1278	33	39	2	5
933619	1	258	1	10	257	27	11	2	43
943735	4	1437	4	47	1433	32	45	3	26
950015	1	785	0	26	784	31	26	3	17
955920	2	974	1	37	973	28	38	5	16
957852	2	477	2	19	475	27	19	2	5
959860	2	890	0	33	890	28	32	3	24
961657	1	529	12	18	517	31	19	3	2
968126	0	504	17	16	487	32	17	2	1

```
    -- Creating the Adjustments Table
    CREATE TABLE Adjustments (
      -- Columns for the Adjustments Table
      AccountId char(6) not null,
      AdjustmentDate datetime not null default GetDate(),
      AdjustmentReason varchar(10) not null,
      AdjustmentTotal int not null,
      -- Constraints on the Adjustments Table
      CONSTRAINT PK_Adjustments PRIMARY KEY(AccountId),
      CONSTRAINT U1_Adjustments UNIQUE(AccountId),
      CONSTRAINT FK1_Adjustments FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
    )
    -- End Creating the Adjustments Table

    -- Insert values into Adjustments Table

    INSERT INTO Adjustments(AccountId,AdjustmentDate,AdjustmentReason,AdjustmentTotal)
      VALUES
        ('445556','2020-02-15','EEDG3', 11),
        ('957852','2020-02-27','B2BGOV',123),
        ('933619','2020-02-23','BAD20P',237),
```

```sql
        ('968126','2020-02-26','GSAPP',6865),
        ('881573','2020-02-23','GSAVAD',545),
        ('959860','2020-02-23','GSAVAD',488),
        ('950015','2020-02-23','GSAVAD',312),
        ('943735','2020-02-29','GBEQO',2163),
        ('961657','2020-02-25','GSAVAD',93),
        ('955920','2020-02-25','GBEQO',2974)

/*Creating a procedure for Adjustments. We can only add data into the table if the AccountID specified by the user
is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
If it is, the user cannot add data to the Adjustments Table.
*/
GO
CREATE PROCEDURE AddAdjustments(
    @AccountId char(6),
    @AdjustmentDate datetime,
    @AdjustmentReason varchar(10),
    @AdjustmentTotal int)
AS
BEGIN

    -- We need the AccountID from the hist_account table
    -- First, declare a variable to hold the ID
    DECLARE @GetAccountID int

    -- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
    SELECT @GetAccountID = AccountId FROM hist_account
    WHERE hist_account.AccountId IS NOT NULL
    AND hist_account.AccountId = @AccountId

    --Now we can add the row using an insert statement
    INSERT INTO Adjustments(AccountId,AdjustmentDate,AdjustmentReason,AdjustmentTotal)
    VALUES (@GetAccountID,@AdjustmentDate,@AdjustmentReason,@AdjustmentTotal)

    --Now return the @@identity so the calling code knows where
    -- the data ended up
    RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

--View all the values in the Adjustments Table

SELECT * FROM Adjustments
```

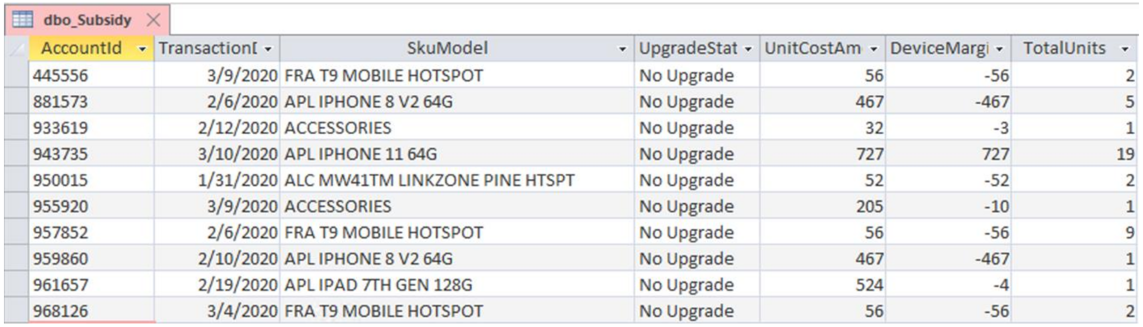


AccountId	AdjustmentDate	AdjustmentReas	AdjustmentTo
445556	2/15/2020	EEDG3	11
881573	2/23/2020	GSAVAD	545
933619	2/23/2020	BAD20P	237
943735	2/29/2020	GBEQO	2163
950015	2/23/2020	GSAVAD	312
955920	2/25/2020	GBEQO	2974
957852	2/27/2020	B2BGOV	123
959860	2/23/2020	GSAVAD	488
961657	2/25/2020	GSAVAD	93
968126	2/26/2020	GSAPP	6865

```
-- Creating the Subsidy Table
CREATE TABLE Subsidy (
  -- Columns for the Subsidy Table
  AccountId char(6) not null,
  TransactionDate datetime not null default GetDate(),
  SkuModel varchar(100) not null,
  UpgradeStatus varchar(50) not null,
  UnitCostAmount int not null,
  DeviceMargin int not null,
  TotalUnits int not null,
  -- Constraints on the Subsidy Table
  CONSTRAINT PK_Subsidy PRIMARY KEY(AccountId),
  CONSTRAINT U1_Subsidy UNIQUE(AccountId),
  CONSTRAINT FK1_Subsidy FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the Subsidy Table

-- Insert values into Subsidy Table

INSERT INTO Subsidy(AccountId,TransactionDate,SkuModel,UpgradeStatus,UnitCostAmount,DeviceMargin,TotalUnits)
  VALUES
    ('445556','2020-03-09','FRA T9 MOBILE HOTSPOT','No Upgrade',56,-56,2),
    ('957852','2020-02-06','FRA T9 MOBILE HOTSPOT','No Upgrade',56,-56,9),
    ('933619','2020-02-12','ACCESSORIES','No Upgrade',32,-3,1),
    ('968126','2020-03-04','FRA T9 MOBILE HOTSPOT','No Upgrade',56,-56,2),
    ('881573','2020-02-06','APL IPHONE 8 V2 64G','No Upgrade',467,-467,5),
    ('959860','2020-02-10','APL IPHONE 8 V2 64G','No Upgrade',467,-467,1),
    ('950015','2020-01-31','ALC MW41TM LINKZONE PINE HTSPT','No Upgrade',52,-52,2),
    ('943735','2020-03-10','APL IPHONE 11 64G','No Upgrade',727,727,19),
```

```sql
        ('961657','2020-02-19','APL IPAD 7TH GEN 128G','No Upgrade',524,-4,1),
        ('955920','2020-03-09','ACCESSORIES','No Upgrade',205,-10,1)

/*Creating a procedure for Subsidy Table. We can only add data into the table if the AccountID specified by the user
is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
If it is, the user cannot add data to the Subsidy Table.
*/
GO
CREATE PROCEDURE AddSubsidy(
    @AccountId char(6),
    @TransactionDate datetime,
    @SkuModel varchar(100),
    @UpgradeStatus varchar(50),
    @UnitCostAmount int,
    @DeviceMargin int,
    @TotalUnits int)
AS
BEGIN

    -- We need the AccountID from the hist_account table
    -- First, declare a variable to hold the ID
    DECLARE @GetAccountID int

    -- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
    SELECT @GetAccountID = AccountId FROM hist_account
    WHERE hist_account.AccountId IS NOT NULL
    AND hist_account.AccountId = @AccountId

    --Now we can add the row using an insert statement
    INSERT INTO Subsidy(AccountId,TransactionDate,SkuModel,UpgradeStatus,UnitCostAmount,DeviceMargin,TotalUnits)
    VALUES (@GetAccountID,@TransactionDate,@SkuModel,@UpgradeStatus,@UnitCostAmount,@DeviceMargin,@TotalUnits)

    --Now return the @@identity so the calling code knows where
    -- the data ended up
    RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

-- View all the values in the Subsidy Table
```

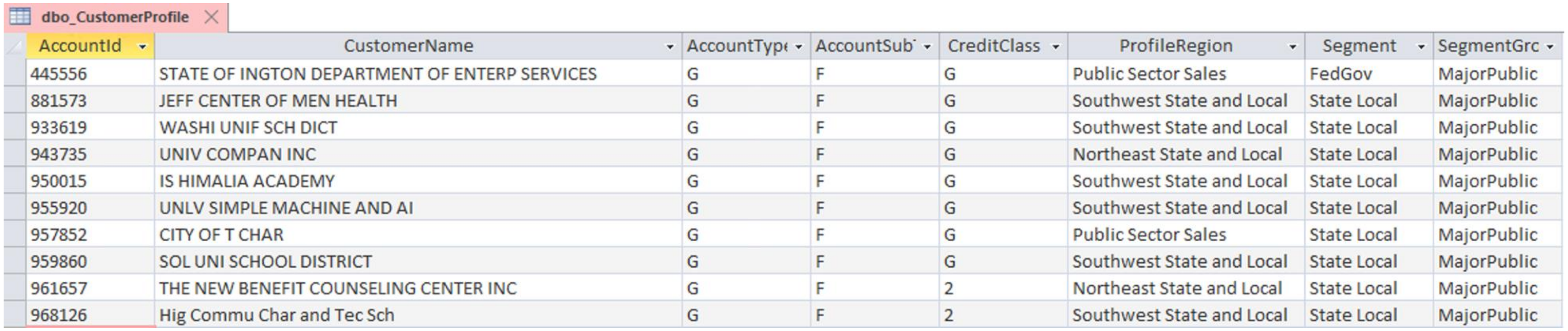
```sql
SELECT * FROM Subsidy
```



dbo_Subsidy

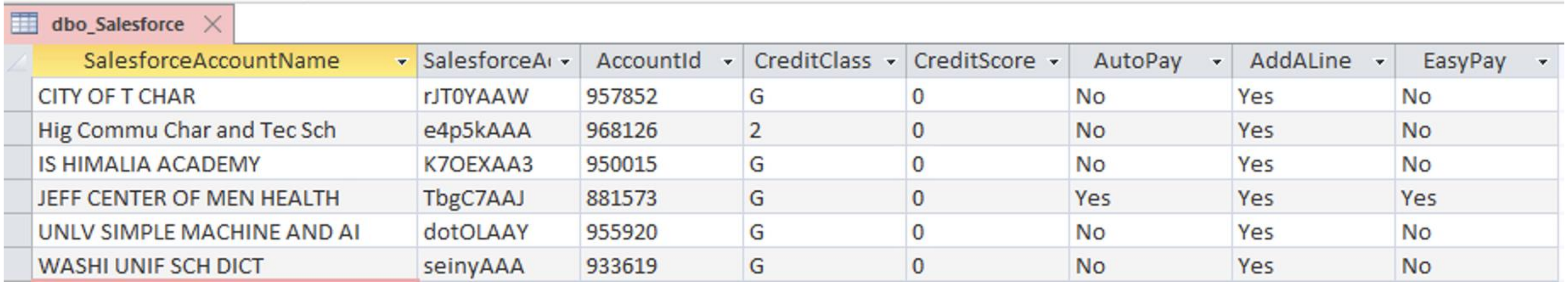
AccountId	TransactionI	SkuModel	UpgradeStat	UnitCostAm	DeviceMargi	TotalUnits
445556	3/9/2020	FRA T9 MOBILE HOTSPOT	No Upgrade	56	-56	2
881573	2/6/2020	APL IPHONE 8 V2 64G	No Upgrade	467	-467	5
933619	2/12/2020	ACCESSORIES	No Upgrade	32	-3	1
943735	3/10/2020	APL IPHONE 11 64G	No Upgrade	727	727	19
950015	1/31/2020	ALC MW41TM LINKZONE PINE HTSPT	No Upgrade	52	-52	2
955920	3/9/2020	ACCESSORIES	No Upgrade	205	-10	1
957852	2/6/2020	FRA T9 MOBILE HOTSPOT	No Upgrade	56	-56	9
959860	2/10/2020	APL IPHONE 8 V2 64G	No Upgrade	467	-467	1
961657	2/19/2020	APL IPAD 7TH GEN 128G	No Upgrade	524	-4	1
968126	3/4/2020	FRA T9 MOBILE HOTSPOT	No Upgrade	56	-56	2

```sql
-- Creating the CustomerProfile Table
CREATE TABLE CustomerProfile (
   -- Columns for the CustomerProfile Table
   AccountId char(6) not null,
   CustomerName varchar(200) not null,
   AccountType char(1) not null,
   AccountSubType char(1) not null,
   CreditClass char(1) not null,
   ProfileRegion varchar(100) not null,
   Segment varchar(50) not null,
   SegmentGroup char(11) not null,
   -- Constraints on the CustomerProfile Table
   CONSTRAINT PK_CustomerProfile PRIMARY KEY(AccountId),
   CONSTRAINT U1_CustomerProfile UNIQUE(AccountId),
   CONSTRAINT U2_CustomerProfile UNIQUE(CustomerName),
   CONSTRAINT FK1_CustomerProfile FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the CustomerProfile Table

-- Insert values into CustomerProfile Table

INSERT INTO CustomerProfile(AccountId,CustomerName,AccountType,AccountSubType,CreditClass,ProfileRegion,Segment,SegmentGroup)
   VALUES
      ('445556','STATE OF INGTON DEPARTMENT OF ENTERP SERVICES','G','F','G','Public Sector Sales','FedGov','MajorPublic'),
      ('957852','CITY OF T CHAR','G','F','G','Public Sector Sales','State Local','MajorPublic'),
      ('933619','WASHI UNIF SCH DICT','G','F','G','Southwest State and Local','State Local','MajorPublic'),
      ('968126','Hig Commu Char and Tec Sch','G','F','2','Southwest State and Local','State Local','MajorPublic'),
      ('881573','JEFF CENTER OF MEN HEALTH','G','F','G','Southwest State and Local','State Local','MajorPublic'),
        ('959860','SOL UNI SCHOOL DISTRICT','G','F','G','Southwest State and Local','State Local','MajorPublic'),
        ('950015','IS HIMALIA ACADEMY','G','F','G','Southwest State and Local','State Local','MajorPublic'),
        ('943735','UNIV COMPAN INC','G','F','G','Northeast State and Local','State Local','MajorPublic'),
        ('961657','THE NEW BENEFIT COUNSELING CENTER INC','G','F','2','Northeast State and Local','State Local','MajorPublic'),
        ('955920','UNLV SIMPLE MACHINE AND AI','G','F','G','Southwest State and Local','State Local','MajorPublic')

    /*Creating a procedure for CustomerProfile Table. We can only add data into the table if the AccountID specified by the user
    is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
    If it is, the user cannot add data to the CustomerProfile Table.
    */
    GO
    CREATE PROCEDURE AddCustomerProfile(
      @AccountId char(6),
      @CustomerName varchar(200),
      @AccountType char(1),
      @AccountSubType char(1),
      @CreditClass char(1),
      @ProfileRegion varchar(100),
      @Segment varchar(50),
      @SegmentGroup char(11))
    AS
    BEGIN

      -- We need the AccountID from the hist_account table
      -- First, declare a variable to hold the ID
      DECLARE @GetAccountID int

      -- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
      SELECT @GetAccountID = AccountId FROM hist_account
      WHERE hist_account.AccountId IS NOT NULL
      AND hist_account.AccountId = @AccountId

      --Now we can add the row using an insert statement
      INSERT INTO CustomerProfile(AccountId,CustomerName,AccountType,AccountSubType,CreditClass,ProfileRegion,Segment,SegmentGroup)
      VALUES (@GetAccountID,@CustomerName,@AccountType,@AccountSubType,@CreditClass,@ProfileRegion,@Segment,@SegmentGroup)

      --Now return the @@identity so the calling code knows where
      -- the data ended up
      RETURN SCOPE_IDENTITY()
    END
    GO

    --End of creating the stored procedure

    -- View all the values in the CustomerProfile Table

SELECT * FROM CustomerProfile
```



AccountId	CustomerName	AccountTyp	AccountSub	CreditClass	ProfileRegion	Segment	SegmentGrc
445556	STATE OF INGTON DEPARTMENT OF ENTERP SERVICES	G	F	G	Public Sector Sales	FedGov	MajorPublic
881573	JEFF CENTER OF MEN HEALTH	G	F	G	Southwest State and Local	State Local	MajorPublic
933619	WASHI UNIF SCH DICT	G	F	G	Southwest State and Local	State Local	MajorPublic
943735	UNIV COMPAN INC	G	F	G	Northeast State and Local	State Local	MajorPublic
950015	IS HIMALIA ACADEMY	G	F	G	Southwest State and Local	State Local	MajorPublic
955920	UNLV SIMPLE MACHINE AND AI	G	F	G	Southwest State and Local	State Local	MajorPublic
957852	CITY OF T CHAR	G	F	G	Public Sector Sales	State Local	MajorPublic
959860	SOL UNI SCHOOL DISTRICT	G	F	G	Southwest State and Local	State Local	MajorPublic
961657	THE NEW BENEFIT COUNSELING CENTER INC	G	F	2	Northeast State and Local	State Local	MajorPublic
968126	Hig Commu Char and Tec Sch	G	F	2	Southwest State and Local	State Local	MajorPublic

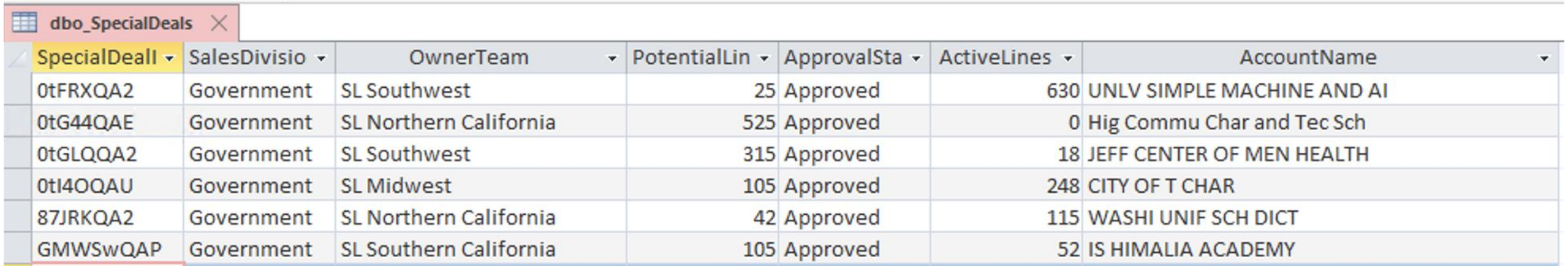
```
-- Creating the Salesforce Table
CREATE TABLE Salesforce (
   -- Columns for the Salesforce Table
   SalesforceAccountName varchar(200),
   SalesforceAccountID char(8) not null,
   AccountId char(6) not null,
   CreditClass varchar(3) not null,
   CreditScore varchar(3) not null,
   AutoPay varchar(5) not null,
   AddALine varchar(5) not null,
   EasyPay varchar(5) not null,
   -- Constraints on the Salesforce Table
   CONSTRAINT PK_Salesforce PRIMARY KEY(SalesforceAccountName),
   CONSTRAINT U1_Salesforce UNIQUE(SalesforceAccountName),
   CONSTRAINT U2_Salesforce UNIQUE(SalesforceAccountID),
   CONSTRAINT U3_Salesforce UNIQUE(AccountId),
   CONSTRAINT FK1_Salesforce FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the Salesforce Table

-- Insert values into Salesforce Table

INSERT INTO Salesforce(SalesforceAccountName,SalesforceAccountID,AccountId,CreditClass,CreditScore,AutoPay,AddALine,EasyPay)
   VALUES
      ('CITY OF T CHAR','rJT0YAAW','957852','G','0','No','Yes','No'),
      ('WASHI UNIF SCH DICT','seinyAAA','933619','G','0','No','Yes','No'),
      ('Hig Commu Char and Tec Sch','e4p5kAAA','968126','2','0','No','Yes','No'),
      ('JEFF CENTER OF MEN HEALTH','TbgC7AAJ','881573','G ','0','Yes','Yes','Yes'),
      ('IS HIMALIA ACADEMY','K7OEXAA3','950015','G','0','No','Yes','No'),
        ('UNLV SIMPLE MACHINE AND AI','dotOLAAY','955920','G','0','No','Yes','No')

    -- View all the values in the Salesforce Table


SELECT * FROM Salesforce
```



SalesforceAccountName	SalesforceA	AccountId	CreditClass	CreditScore	AutoPay	AddALine	EasyPay
CITY OF T CHAR	rJT0YAAW	957852	G	0	No	Yes	No
Hig Commu Char and Tec Sch	e4p5kAAA	968126	2	0	No	Yes	No
IS HIMALIA ACADEMY	K7OEXAA3	950015	G	0	No	Yes	No
JEFF CENTER OF MEN HEALTH	TbgC7AAJ	881573	G	0	Yes	Yes	Yes
UNLV SIMPLE MACHINE AND AI	dotOLAAY	955920	G	0	No	Yes	No
WASHI UNIF SCH DICT	seinyAAA	933619	G	0	No	Yes	No

```
    -- Creating the SpecialDeals Table
    CREATE TABLE SpecialDeals (
       -- Columns for the SpecialDeals Table
       SpecialDealID char(8) not null,
       SalesDivision varchar(30) not null,
       OwnerTeam varchar(100) not null,
       PotentialLines int  not null,
       ApprovalStatus varchar(50) not null,
       ActiveLines int not null,
       AccountName varchar(200) not null,
       -- Constraints on the SpecialDeals Table
       CONSTRAINT PK_SpecialDeals PRIMARY KEY(SpecialDealID),
       CONSTRAINT U1_SpecialDeals UNIQUE(SpecialDealID),
       CONSTRAINT U2_SpecialDeals UNIQUE(AccountName),
       CONSTRAINT FK1_SpecialDeals FOREIGN KEY (AccountName) REFERENCES Salesforce(SalesforceAccountName)
    )
    -- End Creating the SpecialDeals Table

    -- Insert values into SpecialDeals Table

    INSERT INTO SpecialDeals(SpecialDealID,SalesDivision,OwnerTeam,PotentialLines,ApprovalStatus,ActiveLines,AccountName)
       VALUES
          ('0tI4OQAU','Government','SL Midwest',105,'Approved',248,'CITY OF T CHAR'),
          ('87JRKQA2','Government','SL Northern California',42,'Approved',115,'WASHI UNIF SCH DICT'),
          ('0tG44QAE','Government','SL Northern California',525,'Approved',0,'Hig Commu Char and Tec Sch'),
```

```sql
        ('0tGLQQA2','Government','SL Southwest',315,'Approved',18,'JEFF CENTER OF MEN HEALTH'),
        ('GMWSwQAP','Government','SL Southern California',105,'Approved',52,'IS HIMALIA ACADEMY'),
        ('0tFRXQA2','Government','SL Southwest',25,'Approved',630,'UNLV SIMPLE MACHINE AND AI')

/*Creating a procedure for SpecialDeals Table. We can only add data into the table if the Account Name specified by the user
is equal to the same Account Name in the Salesforce table. Also, the Account Name in the Salesforce table cannot be null.
If it is, the user cannot add data to the SpecialDeals Table.
*/
GO
CREATE PROCEDURE AddSpecialDeals(
    @SpecialDealID char(8),
    @SalesDivision varchar(30),
    @OwnerTeam varchar(100),
    @PotentialLines int,
    @ApprovalStatus varchar(50),
    @ActiveLines int,
    @AccountName varchar(200))
AS
BEGIN

    -- We need the Account Name from the Salesforce Table
    -- First, declare a variable to hold the name
    DECLARE @GetAccountName varchar(200)

    -- Get the Account Name from the Salesforce Table from the Account Name provided by the user and store it in @GetAccountName
    SELECT @GetAccountName = SalesforceAccountName FROM Salesforce
    WHERE Salesforce.SalesforceAccountName IS NOT NULL
    AND Salesforce.SalesforceAccountName = @AccountName

    --Now we can add the row using an insert statement
    INSERT INTO SpecialDeals(SpecialDealID,SalesDivision,OwnerTeam,PotentialLines,ApprovalStatus,ActiveLines,AccountName)
    VALUES (@SpecialDealID,@SalesDivision,@OwnerTeam,@PotentialLines,@ApprovalStatus,@ActiveLines,@GetAccountName)

    --Now return the @@identity so the calling code knows where
    -- the data ended up
    RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

-- View all the values in the SpecialDeals Table

SELECT * FROM SpecialDeals
```

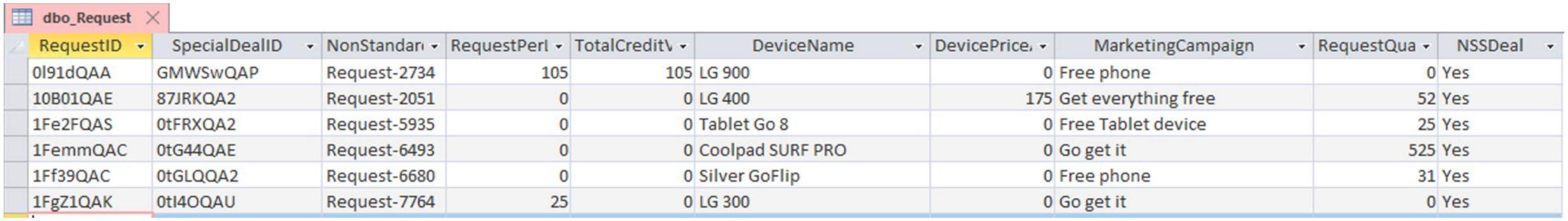


SpecialDealI	SalesDivisio	OwnerTeam	PotentialLin	ApprovalSta	ActiveLines	AccountName
0tFRXQA2	Government	SL Southwest	25	Approved	630	UNLV SIMPLE MACHINE AND AI
0tG44QAE	Government	SL Northern California	525	Approved	0	Hig Commu Char and Tec Sch
0tGLQQA2	Government	SL Southwest	315	Approved	18	JEFF CENTER OF MEN HEALTH
0tI4OQAU	Government	SL Midwest	105	Approved	248	CITY OF T CHAR
87JRKQA2	Government	SL Northern California	42	Approved	115	WASHI UNIF SCH DICT
GMWSwQAP	Government	SL Southern California	105	Approved	52	IS HIMALIA ACADEMY

```
-- Creating the Request Table
CREATE TABLE Request (
  -- Columns for the Request Table
  RequestID varchar(50) not null,
  SpecialDealID char(8) not null,
  NonStandardNumber varchar(50) not null,
  RequestPerLine int  not null,
  TotalCreditValue int not null,
  DeviceName varchar(100) not null,
  DevicePriceAfterDiscount int not null,
  MarketingCampaign varchar(200) not null,
  RequestQuantity int not null,
  NSSDeal varchar(5) not null,
  -- Constraints on the Request Table
  CONSTRAINT PK_Request PRIMARY KEY(RequestID),
  CONSTRAINT U1_Request UNIQUE(RequestID),
  CONSTRAINT U2_Request UNIQUE(SpecialDealID),
  CONSTRAINT U3_Request UNIQUE(NonStandardNumber),
  CONSTRAINT FK1_Request FOREIGN KEY (SpecialDealID) REFERENCES SpecialDeals(SpecialDealID)
)
-- End Creating the Request Table

-- Insert values into Request Table

INSERT INTO
Request(RequestID,SpecialDealID,NonStandardNumber,RequestPerLine,TotalCreditValue,DeviceName,DevicePriceAfterDiscount,MarketingCampaign,RequestQuantity,NSSDeal)
  VALUES
    ('1FgZ1QAK','0tI4OQAU','Request-7764',25,0,'LG 300',0,'Go get it',0,'Yes'),
    ('10B01QAE','87JRKQA2','Request-2051',0,0,'LG 400',175,'Get everything free',52,'Yes'),
    ('1FemmQAC','0tG44QAE','Request-6493',0,0,'Coolpad SURF PRO',0,'Go get it',525,'Yes'),
    ('1Ff39QAC','0tGLQQA2','Request-6680',0,0,'Silver GoFlip',0,'Free phone',31,'Yes'),
    ('0I91dQAA','GMWSwQAP','Request-2734',105,105,'LG 900',0,'Free phone',0,'Yes'),
    ('1Fe2FQAS','0tFRXQA2','Request-5935',0,0,'Tablet Go 8',0,'Free Tablet device',25,'Yes')
```

```
/*Creating a procedure for the Request Table. We can only add data into the Request Table if the Special Deal ID specified by the user
is equal to the same Special ID in the Special Deals table. Also, the Special Deals ID in the Special Deals table cannot be null.
If it is, the user cannot add data to the Request Table.
*/
GO
CREATE PROCEDURE AddRequest(
    @RequestID varchar(50),
    @SpecialDealID char(8),
    @NonStandardNumber varchar(50),
    @RequestPerLine int,
    @TotalCreditValue int,
    @DeviceName varchar(100),
    @DevicePriceAfterDiscount int,
    @MarketingCampaign varchar(200),
    @RequestQuantity int,
    @NSSDeal varchar(5))
AS
BEGIN

    -- We need the SpecialDealID from the Special Deals Table
    -- First, declare a variable to hold the ID
    DECLARE @GetAccountID int

    -- Get the SpecialDealID from the Special Deals  Table from the ID provided by the user and store it in @GetAccountID
    SELECT @GetAccountID = SpecialDealID FROM SpecialDeals
    WHERE SpecialDeals.SpecialDealID IS NOT NULL
    AND SpecialDeals.SpecialDealID = @SpecialDealID

    --Now we can add the row using an insert statement
    INSERT INTO
Request(RequestID,SpecialDealID,NonStandardNumber,RequestPerLine,TotalCreditValue,DeviceName,DevicePriceAfterDiscount,MarketingCampaign,RequestQuantity,NSSDeal)
    VALUES
(@RequestID,@GetAccountID,@NonStandardNumber,@RequestPerLine,@TotalCreditValue,@DeviceName,@DevicePriceAfterDiscount,@MarketingCampaign,@RequestQuantity, @NSSDeal)

    --Now return the @@identity so the calling code knows where
    -- the data ended up
    RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

-- View all the values in the Request Table
```

```
SELECT * FROM Request
```

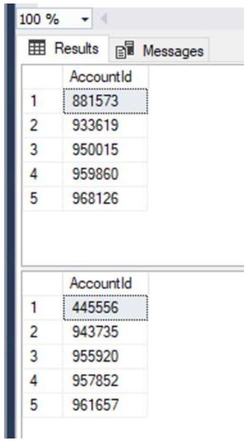


dbo_Request

RequestID	SpecialDealID	NonStandar	RequestPerl	TotalCreditV	DeviceName	DevicePrice	MarketingCampaign	RequestQua	NSSDeal
0I91dQAA	GMWSwQAP	Request-2734	105	105	LG 900	0	Free phone	0	Yes
10B01QAE	87JRKQA2	Request-2051	0	0	LG 400	175	Get everything free	52	Yes
1Fe2FQAS	0tFRXQA2	Request-5935	0	0	Tablet Go 8	0	Free Tablet device	25	Yes
1FemmQAC	0tG44QAE	Request-6493	0	0	Coolpad SURF PRO	0	Go get it	525	Yes
1Ff39QAC	0tGLQQA2	Request-6680	0	0	Silver GoFlip	0	Free phone	31	Yes
1FgZ1QAK	0tI4OQAU	Request-7764	25	0	LG 300	0	Go get it	0	Yes

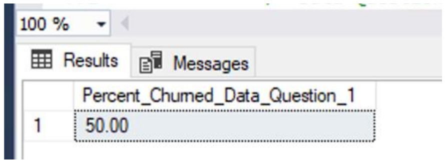
```
/*---------------------------------------------------------------------------------------------------------- DATA QUESTIONS----------------------------------------------------------------------------------------------------------*/

    /* Data Question 1

    1)  What percent of accounts churn?

    */

GO

-- Create a view of all accounts that Churned

CREATE VIEW AccountsChurned AS (
SELECT
hist_account.AccountId
FROM hist_account
WHERE hist_account.Churn = 'Churn'
)


GO

-- Create a view of all accounts that did not churn

CREATE VIEW AccountsNoChurn AS (
SELECT
hist_account.AccountId
FROM hist_account
WHERE hist_account.Churn = 'No Churn'
)
```

```sql
GO

-- Query to get the percent of the accounts that churned

SELECT
CAST(CAST(a.Total_Churned AS decimal (12,4)) /  CAST(a.Total_Accounts AS decimal (12,4)) AS decimal(12,2))*100 as Percent_Churned_Data_Question_1
FROM
(
SELECT
(SELECT COUNT(*) FROM AccountsChurned) as Total_Churned,
COUNT(DISTINCT hist_account.AccountId) as Total_Accounts
FROM hist_account
) a

GO


/* Data Question 1 Answer

  1)  50.00%

  */
```



	AccountId
1	881573
2	933619
3	950015
4	959860
5	968126

	AccountId
1	445556
2	943735
3	955920
4	957852
5	961657



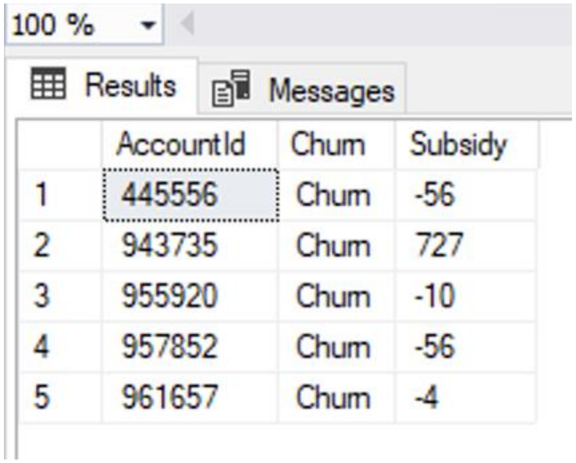
	Percent_Churned_Data_Question_1
1	50.00

```sql
 /* Data Question 2

 2)  Of those accounts that churn, what is the average subsidy?

 */

--Query to get the average subsidy for all the accounts that churned

SELECT
CAST(AVG(Subsidy.DeviceMargin) as decimal (12,2)) as Average_Subsidy_Data_Question_2
FROM Subsidy
WHERE Subsidy.AccountId IN (SELECT AccountId FROM AccountsChurned)
-- Check

GO

--Creating a view of those accounts that churned and took a subsidy
CREATE VIEW ChurnSubsidy AS (
SELECT
b.AccountId,
b.Churn,
DeviceMargin as Subsidy
FROM Subsidy a
INNER JOIN hist_account b on a.AccountId = b.AccountId

)

GO

--Query to confirm the average susbidy amount for those accounts that churned

SELECT
CAST(AVG(a.Subsidy) as decimal (12,2)) as Average_Subsidy_Check
FROM
(
SELECT
AccountId,
Churn,
Subsidy
FROM ChurnSubsidy
WHERE Churn = 'Churn'
-- 5 accounts that churn with a subsidy
) a

-- Check complete. $120.00
```

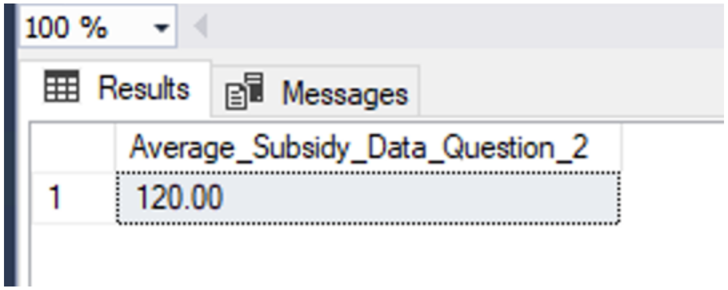
/* Data Question 2 Answer

2) $120.00

*/



	AccountId	Churn	Subsidy
1	445556	Churn	-56
2	943735	Churn	727
3	955920	Churn	-10
4	957852	Churn	-56
5	961657	Churn	-4



	Average_Subsidy_Data_Question_2
1	120.00

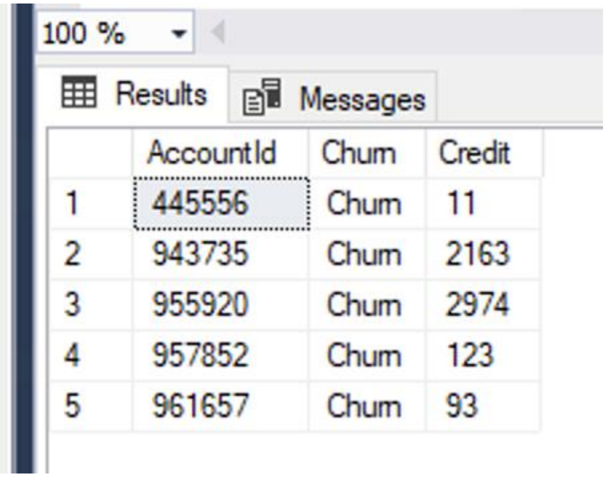
/* Data Question 3

3) Of those accounts that churn, what is the average credit received?

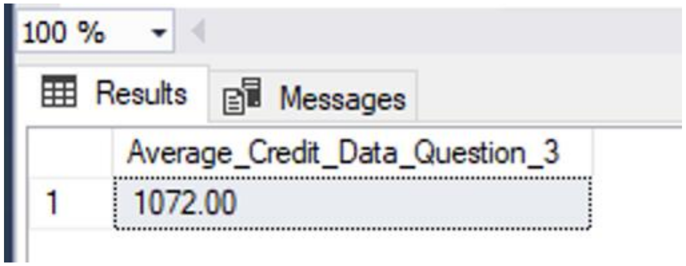
*/

-- Query to get the average credit recieved across the accounts that churned

```sql
SELECT
CAST(AVG(Adjustments.AdjustmentTotal) as decimal (12,2)) as Average_Credit_Data_Question_3
FROM Adjustments
WHERE Adjustments.AccountId IN (SELECT AccountId FROM AccountsChurned)
-- Check

GO

-- Creating a view that selects all the accounts that churned and received a credit

CREATE VIEW ChurnAdjustment AS (
SELECT
b.AccountId,
b.Churn,
a.AdjustmentTotal as Credit
FROM Adjustments a
INNER JOIN hist_account b on a.AccountId = b.AccountId
)

GO

-- Performing a query to validate the intial query in terms of the average credit

SELECT
CAST(AVG(a.Credit) as decimal (12,2)) as Average_Credit_Check
FROM
(
SELECT
AccountId,
Churn,
Credit
FROM ChurnAdjustment
WHERE Churn = 'Churn'
-- 5 accounts that churn with a credit
) a

-- Check complete. $1,072

SELECT * FROM ChurnAdjustment

/* Data Question 3 Answer

  3)  $1,072

  */
```



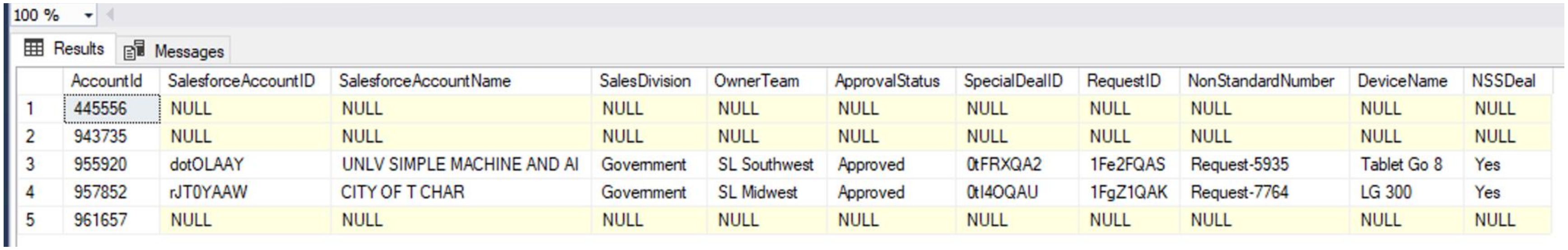
	AccountId	Churn	Credit
1	445556	Churn	11
2	943735	Churn	2163
3	955920	Churn	2974
4	957852	Churn	123
5	961657	Churn	93



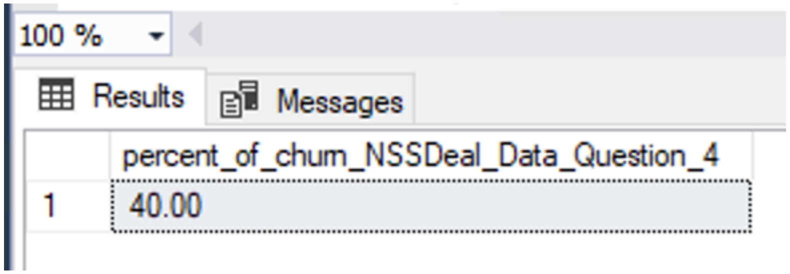
	Average_Credit_Data_Question_3
1	1072.00

```
/* Data Question 4

4) Of those accounts that churn, what percent of them are NSS Deals?

*/

-- Query to select those accounts that churn and of those accounts that churned, what percent of them were designated as an NSS Deal?
SELECT
CAST(CAST(COUNT(a.NSSDeal) as decimal (12,2)) / CAST(COUNT(a.AccountId) as decimal (12,2)) as decimal (12,2))*100 as percent_of_churn_NSSDeal_Data_Question_4
FROM
(SELECT
a.AccountId,
b.SalesforceAccountID,
b.SalesforceAccountName,
c.SalesDivision,
c.OwnerTeam,
```

```sql
c.ApprovalStatus,
c.SpecialDealID,
d.RequestID,
d.NonStandardNumber,
d.DeviceName,
d.NSSDeal
FROM AccountsChurned a
LEFT OUTER JOIN Salesforce b on b.AccountId = a.AccountId
LEFT OUTER JOIN SpecialDeals c on c.AccountName = b.SalesforceAccountName
LEFT OUTER JOIN Request d on d.SpecialDealID = c.SpecialDealID
) a

 /* Data Question 4 Answer

 4)  40.00%

 */
```



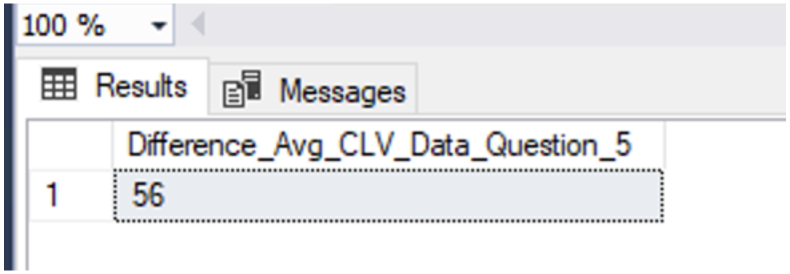
	AccountId	SalesforceAccountID	SalesforceAccountName	SalesDivision	OwnerTeam	ApprovalStatus	SpecialDealID	RequestID	NonStandardNumber	DeviceName	NSSDeal
1	445556	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
2	943735	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
3	955920	dotOLAAY	UNLV SIMPLE MACHINE AND AI	Government	SL Southwest	Approved	0tFRXQA2	1Fe2FQAS	Request-5935	Tablet Go 8	Yes
4	957852	rJT0YAAW	CITY OF T CHAR	Government	SL Midwest	Approved	0tI4OQAU	1FgZ1QAK	Request-7764	LG 300	Yes
5	961657	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



	percent_of_chum_NSSDeal_Data_Question_4
1	40.00

```sql
 /* Data Question 5

 5) What is the difference in the average Net CLV of those accounts that churn from the average NET CLV of those accounts that do not churn?

 */
```

```
GO

-- Creating a view to retreive the average NET CLV of those accounts that churn
CREATE VIEW NetCLVChurn AS (
SELECT
AVG(a.NetCLV) as Average_CLV
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'Churn'

)

GO

-- Creating a view to retreive the average NET CLV of those accounts that do not churn
CREATE VIEW NetCLVNoChurn AS (
SELECT
AVG(a.NetCLV) as Average_CLV
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'No Churn'

)

GO

-- Query to get the difference in Net CLVs (accounts that churn vs. accounts that do not churn)
SELECT
SUM(a.Average_CLV) as Difference_Avg_CLV_Data_Question_5
FROM
(
SELECT
Average_CLV as Average_CLV
FROM NetCLVChurn
UNION
SELECT
Average_CLV*-1 as Average_CLV
FROM NetCLVNoChurn
) a

 /* Data Question 5 Answer

 5)  56
 */
```

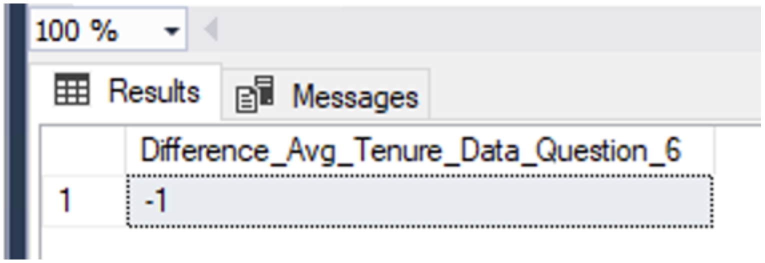


	Difference_Avg_CLV_Data_Question_5
1	56

```
 /* Data Question 6

 6)  What is the difference in the average Tenure of those accounts that churn from the average Tenure of those accounts that do not churn?

 */
GO

--Creating a view to retreive the average tenure of those accounts that churn
CREATE VIEW AvgTenureChurn AS (
SELECT
AVG(a.Tenure60) as Average_Tenure
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'Churn'

)

GO

--Creating a view to retreive the average tenure of those accounts that do not churn
CREATE VIEW AvgTenureNoChurn AS (
SELECT
AVG(a.Tenure60) as Average_Tenure
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'No Churn'

)

GO
```

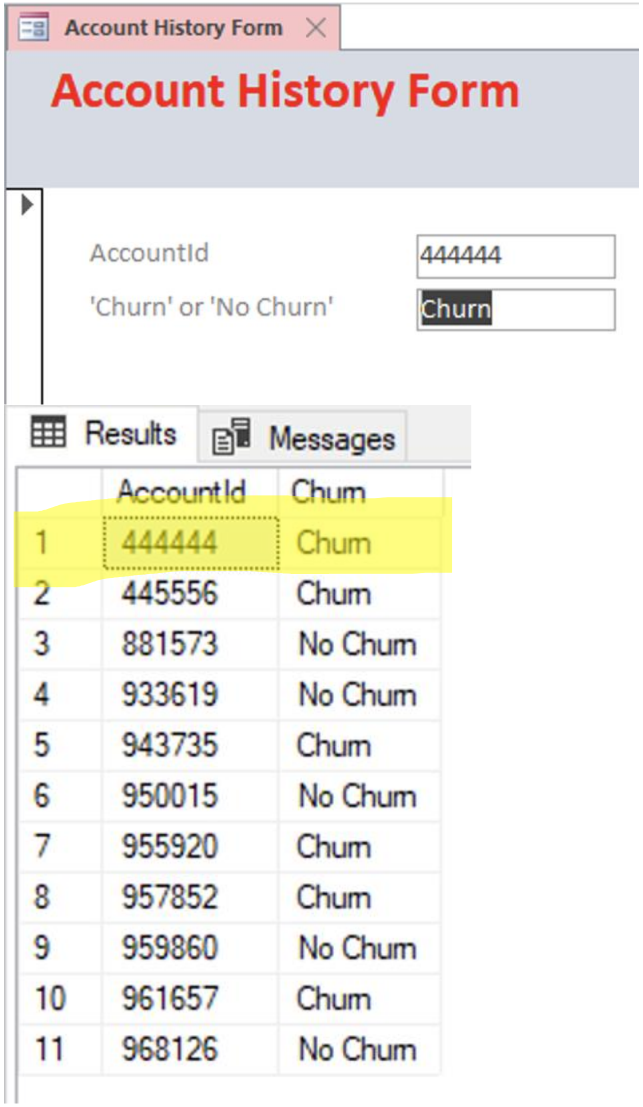
```
--Query to determine the difference in average tenure between those accounts that churn and those accounts that do not churn
SELECT
SUM(a.Average_Tenure) as Difference_Avg_Tenure_Data_Question_6
FROM
(
SELECT
Average_Tenure as Average_Tenure
FROM AvgTenureChurn
UNION
SELECT
Average_Tenure*-1 as Average_Tenure
FROM AvgTenureNoChurn
) a

GO

-- Average tenure is lower by 1 momth for those accounts that churn

SELECT * FROM AvgTenureChurn
SELECT * FROM AvgTenureNoChurn
```

```
 /* Data Question 6 Answer

 6)  Average tenure is lower by 1 month for those accounts that churn

 */
```



	Difference_Avg_Tenure_Data_Question_6
1	-1

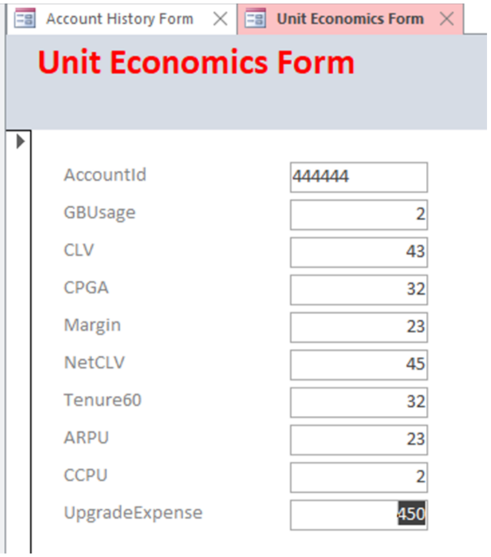
/*--- MS ACCESS FORMS – ADDING IN NEW DATA ---*/



Account History Form

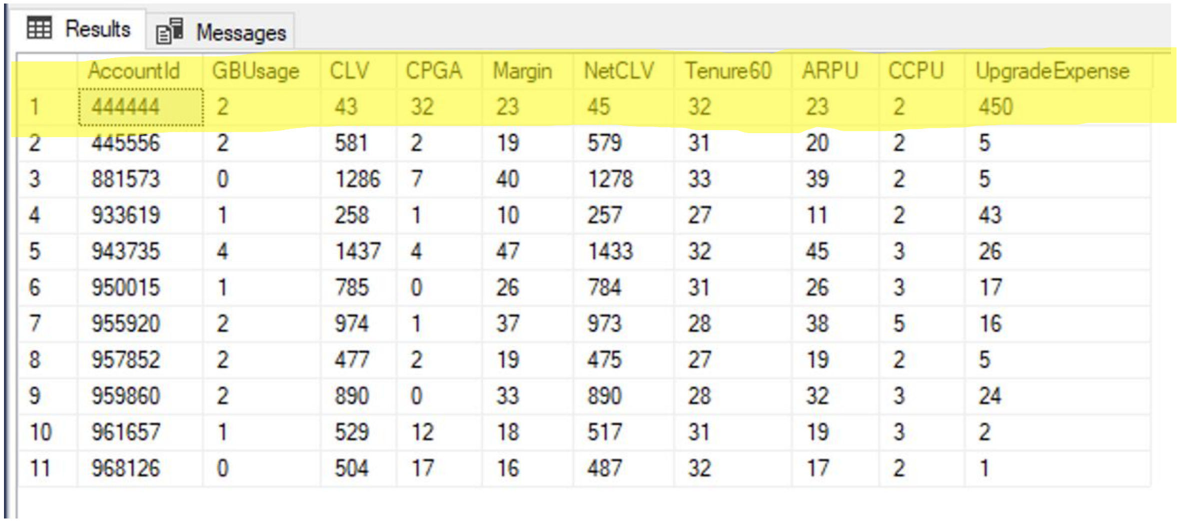
AccountId	444444
'Churn' or 'No Churn'	Churn

	AccountId	Churn
1	444444	Churn
2	445556	Churn
3	881573	No Churn
4	933619	No Churn
5	943735	Churn
6	950015	No Churn
7	955920	Churn
8	957852	Churn
9	959860	No Churn
10	961657	Churn
11	968126	No Churn

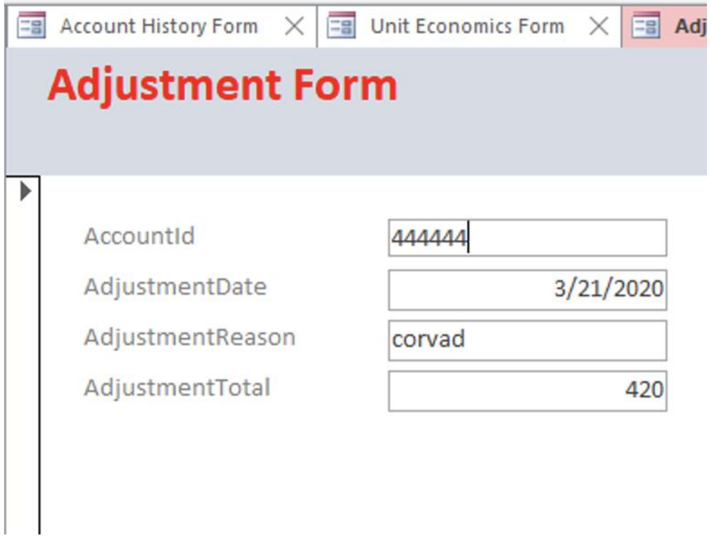


Unit Economics Form

AccountId	444444
GBUsage	2
CLV	43
CPGA	32
Margin	23
NetCLV	45
Tenure60	32
ARPU	23
CCPU	2
UpgradeExpense	450

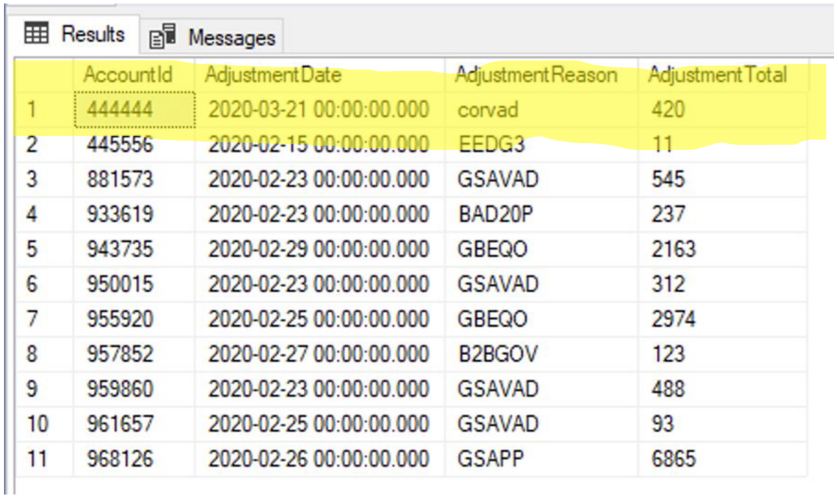


	AccountId	GBUsage	CLV	CPGA	Margin	NetCLV	Tenure60	ARPU	CCPU	UpgradeExpense
1	444444	2	43	32	23	45	32	23	2	450
2	445556	2	581	2	19	579	31	20	2	5
3	881573	0	1286	7	40	1278	33	39	2	5
4	933619	1	258	1	10	257	27	11	2	43
5	943735	4	1437	4	47	1433	32	45	3	26
6	950015	1	785	0	26	784	31	26	3	17
7	955920	2	974	1	37	973	28	38	5	16
8	957852	2	477	2	19	475	27	19	2	5
9	959860	2	890	0	33	890	28	32	3	24
10	961657	1	529	12	18	517	31	19	3	2
11	968126	0	504	17	16	487	32	17	2	1

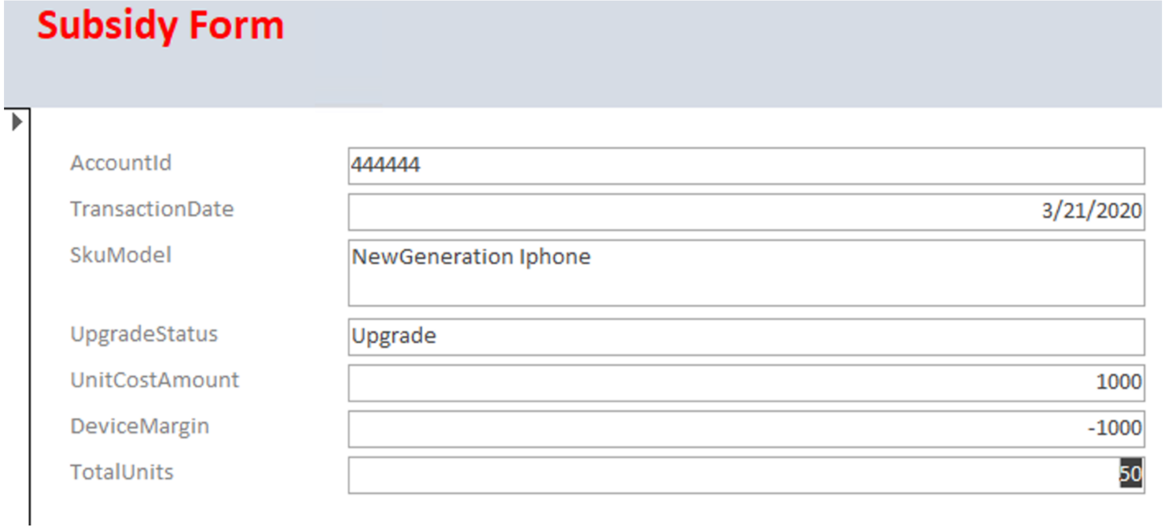


Adjustment Form

AccountId	444444
AdjustmentDate	3/21/2020
AdjustmentReason	corvad
AdjustmentTotal	420

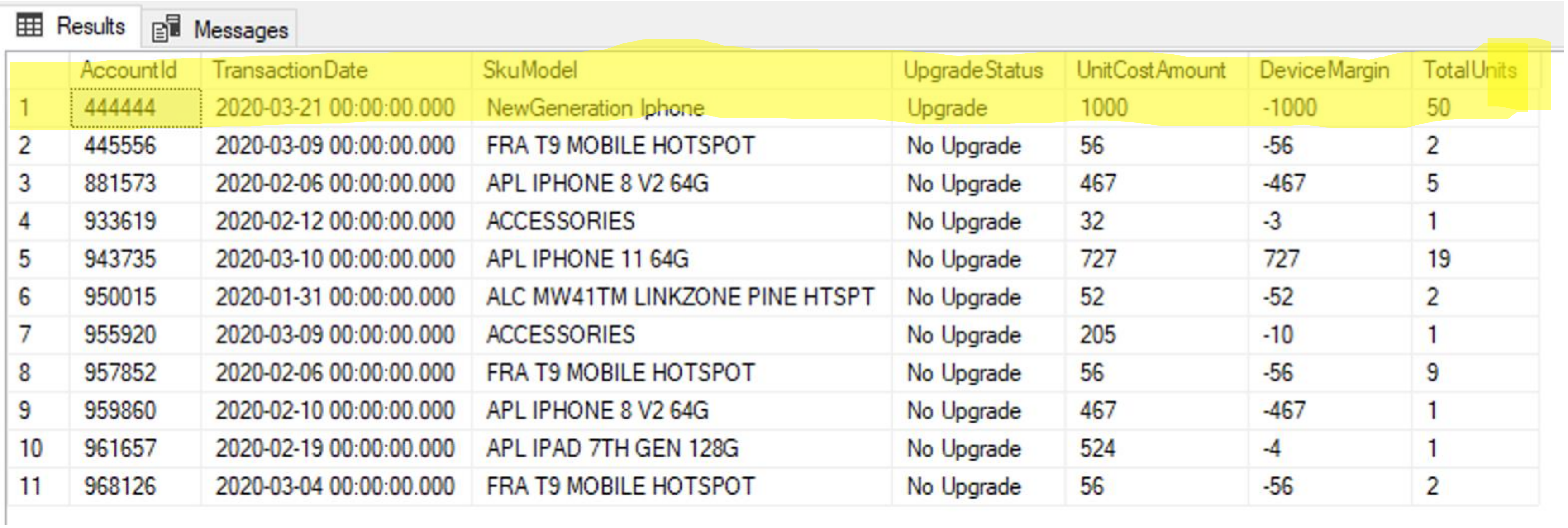


	AccountId	AdjustmentDate	AdjustmentReason	AdjustmentTotal
1	444444	2020-03-21 00:00:00.000	corvad	420
2	445556	2020-02-15 00:00:00.000	EEDG3	11
3	881573	2020-02-23 00:00:00.000	GSAVAD	545
4	933619	2020-02-23 00:00:00.000	BAD20P	237
5	943735	2020-02-29 00:00:00.000	GBEQO	2163
6	950015	2020-02-23 00:00:00.000	GSAVAD	312
7	955920	2020-02-25 00:00:00.000	GBEQO	2974
8	957852	2020-02-27 00:00:00.000	B2BGOV	123
9	959860	2020-02-23 00:00:00.000	GSAVAD	488
10	961657	2020-02-25 00:00:00.000	GSAVAD	93
11	968126	2020-02-26 00:00:00.000	GSAPP	6865

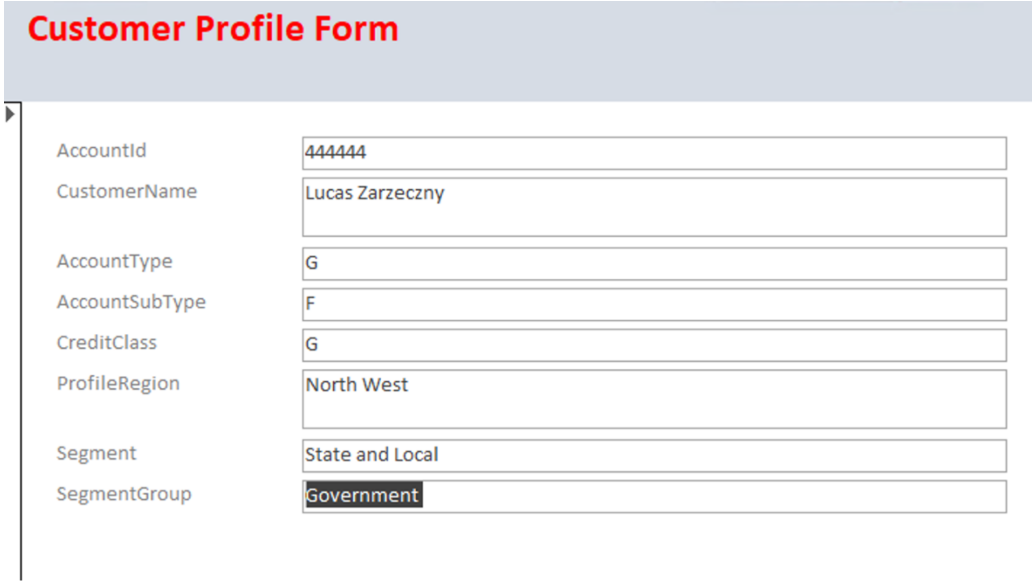


Subsidy Form

AccountId	444444
TransactionDate	3/21/2020
SkuModel	NewGeneration Iphone
UpgradeStatus	Upgrade
UnitCostAmount	1000
DeviceMargin	-1000
TotalUnits	50

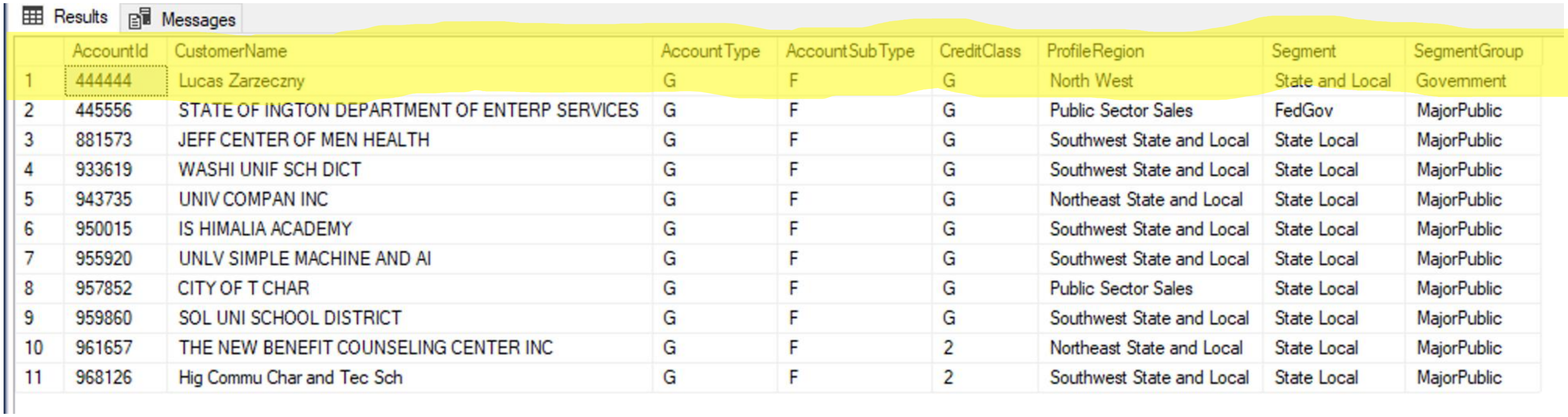


	AccountId	TransactionDate	SkuModel	UpgradeStatus	UnitCostAmount	DeviceMargin	TotalUnits
1	444444	2020-03-21 00:00:00.000	NewGeneration Iphone	Upgrade	1000	-1000	50
2	445556	2020-03-09 00:00:00.000	FRA T9 MOBILE HOTSPOT	No Upgrade	56	-56	2
3	881573	2020-02-06 00:00:00.000	APL IPHONE 8 V2 64G	No Upgrade	467	-467	5
4	933619	2020-02-12 00:00:00.000	ACCESSORIES	No Upgrade	32	-3	1
5	943735	2020-03-10 00:00:00.000	APL IPHONE 11 64G	No Upgrade	727	727	19
6	950015	2020-01-31 00:00:00.000	ALC MW41TM LINKZONE PINE HTSPT	No Upgrade	52	-52	2
7	955920	2020-03-09 00:00:00.000	ACCESSORIES	No Upgrade	205	-10	1
8	957852	2020-02-06 00:00:00.000	FRA T9 MOBILE HOTSPOT	No Upgrade	56	-56	9
9	959860	2020-02-10 00:00:00.000	APL IPHONE 8 V2 64G	No Upgrade	467	-467	1
10	961657	2020-02-19 00:00:00.000	APL IPAD 7TH GEN 128G	No Upgrade	524	-4	1
11	968126	2020-03-04 00:00:00.000	FRA T9 MOBILE HOTSPOT	No Upgrade	56	-56	2

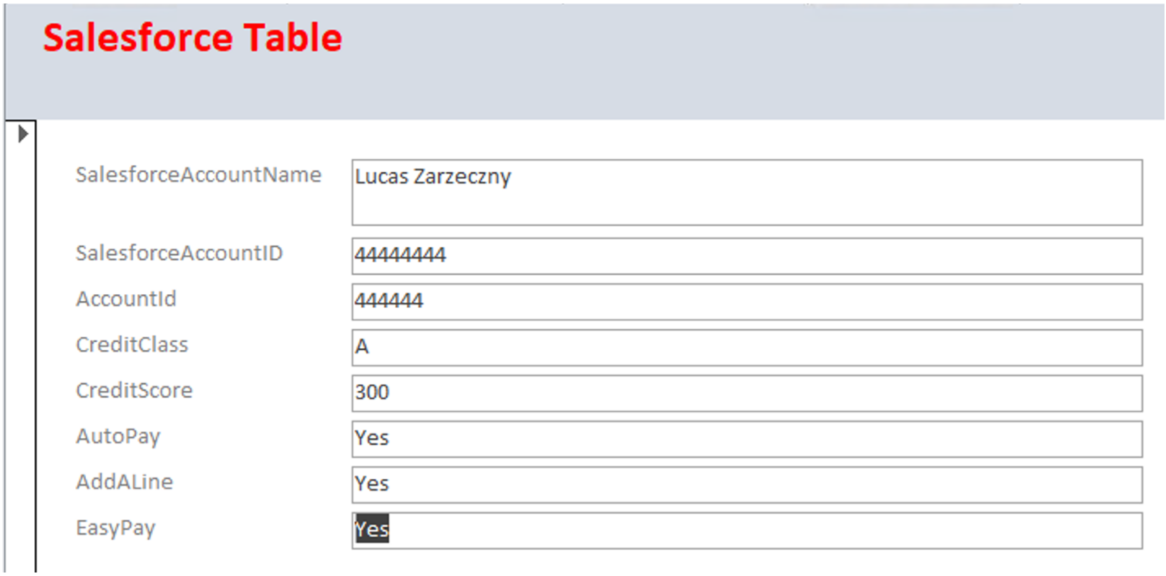


Customer Profile Form

AccountId	444444
CustomerName	Lucas Zarzeczny
AccountType	G
AccountSubType	F
CreditClass	G
ProfileRegion	North West
Segment	State and Local
SegmentGroup	Government

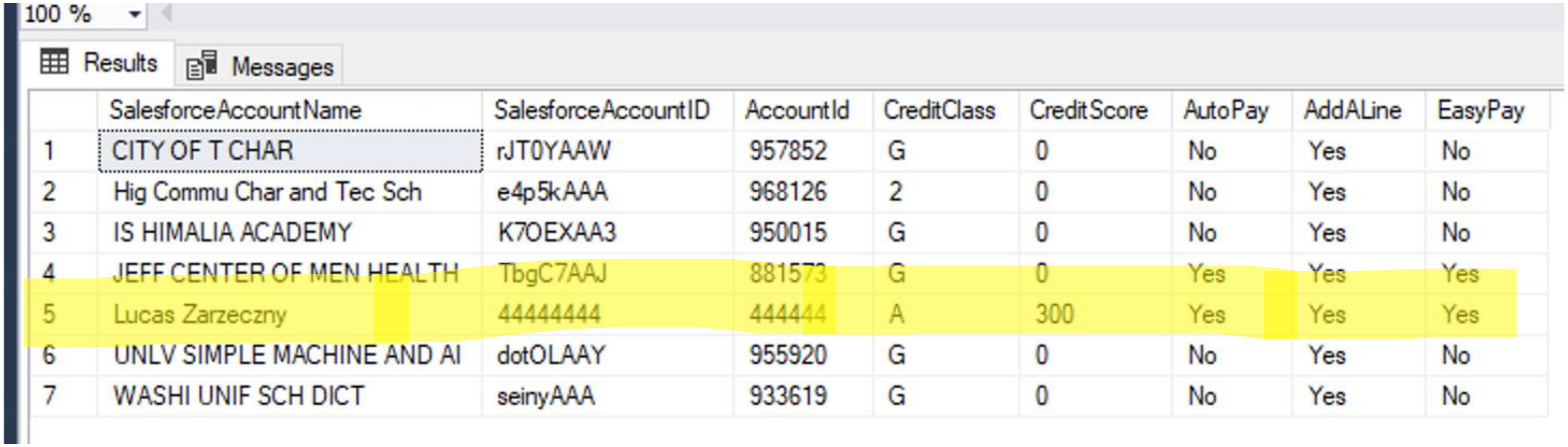


	AccountId	CustomerName	AccountType	AccountSubType	CreditClass	ProfileRegion	Segment	SegmentGroup
1	444444	Lucas Zarzeczny	G	F	G	North West	State and Local	Government
2	445556	STATE OF INGTON DEPARTMENT OF ENTERP SERVICES	G	F	G	Public Sector Sales	FedGov	MajorPublic
3	881573	JEFF CENTER OF MEN HEALTH	G	F	G	Southwest State and Local	State Local	MajorPublic
4	933619	WASHI UNIF SCH DICT	G	F	G	Southwest State and Local	State Local	MajorPublic
5	943735	UNIV COMPAN INC	G	F	G	Northeast State and Local	State Local	MajorPublic
6	950015	IS HIMALIA ACADEMY	G	F	G	Southwest State and Local	State Local	MajorPublic
7	955920	UNLV SIMPLE MACHINE AND AI	G	F	G	Southwest State and Local	State Local	MajorPublic
8	957852	CITY OF T CHAR	G	F	G	Public Sector Sales	State Local	MajorPublic
9	959860	SOL UNI SCHOOL DISTRICT	G	F	G	Southwest State and Local	State Local	MajorPublic
10	961657	THE NEW BENEFIT COUNSELING CENTER INC	G	F	2	Northeast State and Local	State Local	MajorPublic
11	968126	Hig Commu Char and Tec Sch	G	F	2	Southwest State and Local	State Local	MajorPublic

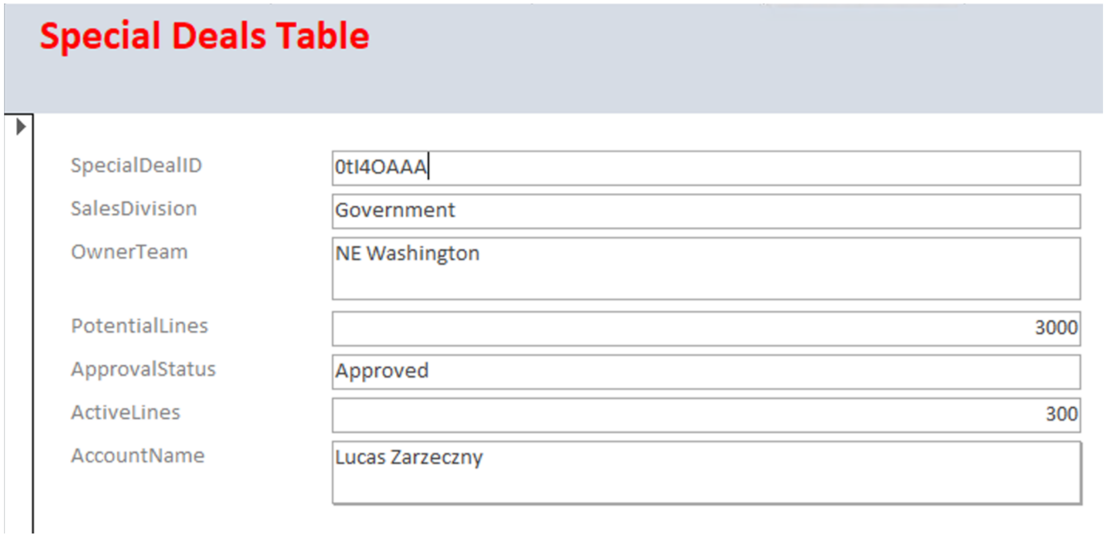


Salesforce Table

SalesforceAccountName	Lucas Zarzeczny
SalesforceAccountID	44444444
AccountId	444444
CreditClass	A
CreditScore	300
AutoPay	Yes
AddALine	Yes
EasyPay	Yes

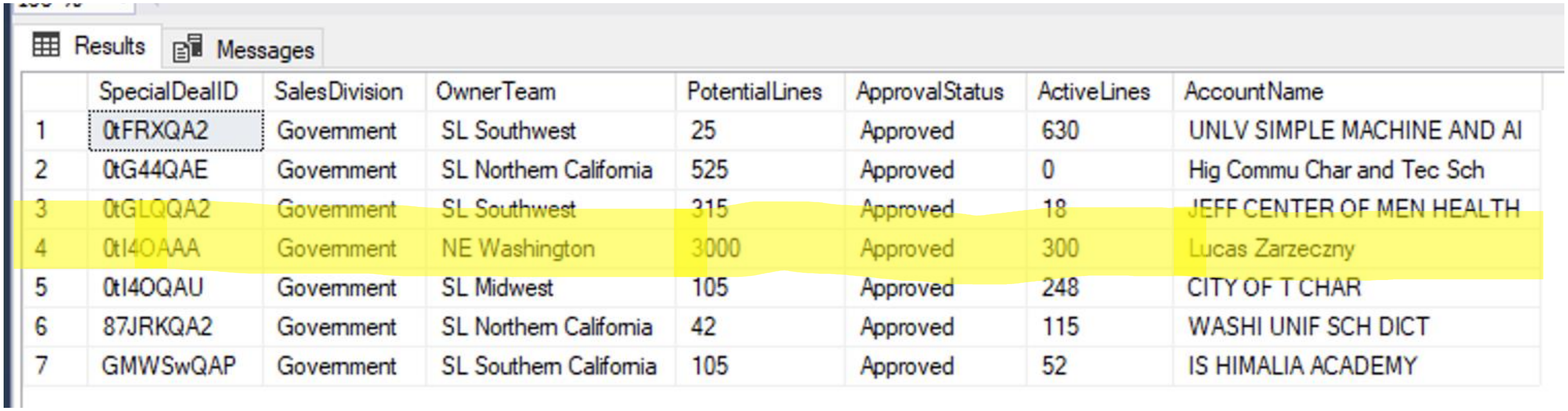


	SalesforceAccountName	SalesforceAccountID	AccountId	CreditClass	CreditScore	AutoPay	AddALine	EasyPay
1	CITY OF T CHAR	rJT0YAAW	957852	G	0	No	Yes	No
2	Hig Commu Char and Tec Sch	e4p5kAAA	968126	2	0	No	Yes	No
3	IS HIMALIA ACADEMY	K7OEXAA3	950015	G	0	No	Yes	No
4	JEFF CENTER OF MEN HEALTH	TbgC7AAJ	881573	G	0	Yes	Yes	Yes
5	Lucas Zarzeczny	44444444	444444	A	300	Yes	Yes	Yes
6	UNLV SIMPLE MACHINE AND AI	dotOLAAY	955920	G	0	No	Yes	No
7	WASHI UNIF SCH DICT	seinyAAA	933619	G	0	No	Yes	No

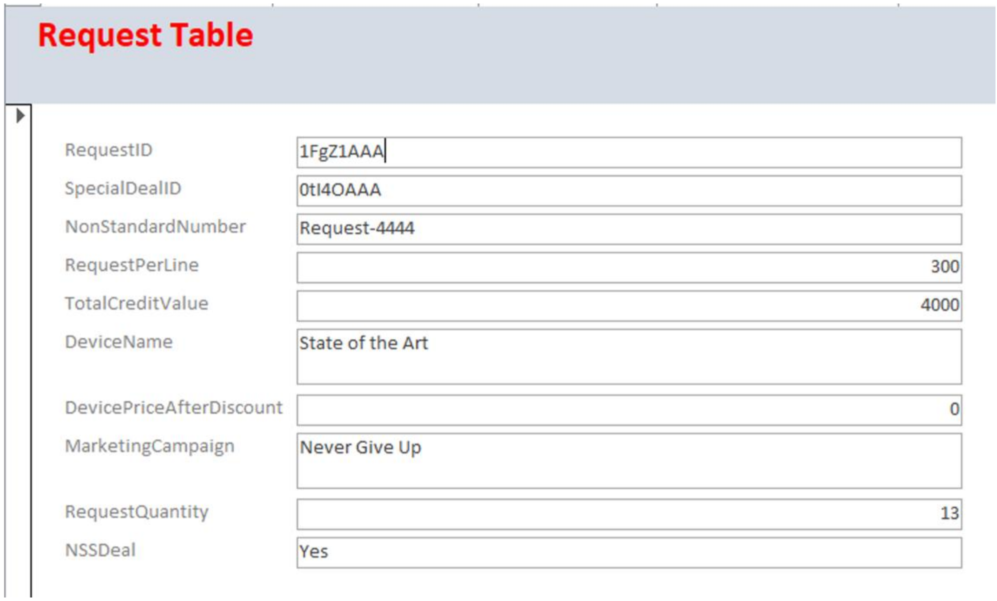


Special Deals Table

SpecialDealID	0tI4OAAA
SalesDivision	Government
OwnerTeam	NE Washington
PotentialLines	3000
ApprovalStatus	Approved
ActiveLines	300
AccountName	Lucas Zarzeczny

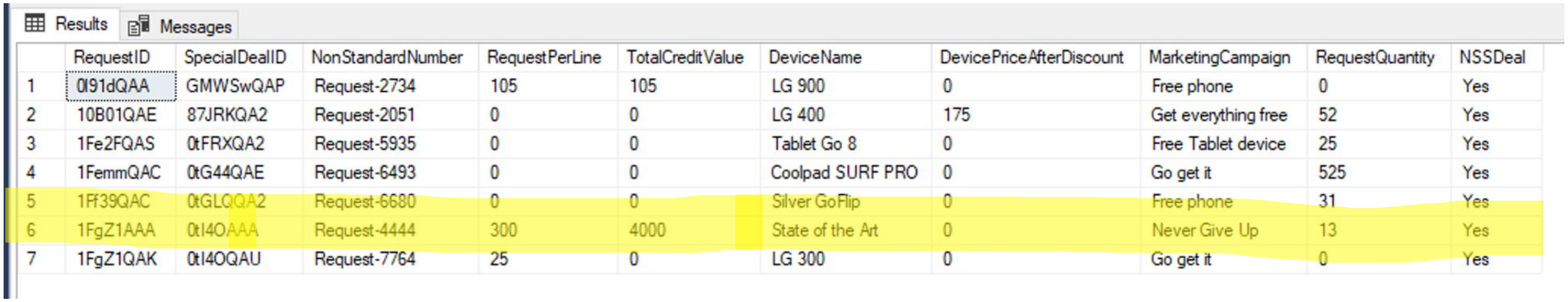


	SpecialDealID	SalesDivision	OwnerTeam	PotentialLines	ApprovalStatus	ActiveLines	AccountName
1	0tFRXQA2	Government	SL Southwest	25	Approved	630	UNLV SIMPLE MACHINE AND AI
2	0tG44QAE	Government	SL Northern California	525	Approved	0	Hig Commu Char and Tec Sch
3	0tGLQQA2	Government	SL Southwest	315	Approved	18	JEFF CENTER OF MEN HEALTH
4	0tI4OAAA	Government	NE Washington	3000	Approved	300	Lucas Zarzeczny
5	0tI4OQAU	Government	SL Midwest	105	Approved	248	CITY OF T CHAR
6	87JRKQA2	Government	SL Northern California	42	Approved	115	WASHI UNIF SCH DICT
7	GMWSwQAP	Government	SL Southern California	105	Approved	52	IS HIMALIA ACADEMY



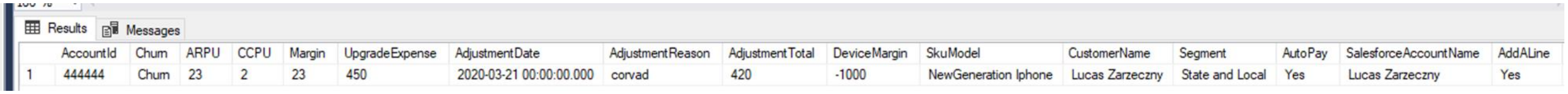
Request Table

RequestID	1FgZ1AAA
SpecialDealID	0tI4OAAA
NonStandardNumber	Request-4444
RequestPerLine	300
TotalCreditValue	4000
DeviceName	State of the Art
DevicePriceAfterDiscount	0
MarketingCampaign	Never Give Up
RequestQuantity	13
NSSDeal	Yes

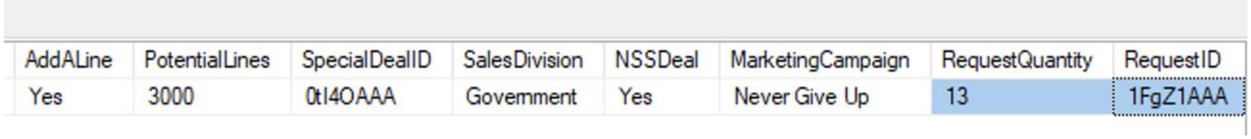


	RequestID	SpecialDealID	NonStandardNumber	RequestPerLine	TotalCreditValue	DeviceName	DevicePriceAfterDiscount	MarketingCampaign	RequestQuantity	NSSDeal
1	0I91dQAA	GMWSwQAP	Request-2734	105	105	LG 900	0	Free phone	0	Yes
2	10B01QAE	87JRKQA2	Request-2051	0	0	LG 400	175	Get everything free	52	Yes
3	1Fe2FQAS	0tFRXQA2	Request-5935	0	0	Tablet Go 8	0	Free Tablet device	25	Yes
4	1FemmQAC	0tG44QAE	Request-6493	0	0	Coolpad SURF PRO	0	Go get it	525	Yes
5	1Ff39QAC	0tGLQQA2	Request-6680	0	0	Silver GoFlip	0	Free phone	31	Yes
6	1FgZ1AAA	0tI4OAAA	Request-4444	300	4000	State of the Art	0	Never Give Up	13	Yes
7	1FgZ1QAK	0tI4OQAU	Request-7764	25	0	LG 300	0	Go get it	0	Yes

```sql
/*------------------------------------------------------------------------------------------------------------ Query / Table Including New Data From Forms ------------------------------------------------------------------------------------------------------------*/

   -- Query to get all selected information regarding the new account that was added through microsoft access
SELECT
a.AccountId,
a.Churn,
b.ARPU,
b.CCPU,
b.Margin,
b.UpgradeExpense,
c.AdjustmentDate,
c.AdjustmentReason,
c.AdjustmentTotal,
d.DeviceMargin,
d.SkuModel,
e.CustomerName,
e.Segment,
f.AutoPay,
f.SalesforceAccountName,
f.AddALine,
g.PotentialLines,
g.SpecialDealID,
g.SalesDivision,
h.NSSDeal,
h.MarketingCampaign,
h.RequestQuantity,
h.RequestID
FROM hist_account a
INNER JOIN UnitEconomics b on b.AccountId = a.AccountId
INNER JOIN Adjustments c on c.AccountId = a.AccountId
INNER JOIN Subsidy d on d.AccountId = a.AccountId
INNER JOIN CustomerProfile e on e.AccountId = a.AccountId
LEFT OUTER JOIN Salesforce f on f.AccountId = a.AccountId
LEFT OUTER JOIN SpecialDeals g on g.AccountName = f.SalesforceAccountName
LEFT OUTER JOIN Request h on h.SpecialDealID = g.SpecialDealID
WHERE a.AccountId = '444444'
   -- This is new account I added through Microsoft Access
```

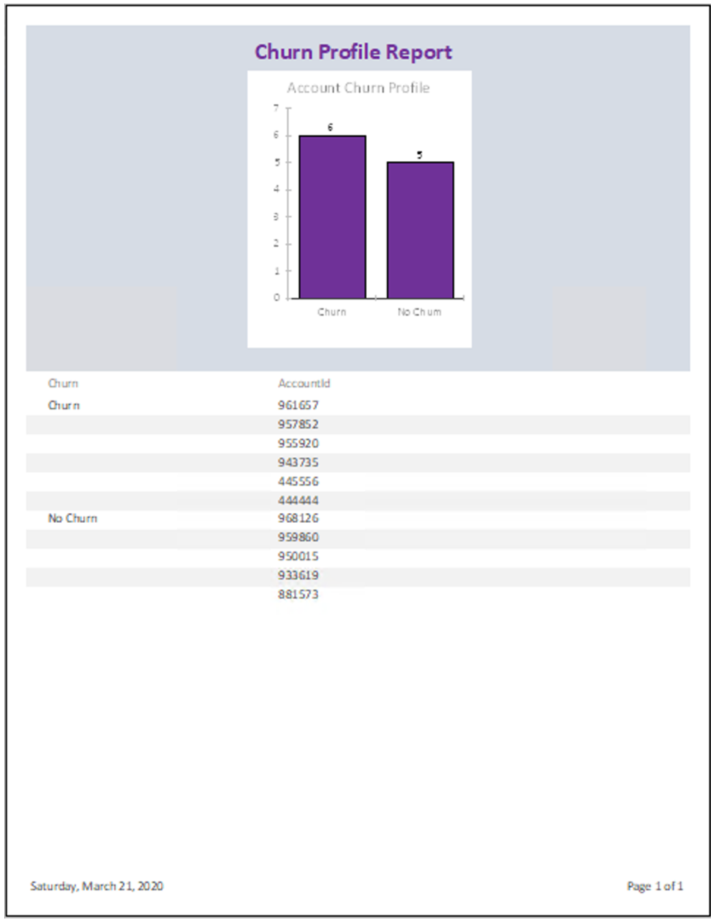


	AccountId	Churn	ARPU	CCPU	Margin	UpgradeExpense	AdjustmentDate	AdjustmentReason	AdjustmentTotal	DeviceMargin	SkuModel	CustomerName	Segment	AutoPay	SalesforceAccountName	AddALine
1	444444	Churn	23	2	23	450	2020-03-21 00:00:00.000	corvad	420	-1000	NewGeneration Iphone	Lucas Zarzeczny	State and Local	Yes	Lucas Zarzeczny	Yes



AddALine	PotentialLines	SpecialDealID	SalesDivision	NSSDeal	MarketingCampaign	RequestQuantity	RequestID
Yes	3000	0tI4OAAA	Government	Yes	Never Give Up	13	1FgZ1AAA

```
/*---------------------------------------------------------------------------------------------- MICROSOFT ACCESS REPORTING ----------------------------------------------------------------------------------------------*/
```

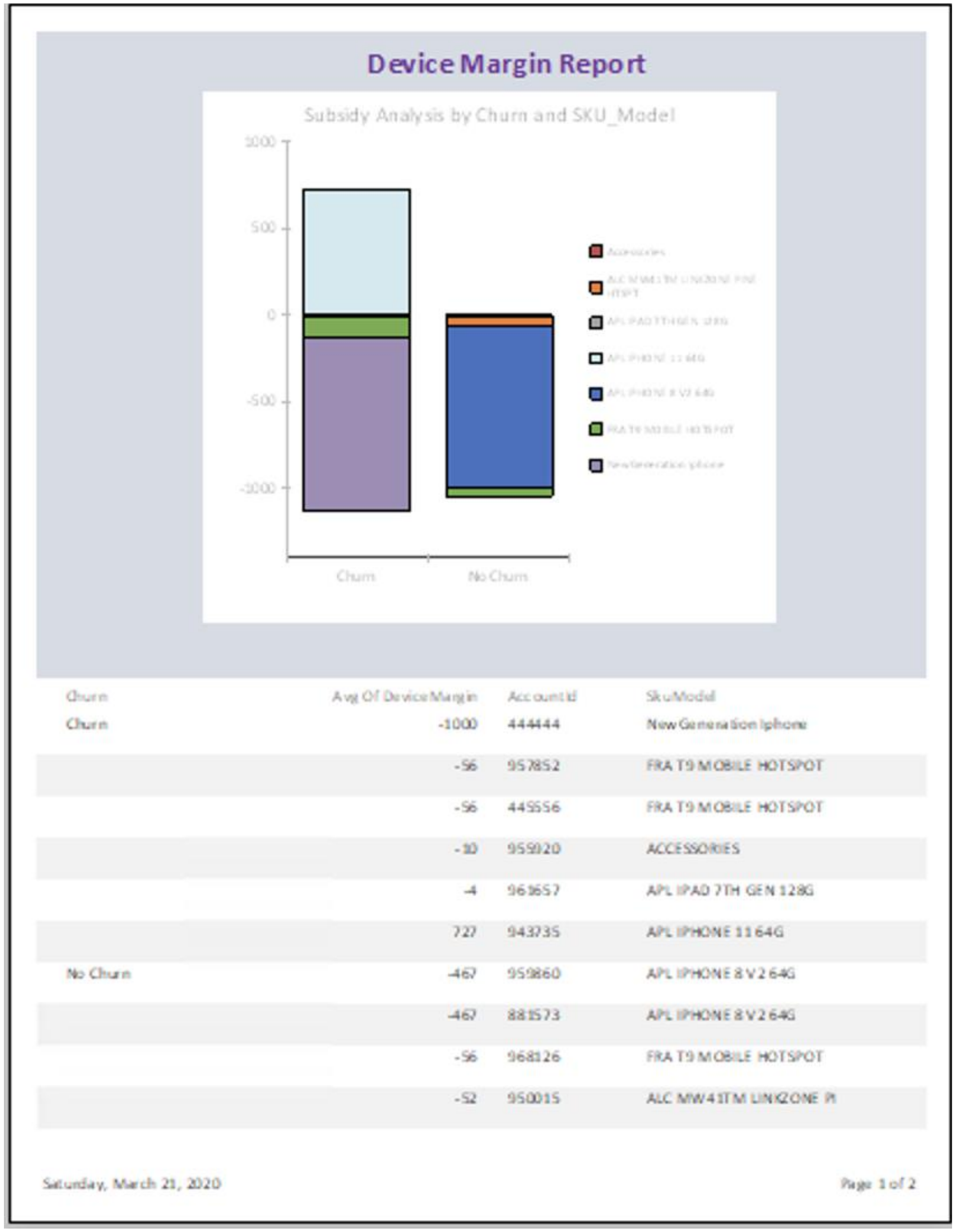


Churn Profile Report

Churn	AccountId
Churn	961657
	957852
	955920
	943735
	445556
	444444
No Churn	968126
	959860
	950015
	933619
	881573

Saturday, March 21, 2020

Page 1 of 1

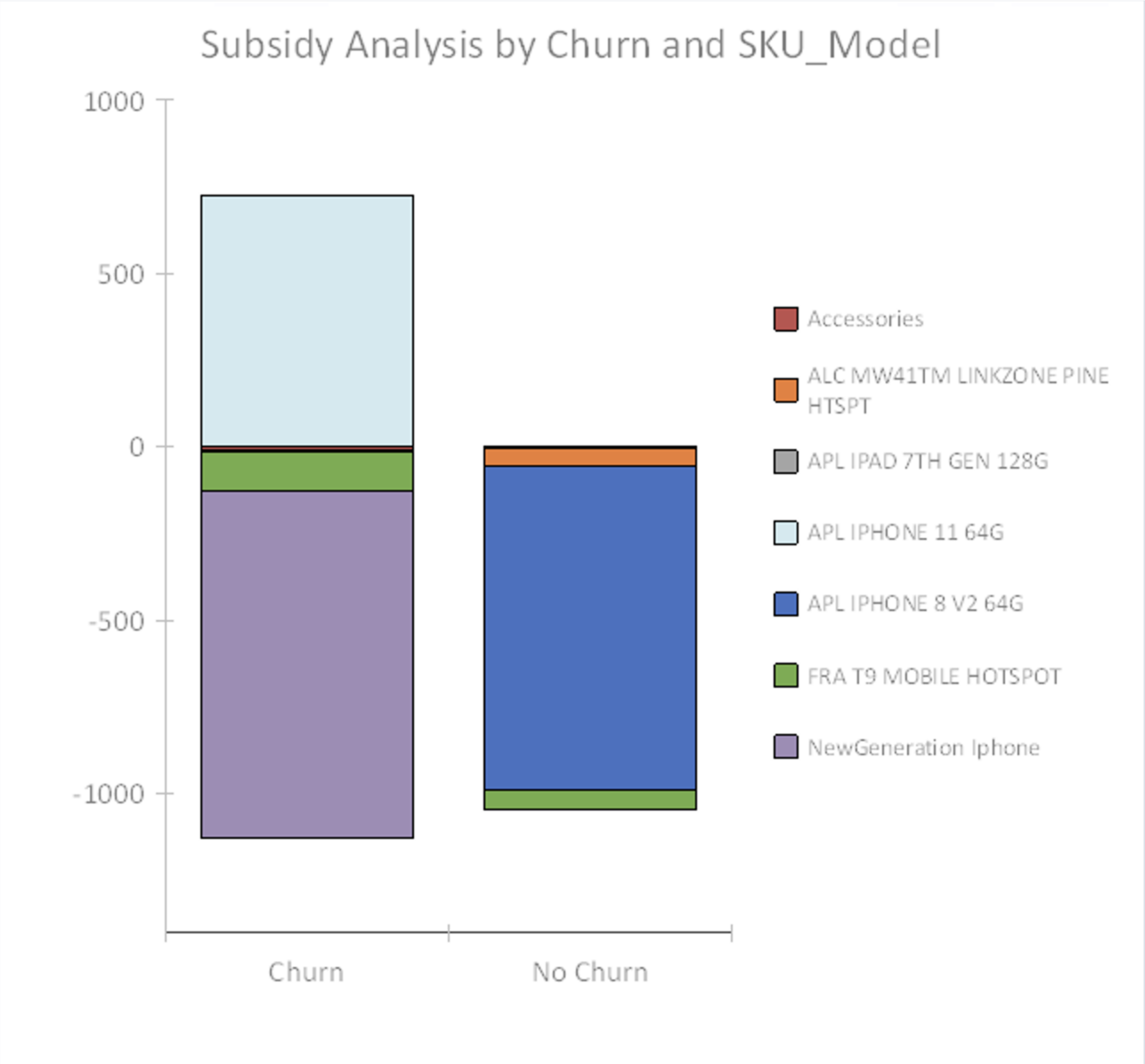


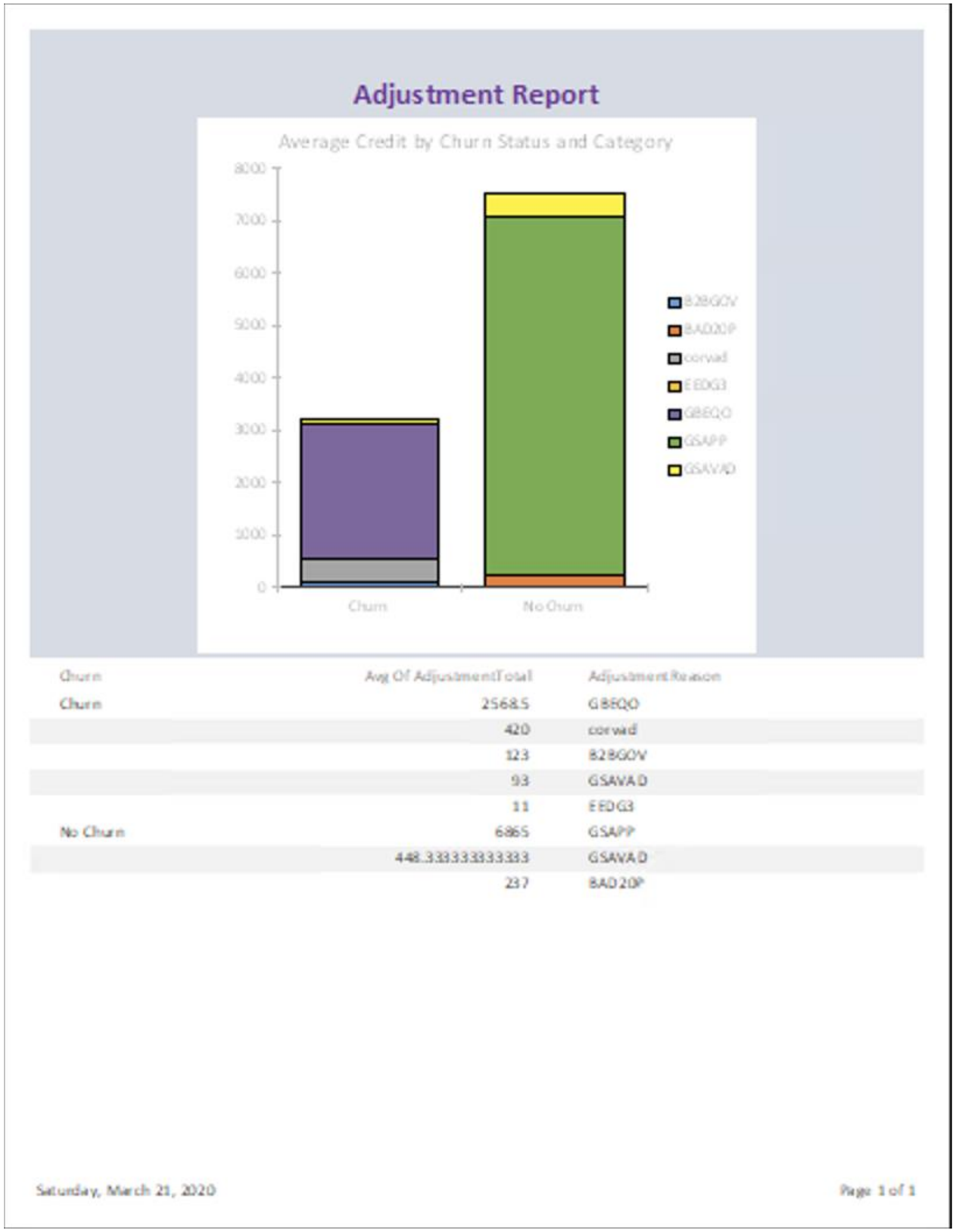
Device Margin Report

Churn	Avg Of DeviceMargin	Accountid	SkuModel
Churn	-1000	444444	New Generation Iphone
	-56	957852	FRA T9 MOBILE HOTSPOT
	-56	445556	FRA T9 MOBILE HOTSPOT
	-10	959920	ACCESSORIES
	-4	961657	APL IPAD 7TH GEN 128G
	727	943735	APL IPHONE 11 64G
No Churn	-467	959860	APL IPHONE 8 V2 64G
	-467	881573	APL IPHONE 8 V2 64G
	-56	968126	FRA T9 MOBILE HOTSPOT
	-52	950015	ALC MW41TM LINKZONE PI

Saturday, March 21, 2020

Page 1 of 2


Subsidy Analysis by Churn and SKU_Model
1000
500
0
-500
-1000
Churn
No Churn
Accessories
ALC MW41TM LINKZONE PINE HTSPT
APL IPAD 7TH GEN 128G
APL IPHONE 11 64G
APL IPHONE 8 V2 64G
FRA T9 MOBILE HOTSPOT
NewGeneration Iphone



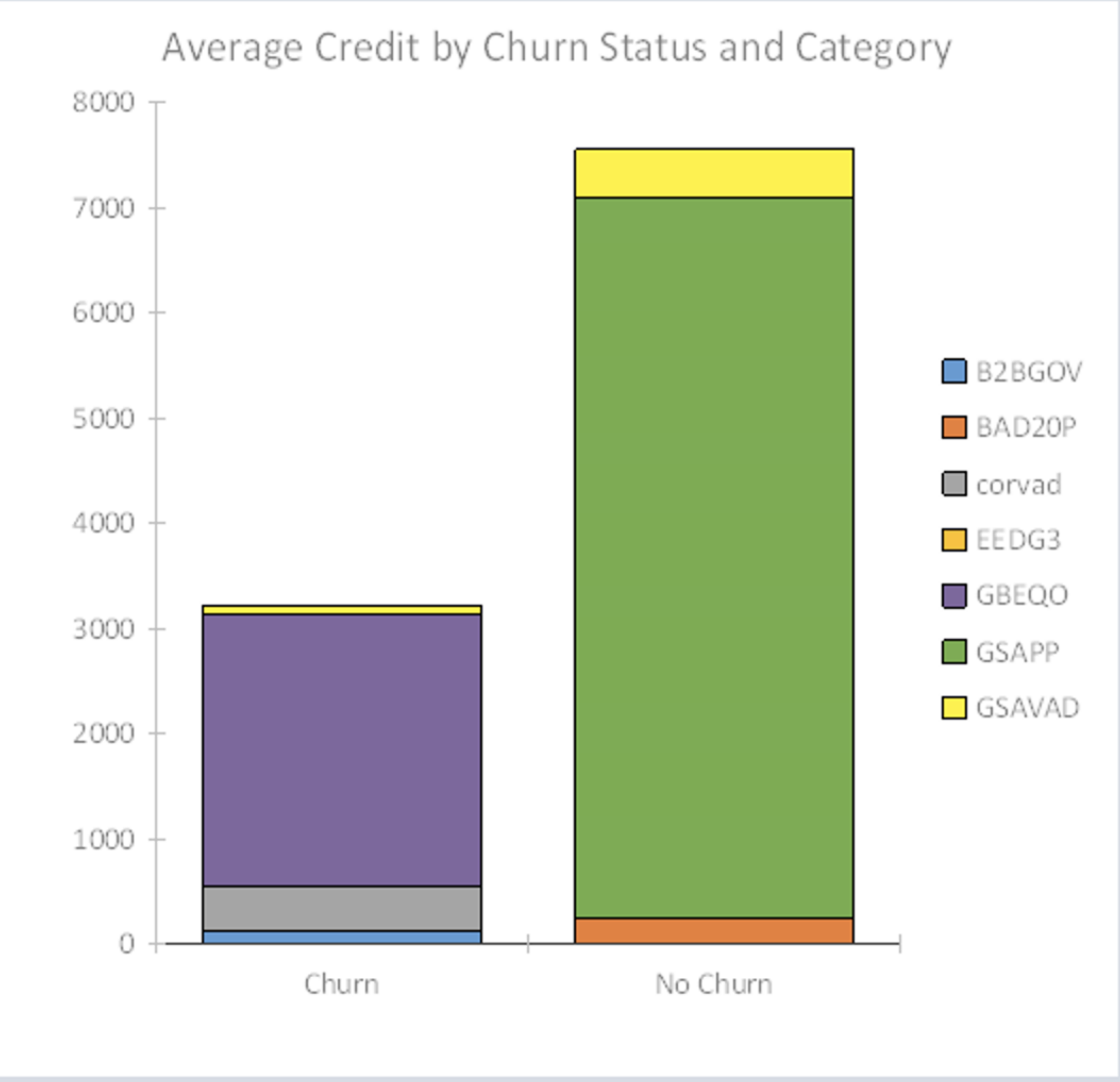
Adjustment Report

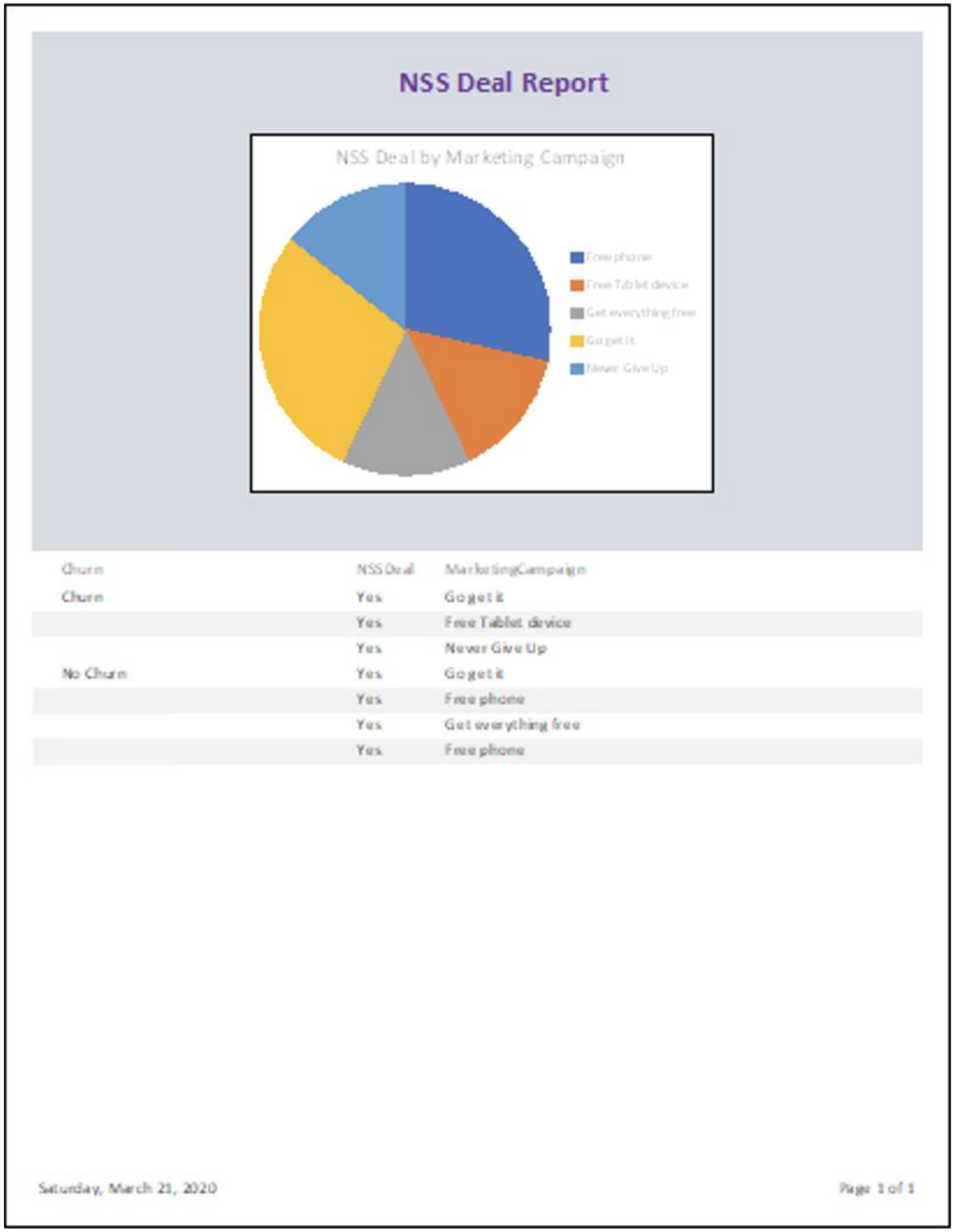
Average Credit by Churn Status and Category

Churn	Avg Of AdjustmentTotal	AdjustmentReason
Churn	2568.5	GBEQO
	420	corvaid
	123	B2BGOV
	93	GSAVAD
	11	EEDG3
No Churn	6865	GSAPP
	448.333333333333	GSAVAD
	237	BAD20P

Saturday, March 21, 2020

Page 1 of 1


Average Credit by Churn Status and Category
8000
7000
6000
5000
4000
3000
2000
1000
0
Churn
No Churn
B2BGOV
BAD20P
corvad
EEDG3
GBEQO
GSAPP
GSAVAD

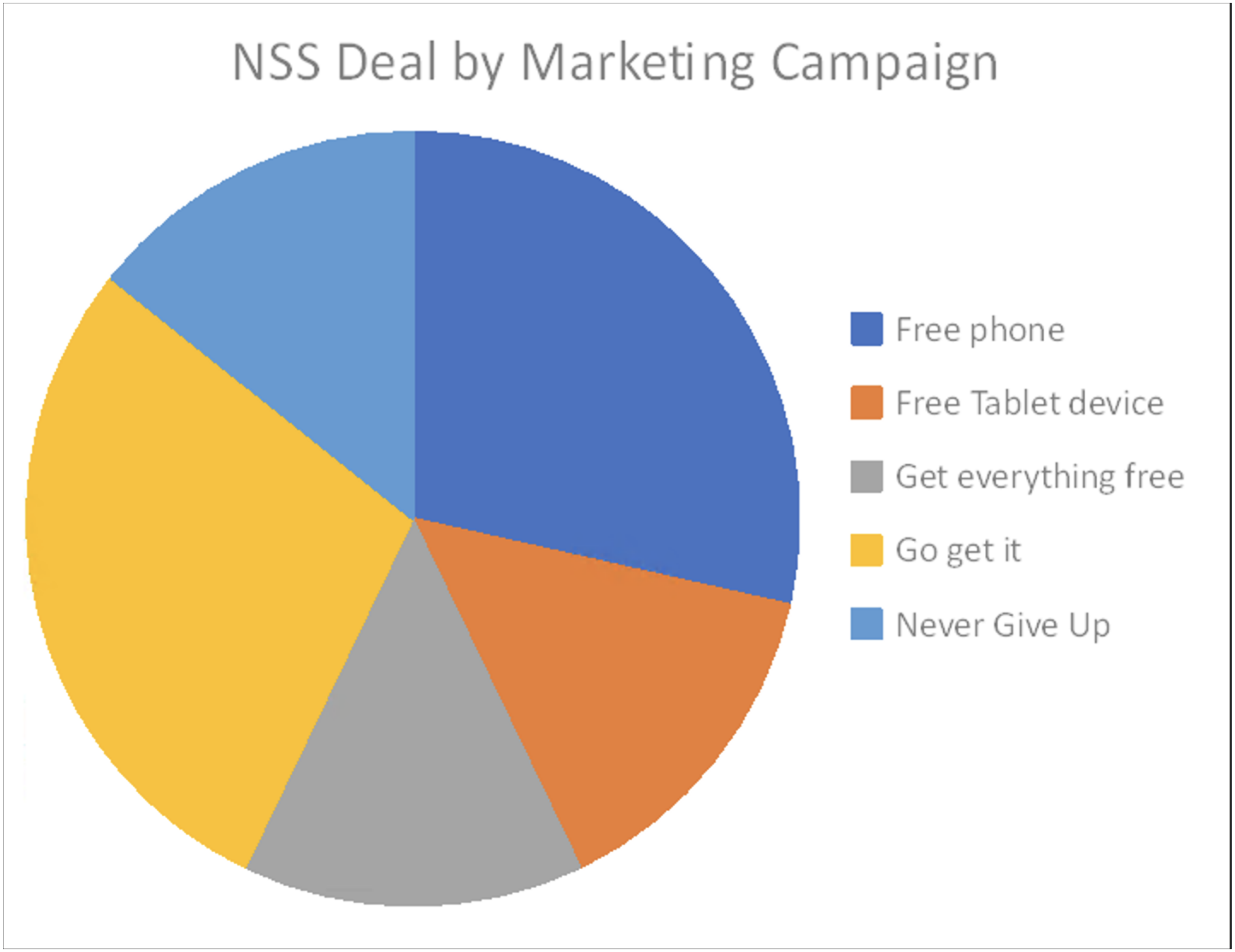


NSS Deal Report

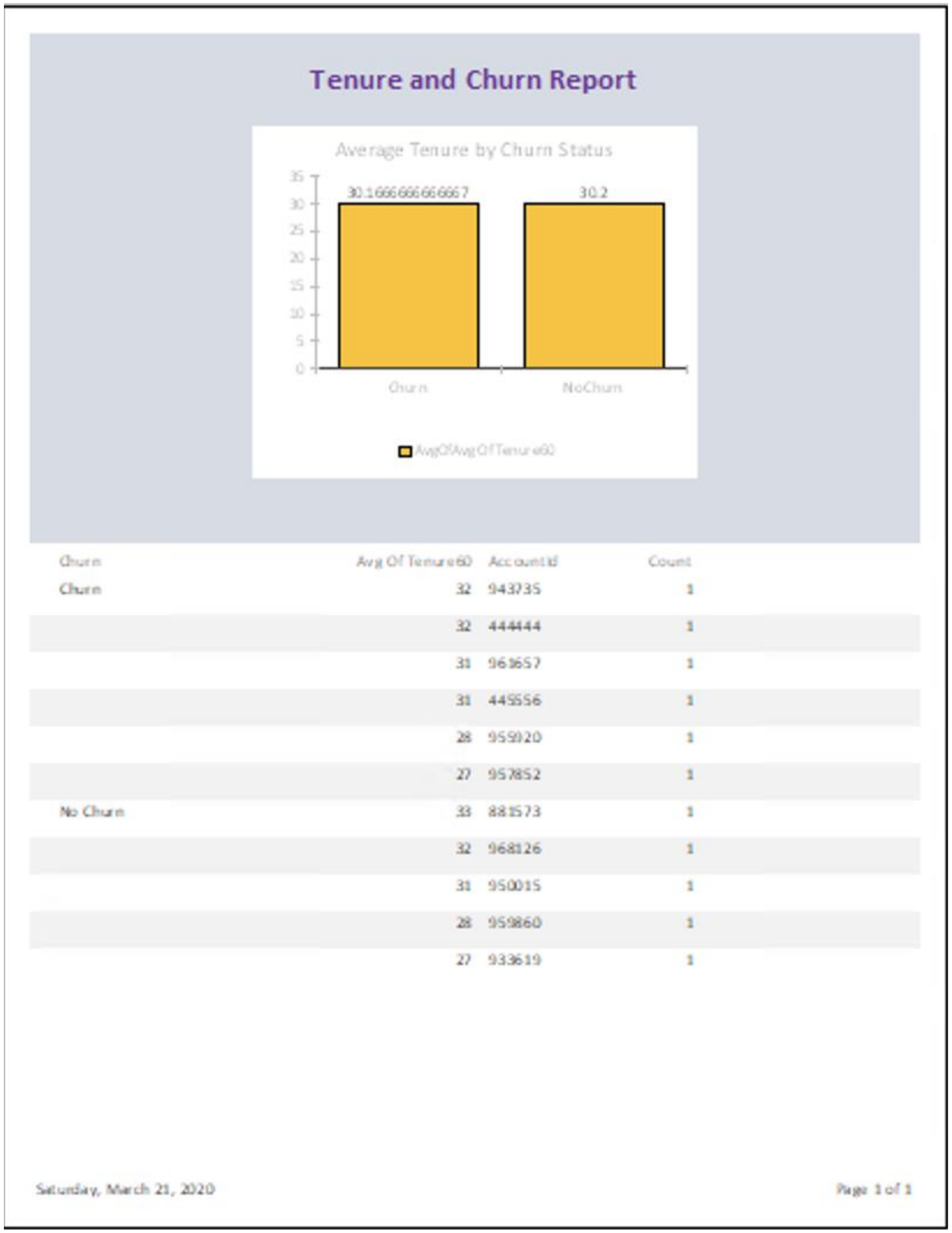
Churn	NSSDeal	MarketingCampaign
Churn	Yes	Go get it
	Yes	Free Tablet device
	Yes	Never Give Up
No Churn	Yes	Go get it
	Yes	Free phone
	Yes	Get everything free
	Yes	Free phone

Saturday, March 21, 2020

Page 1 of 1



NSS Deal by Marketing Campaign

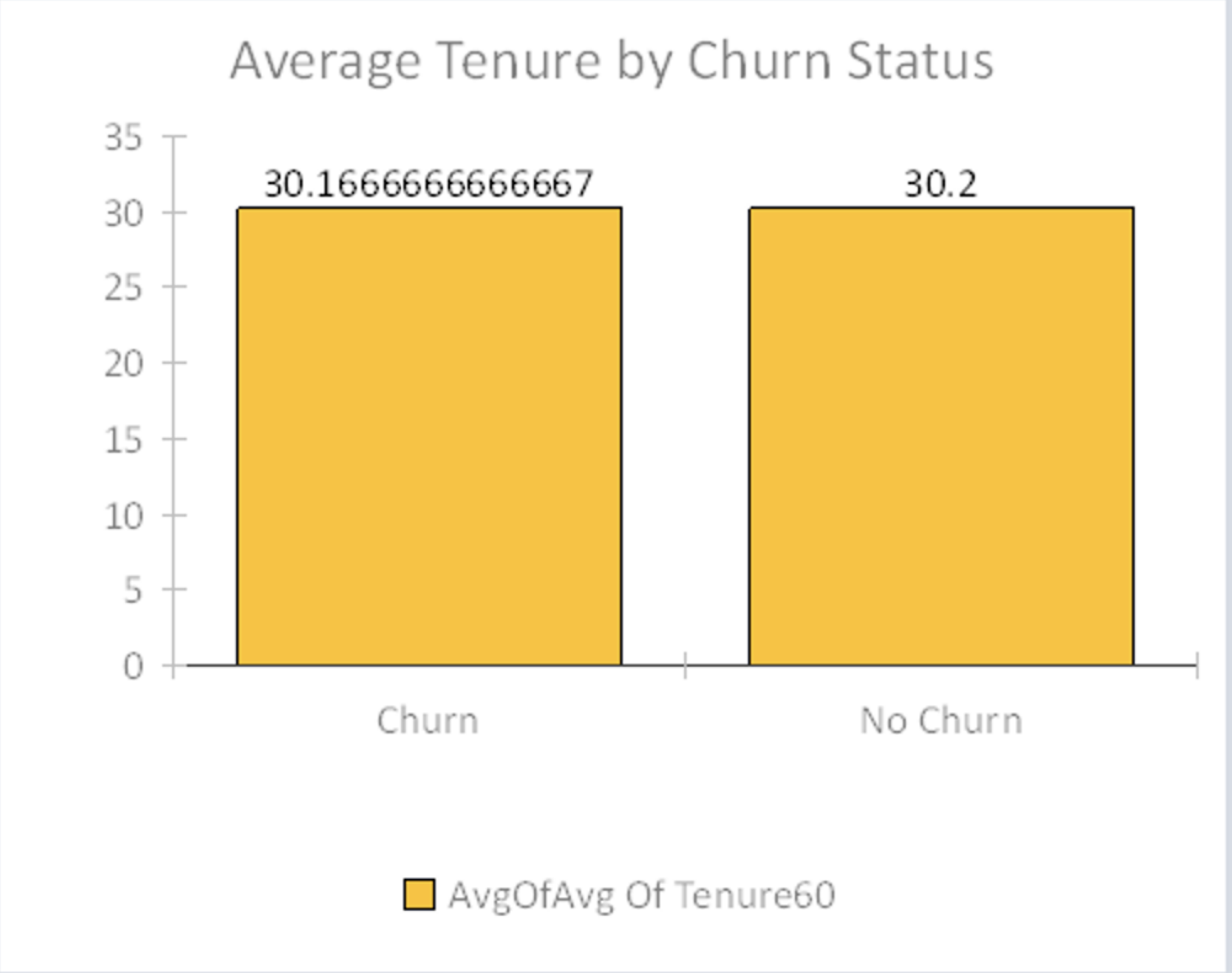


Tenure and Churn Report

Churn	Avg Of Tenure60	AccountId	Count
Churn	32	943735	1
	32	444444	1
	31	961657	1
	31	445556	1
	28	955920	1
	27	957852	1
No Churn	33	881573	1
	32	968126	1
	31	950015	1
	28	959860	1
	27	933619	1

Saturday, March 21, 2020

Page 1 of 1


Average Tenure by Churn Status
35
30
25
20
15
10
5
0
30.1666666666667
30.2
Churn
No Churn
AvgOfAvg Of Tenure60

```
/*----------------------------------------------------------------------------------------------- FULL REPEATABLE SCRIPT -------------------------------------------------------------------------------------------------*/




/*------------------------------------------------------------ DROPPING STORED PROCEDURES ----------------------------------------------------------*/
-- Drop Stored Procedure AddUnitEconomics
IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddUnitEconomics' AND type = 'P')
BEGIN
DROP PROCEDURE AddUnitEconomics
END
GO

-- Drop Stored Procedure AddAdjustments
IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddAdjustments' AND type = 'P')
BEGIN
DROP PROCEDURE AddAdjustments
END
GO

-- Drop Stored Procedure AddSubsidy
IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddSubsidy' AND type = 'P')
BEGIN
DROP PROCEDURE AddSubsidy
END
GO

-- Drop Stored Procedure AddCustomerProfile
IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddCustomerProfile' AND type = 'P')
BEGIN
DROP PROCEDURE AddCustomerProfile
END
GO

-- Drop Stored Procedure AddSpecialDeals
IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddSpecialDeals' AND type = 'P')
BEGIN
DROP PROCEDURE AddSpecialDeals
END
GO
```

```sql
-- Drop Stored Procedure AddRequest
IF EXISTS (SELECT * FROM sysobjects WHERE name = 'AddRequest' AND type = 'P')
BEGIN
DROP PROCEDURE AddRequest
END
GO

/*--------------------------------------------------------------- DROPPING VIEWS --------------------------------------------------------------------*/


--Drop AvgTenureNoChurn View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AvgTenureNoChurn')
BEGIN
DROP VIEW AvgTenureNoChurn
END
GO

--Drop AvgTenureChurn View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AvgTenureChurn')
BEGIN
DROP VIEW AvgTenureChurn
END
GO

--Drop ChurnSubsidy View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'ChurnSubsidy')
BEGIN
DROP VIEW ChurnSubsidy
END
GO

--Drop ChurnAdjustment View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'ChurnAdjustment')
BEGIN
DROP VIEW ChurnAdjustment
END
GO

--Drop NetCLVNoChurn View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'NetCLVNoChurn')
BEGIN
DROP VIEW NetCLVNoChurn
END
```

```
GO


--Drop NetCLVChurn View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'NetCLVChurn')
BEGIN
DROP VIEW NetCLVChurn
END
GO

--Drop AccountsChurned View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'ChurnSubsidy')
BEGIN
DROP VIEW ChurnSubsidy
END
GO

--Drop AccountsChurned View
IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AccountsChurned')
BEGIN
DROP VIEW AccountsChurned
END
GO

-- Drop AccountsNoChurn View

IF EXISTS (SELECT * FROM information_schema.tables WHERE table_type = 'VIEW' AND TABLE_NAME = 'AccountsNoChurn')
BEGIN
DROP VIEW AccountsNoChurn
END
GO

/*------------------------------------------------------------------- DROPPING TABLES ----------------------------------------------------------------------*/


-- Dropping Table Request
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Request')
BEGIN
DROP TABLE Request
END
GO

-- Dropping Table SpecialDeals
```

```
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'SpecialDeals')
BEGIN
DROP TABLE SpecialDeals
END
GO


-- Dropping Table Salesforce
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Salesforce')
BEGIN
DROP TABLE Salesforce
END
GO


-- Dropping Table CustomerProfile
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'CustomerProfile')
BEGIN
DROP TABLE CustomerProfile
END
GO


-- Dropping Table Subsidy
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Subsidy')
BEGIN
DROP TABLE Subsidy
END
GO

-- Dropping Table Adjustments
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'Adjustments')
BEGIN
DROP TABLE Adjustments
END
GO

-- Dropping Table UnitEconomics
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'UnitEconomics')
BEGIN
DROP TABLE UnitEconomics
END
GO
```

```sql
-- Dropping Table hist_account
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'hist_account')
BEGIN
DROP TABLE hist_account
END
GO


/*------------------------------------------- CREATING TABLES, INSERTING DATA, CREATING STORED PROCEDURES ------------------------------------------*/

-- Creating the hist_account Table
CREATE TABLE hist_account (
-- Columns for the hist_account Table
AccountId char(6) not null,
Churn varchar(10) not null,
-- Constraints on the hist_account Table
CONSTRAINT PK_hist_account PRIMARY KEY (AccountID),
CONSTRAINT U1_hist_account UNIQUE(AccountID)
)
-- End Creating the hist_account Table

-- Insert Values into the hist_account Table

INSERT INTO hist_account(AccountId, Churn)
VALUES
('445556','Churn'),
('957852','Churn'),
('933619','No Churn'),
('968126','No Churn'),
('881573','No Churn'),
('959860','No Churn'),
('950015','No Churn'),
('943735','Churn'),
('961657','Churn'),
('955920','Churn')

-- View all the values in the hist_account Table

SELECT * FROM hist_account
```

```
-- Creating the UnitEconomics Table
CREATE TABLE UnitEconomics (
-- Columns for the UnitEconomics Table
AccountId char(6) not null,
GBUsage int not null,
CLV int not null,
CPGA int not null,
Margin int not null,
NetCLV int not null,
Tenure60 int not null,
ARPU int not null,
CCPU int not null,
UpgradeExpense int not null,
-- Constraints on the UnitEconomics Table
CONSTRAINT PK_UnitEconomics PRIMARY KEY(AccountId),
CONSTRAINT U1_UnitEconomics UNIQUE(AccountId),
CONSTRAINT FK1_UnitEconomics FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the UnitEconomics Table

-- Insert values into UnitEconomics Table

INSERT INTO UnitEconomics(AccountId,GBUsage,CLV,CPGA,Margin,NetCLV,Tenure60,ARPU,CCPU,UpgradeExpense)
VALUES
('445556',2,581,2,19,579,31,20,2,5),
('957852',2,477,2,19,475,27,19,2,5),
('933619',1,258,1,10,257,27,11,2,43),
('968126',0,504,17,16,487,32,17,2,1),
('881573',0,1286,7,40,1278,33,39,2,5),
('959860',2,890,0,33,890,28,32,3,24),
('950015',1,785,0,26,784,31,26,3,17),
('943735',4,1437,4,47,1433,32,45,3,26),
('961657',1,529,12,18,517,31,19,3,2),
('955920',2,974,1,37,973,28,38,5,16)

/*Creating a procedure for UnitEconomics. We can only add data into the table if the AccountID specified by the user
is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
If it is, the user cannot add data to the UnitEconomics Table.
*/
GO
CREATE PROCEDURE AddUnitEconomics(
@AccountId char(6),
@GBUsage int,
@CLV int,
@CPGA int,
@Margin int,
@NetCLV int,
@Tenure60 int,
@ARPU int,
@CCPU int,
@UpgradeExpense int)
AS
BEGIN

-- We need the AccountID from the hist_account table
-- First, declare a variable to hold the ID
DECLARE @GetAccountID int

-- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
SELECT @GetAccountID = AccountId FROM hist_account
WHERE hist_account.AccountId IS NOT NULL
AND hist_account.AccountId = @AccountId

--Now we can add the row using an insert statement
INSERT INTO UnitEconomics(AccountId,GBUsage,CLV,CPGA,Margin,NetCLV,Tenure60,ARPU,CCPU,UpgradeExpense)
VALUES (@GetAccountID,@GBUsage, @CLV, @CPGA, @Margin, @NetCLV, @Tenure60, @ARPU, @CCPU,@UpgradeExpense)

--Now return the @@identity so the calling code knows where
-- the data ended up
RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

-- View all the values in the UnitEconomics Table

SELECT * FROM UnitEconomics





-- Creating the Adjustments Table
CREATE TABLE Adjustments (
```

```sql
-- Columns for the Adjustments Table
AccountId char(6) not null,
AdjustmentDate datetime not null default GetDate(),
AdjustmentReason varchar(10) not null,
AdjustmentTotal int not null,
-- Constraints on the Adjustments Table
CONSTRAINT PK_Adjustments PRIMARY KEY(AccountId),
CONSTRAINT U1_Adjustments UNIQUE(AccountId),
CONSTRAINT FK1_Adjustments FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the Adjustments Table

-- Insert values into Adjustments Table

INSERT INTO Adjustments(AccountId,AdjustmentDate,AdjustmentReason,AdjustmentTotal)
VALUES
('445556','2020-02-15','EEDG3', 11),
('957852','2020-02-27','B2BGOV',123),
('933619','2020-02-23','BAD20P',237),
('968126','2020-02-26','GSAPP',6865),
('881573','2020-02-23','GSAVAD',545),
('959860','2020-02-23','GSAVAD',488),
('950015','2020-02-23','GSAVAD',312),
('943735','2020-02-29','GBEQO',2163),
('961657','2020-02-25','GSAVAD',93),
('955920','2020-02-25','GBEQO',2974)

/*Creating a procedure for Adjustments. We can only add data into the table if the AccountID specified by the user
is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
If it is, the user cannot add data to the Adjustments Table.
*/
GO
CREATE PROCEDURE AddAdjustments(
@AccountId char(6),
@AdjustmentDate datetime,
@AdjustmentReason varchar(10),
@AdjustmentTotal int)
AS
BEGIN

-- We need the AccountID from the hist_account table
-- First, declare a variable to hold the ID
DECLARE @GetAccountID int
```

```
-- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
SELECT @GetAccountID = AccountId FROM hist_account
WHERE hist_account.AccountId IS NOT NULL
AND hist_account.AccountId = @AccountId

--Now we can add the row using an insert statement
INSERT INTO Adjustments(AccountId,AdjustmentDate,AdjustmentReason,AdjustmentTotal)
VALUES (@GetAccountID,@AdjustmentDate,@AdjustmentReason,@AdjustmentTotal)

--Now return the @@identity so the calling code knows where
-- the data ended up
RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

--View all the values in the Adjustments Table

SELECT * FROM Adjustments
```

```
-- Creating the Subsidy Table
CREATE TABLE Subsidy (
-- Columns for the Subsidy Table
AccountId char(6) not null,
TransactionDate datetime not null default GetDate(),
SkuModel varchar(100) not null,
UpgradeStatus varchar(50) not null,
UnitCostAmount int not null,
DeviceMargin int not null,
TotalUnits int not null,
-- Constraints on the Subsidy Table
CONSTRAINT PK_Subsidy PRIMARY KEY(AccountId),
CONSTRAINT U1_Subsidy UNIQUE(AccountId),
CONSTRAINT FK1_Subsidy FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the Subsidy Table
```

```sql
-- Insert values into Subsidy Table

INSERT INTO Subsidy(AccountId,TransactionDate,SkuModel,UpgradeStatus,UnitCostAmount,DeviceMargin,TotalUnits)
VALUES
('445556','2020-03-09','FRA T9 MOBILE HOTSPOT','No Upgrade',56,-56,2),
('957852','2020-02-06','FRA T9 MOBILE HOTSPOT','No Upgrade',56,-56,9),
('933619','2020-02-12','ACCESSORIES','No Upgrade',32,-3,1),
('968126','2020-03-04','FRA T9 MOBILE HOTSPOT','No Upgrade',56,-56,2),
('881573','2020-02-06','APL IPHONE 8 V2 64G','No Upgrade',467,-467,5),
('959860','2020-02-10','APL IPHONE 8 V2 64G','No Upgrade',467,-467,1),
('950015','2020-01-31','ALC MW41TM LINKZONE PINE HTSPT','No Upgrade',52,-52,2),
('943735','2020-03-10','APL IPHONE 11 64G','No Upgrade',727,727,19),
('961657','2020-02-19','APL IPAD 7TH GEN 128G','No Upgrade',524,-4,1),
('955920','2020-03-09','ACCESSORIES','No Upgrade',205,-10,1)

/*Creating a procedure for Subsidy Table. We can only add data into the table if the AccountID specified by the user
is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
If it is, the user cannot add data to the Subsidy Table.
*/
GO
CREATE PROCEDURE AddSubsidy(
@AccountId char(6),
@TransactionDate datetime,
@SkuModel varchar(100),
@UpgradeStatus varchar(50),
@UnitCostAmount int,
@DeviceMargin int,
@TotalUnits int)
AS
BEGIN

-- We need the AccountID from the hist_account table
-- First, declare a variable to hold the ID
DECLARE @GetAccountID int

-- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
SELECT @GetAccountID = AccountId FROM hist_account
WHERE hist_account.AccountId IS NOT NULL
AND hist_account.AccountId = @AccountId

--Now we can add the row using an insert statement
INSERT INTO Subsidy(AccountId,TransactionDate,SkuModel,UpgradeStatus,UnitCostAmount,DeviceMargin,TotalUnits)
```

```
VALUES (@GetAccountID,@TransactionDate,@SkuModel,@UpgradeStatus,@UnitCostAmount,@DeviceMargin,@TotalUnits)

--Now return the @@identity so the calling code knows where
-- the data ended up
RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

-- View all the values in the Subsidy Table

SELECT * FROM Subsidy





-- Creating the CustomerProfile Table
CREATE TABLE CustomerProfile (
-- Columns for the CustomerProfile Table
AccountId char(6) not null,
CustomerName varchar(200) not null,
AccountType char(1) not null,
AccountSubType char(1) not null,
CreditClass char(1) not null,
ProfileRegion varchar(100) not null,
Segment varchar(50) not null,
SegmentGroup char(11) not null,
-- Constraints on the CustomerProfile Table
CONSTRAINT PK_CustomerProfile PRIMARY KEY(AccountId),
CONSTRAINT U1_CustomerProfile UNIQUE(AccountId),
CONSTRAINT U2_CustomerProfile UNIQUE(CustomerName),
CONSTRAINT FK1_CustomerProfile FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the CustomerProfile Table

-- Insert values into CustomerProfile Table

INSERT INTO CustomerProfile(AccountId,CustomerName,AccountType,AccountSubType,CreditClass,ProfileRegion,Segment,SegmentGroup)
VALUES
('445556','STATE OF INGTON DEPARTMENT OF ENTERP SERVICES','G','F','G','Public Sector Sales','FedGov','MajorPublic'),
('957852','CITY OF T CHAR','G','F','G','Public Sector Sales','State Local','MajorPublic'),
('933619','WASHI UNIF SCH DICT','G','F','G','Southwest State and Local','State Local','MajorPublic'),
('968126','Hig Commu Char and Tec Sch','G','F','2','Southwest State and Local','State Local','MajorPublic'),
('881573','JEFF CENTER OF MEN HEALTH','G','F','G','Southwest State and Local','State Local','MajorPublic'),
('959860','SOL UNI SCHOOL DISTRICT','G','F','G','Southwest State and Local','State Local','MajorPublic'),
('950015','IS HIMALIA ACADEMY','G','F','G','Southwest State and Local','State Local','MajorPublic'),
('943735','UNIV COMPAN INC','G','F','G','Northeast State and Local','State Local','MajorPublic'),
('961657','THE NEW BENEFIT COUNSELING CENTER INC','G','F','2','Northeast State and Local','State Local','MajorPublic'),
('955920','UNLV SIMPLE MACHINE AND AI','G','F','G','Southwest State and Local','State Local','MajorPublic')

/*Creating a procedure for CustomerProfile Table. We can only add data into the table if the AccountID specified by the user
is equal to the same AccountID in the hist_account table. Also, the AccountID in the hist_account table cannot be null.
If it is, the user cannot add data to the CustomerProfile Table.
*/
GO
CREATE PROCEDURE AddCustomerProfile(
@AccountId char(6),
@CustomerName varchar(200),
@AccountType char(1),
@AccountSubType char(1),
@CreditClass char(1),
@ProfileRegion varchar(100),
@Segment varchar(50),
@SegmentGroup char(11))
AS
BEGIN

-- We need the AccountID from the hist_account table
-- First, declare a variable to hold the ID
DECLARE @GetAccountID int

-- Get the AccountID from the hist_account table from the AccountID provided and store it in @GetAccountID
SELECT @GetAccountID = AccountId FROM hist_account
WHERE hist_account.AccountId IS NOT NULL
AND hist_account.AccountId = @AccountId

--Now we can add the row using an insert statement
INSERT INTO CustomerProfile(AccountId,CustomerName,AccountType,AccountSubType,CreditClass,ProfileRegion,Segment,SegmentGroup)
VALUES (@GetAccountID,@CustomerName,@AccountType,@AccountSubType,@CreditClass,@ProfileRegion,@Segment,@SegmentGroup)

--Now return the @@identity so the calling code knows where
-- the data ended up
RETURN SCOPE_IDENTITY()
END
```

```
GO

--End of creating the stored procedure

-- View all the values in the CustomerProfile Table

SELECT * FROM CustomerProfile






-- Creating the Salesforce Table
CREATE TABLE Salesforce (
-- Columns for the Salesforce Table
SalesforceAccountName varchar(200),
SalesforceAccountID char(8) not null,
AccountId char(6) not null,
CreditClass varchar(3) not null,
CreditScore varchar(3) not null,
AutoPay varchar(5) not null,
AddALine varchar(5) not null,
EasyPay varchar(5) not null,
-- Constraints on the Salesforce Table
CONSTRAINT PK_Salesforce PRIMARY KEY(SalesforceAccountName),
CONSTRAINT U1_Salesforce UNIQUE(SalesforceAccountName),
CONSTRAINT U2_Salesforce UNIQUE(SalesforceAccountID),
CONSTRAINT U3_Salesforce UNIQUE(AccountId),
CONSTRAINT FK1_Salesforce FOREIGN KEY (AccountId) REFERENCES hist_account(AccountId)
)
-- End Creating the Salesforce Table

-- Insert values into Salesforce Table

INSERT INTO Salesforce(SalesforceAccountName,SalesforceAccountID,AccountId,CreditClass,CreditScore,AutoPay,AddALine,EasyPay)
VALUES
('CITY OF T CHAR','rJT0YAAW','957852','G','0','No','Yes','No'),
('WASHI UNIF SCH DICT','seinyAAA','933619','G','0','No','Yes','No'),
('Hig Commu Char and Tec Sch','e4p5kAAA','968126','2','0','No','Yes','No'),
('JEFF CENTER OF MEN HEALTH','TbgC7AAJ','881573','G ','0','Yes','Yes','Yes'),
('IS HIMALIA ACADEMY','K7OEXAA3','950015','G','0','No','Yes','No'),
('UNLV SIMPLE MACHINE AND AI','dotOLAAY','955920','G','0','No','Yes','No')
```

```sql
-- View all the values in the Salesforce Table

SELECT * FROM Salesforce







-- Creating the SpecialDeals Table
CREATE TABLE SpecialDeals (
-- Columns for the SpecialDeals Table
SpecialDealID char(8) not null,
SalesDivision varchar(30) not null,
OwnerTeam varchar(100) not null,
PotentialLines int  not null,
ApprovalStatus varchar(50) not null,
ActiveLines int not null,
AccountName varchar(200) not null,
-- Constraints on the SpecialDeals Table
CONSTRAINT PK_SpecialDeals PRIMARY KEY(SpecialDealID),
CONSTRAINT U1_SpecialDeals UNIQUE(SpecialDealID),
CONSTRAINT U2_SpecialDeals UNIQUE(AccountName),
CONSTRAINT FK1_SpecialDeals FOREIGN KEY (AccountName) REFERENCES Salesforce(SalesforceAccountName)
)
-- End Creating the SpecialDeals Table

-- Insert values into SpecialDeals Table

INSERT INTO SpecialDeals(SpecialDealID,SalesDivision,OwnerTeam,PotentialLines,ApprovalStatus,ActiveLines,AccountName)
VALUES
('0tI4OQAU','Government','SL Midwest',105,'Approved',248,'CITY OF T CHAR'),
('87JRKQA2','Government','SL Northern California',42,'Approved',115,'WASHI UNIF SCH DICT'),
('0tG44QAE','Government','SL Northern California',525,'Approved',0,'Hig Commu Char and Tec Sch'),
('0tGLQQA2','Government','SL Southwest',315,'Approved',18,'JEFF CENTER OF MEN HEALTH'),
('GMWSwQAP','Government','SL Southern California',105,'Approved',52,'IS HIMALIA ACADEMY'),
('0tFRXQA2','Government','SL Southwest',25,'Approved',630,'UNLV SIMPLE MACHINE AND AI')

/*Creating a procedure for SpecialDeals Table. We can only add data into the table if the Account Name specified by the user
is equal to the same Account Name in the Salesforce table. Also, the Account Name in the Salesforce table cannot be null.
If it is, the user cannot add data to the SpecialDeals Table.
*/
```

```
GO
CREATE PROCEDURE AddSpecialDeals(
@SpecialDealID char(8),
@SalesDivision varchar(30),
@OwnerTeam varchar(100),
@PotentialLines int,
@ApprovalStatus varchar(50),
@ActiveLines int,
@AccountName varchar(200))
AS
BEGIN

-- We need the Account Name from the Salesforce Table
-- First, declare a variable to hold the name
DECLARE @GetAccountName varchar(200)

-- Get the Account Name from the Salesforce Table from the Account Name provided by the user and store it in @GetAccountName
SELECT @GetAccountName = SalesforceAccountName FROM Salesforce
WHERE Salesforce.SalesforceAccountName IS NOT NULL
AND Salesforce.SalesforceAccountName = @AccountName

--Now we can add the row using an insert statement
INSERT INTO SpecialDeals(SpecialDealID,SalesDivision,OwnerTeam,PotentialLines,ApprovalStatus,ActiveLines,AccountName)
VALUES (@SpecialDealID,@SalesDivision,@OwnerTeam,@PotentialLines,@ApprovalStatus,@ActiveLines,@GetAccountName)

--Now return the @@identity so the calling code knows where
-- the data ended up
RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

-- View all the values in the SpecialDeals Table

SELECT * FROM SpecialDeals
```

```
-- Creating the Request Table
```

```sql
CREATE TABLE Request (
-- Columns for the Request Table
RequestID varchar(50) not null,
SpecialDealID char(8) not null,
NonStandardNumber varchar(50) not null,
RequestPerLine int  not null,
TotalCreditValue int not null,
DeviceName varchar(100) not null,
DevicePriceAfterDiscount int not null,
MarketingCampaign varchar(200) not null,
RequestQuantity int not null,
NSSDeal varchar(5) not null,
-- Constraints on the Request Table
CONSTRAINT PK_Request PRIMARY KEY(RequestID),
CONSTRAINT U1_Request UNIQUE(RequestID),
CONSTRAINT U2_Request UNIQUE(SpecialDealID),
CONSTRAINT U3_Request UNIQUE(NonStandardNumber),
CONSTRAINT FK1_Request FOREIGN KEY (SpecialDealID) REFERENCES SpecialDeals(SpecialDealID)
)
-- End Creating the Request Table

-- Insert values into Request Table

INSERT INTO
Request(RequestID,SpecialDealID,NonStandardNumber,RequestPerLine,TotalCreditValue,DeviceName,DevicePriceAfterDiscount,MarketingCampaign,RequestQuantity,NSSDeal)
VALUES
('1FgZ1QAK','0tl4OQAU','Request-7764',25,0,'LG 300',0,'Go get it',0,'Yes'),
('10B01QAE','87JRKQA2','Request-2051',0,0,'LG 400',175,'Get everything free',52,'Yes'),
('1FemmQAC','0tG44QAE','Request-6493',0,0,'Coolpad SURF PRO',0,'Go get it',525,'Yes'),
('1Ff39QAC','0tGLQQA2','Request-6680',0,0,'Silver GoFlip',0,'Free phone',31,'Yes'),
('0I91dQAA','GMWSwQAP','Request-2734',105,105,'LG 900',0,'Free phone',0,'Yes'),
('1Fe2FQAS','0tFRXQA2','Request-5935',0,0,'Tablet Go 8',0,'Free Tablet device',25,'Yes')


/*Creating a procedure for the Request Table. We can only add data into the Request Table if the Special Deal ID specified by the user
is equal to the same Special ID in the Special Deals table. Also, the Special Deals ID in the Special Deals table cannot be null.
If it is, the user cannot add data to the Request Table.
*/
GO
CREATE PROCEDURE AddRequest(
@RequestID varchar(50),
@SpecialDealID char(8),
@NonStandardNumber varchar(50),
@RequestPerLine int,
@TotalCreditValue int,
@DeviceName varchar(100),
@DevicePriceAfterDiscount int,
@MarketingCampaign varchar(200),
@RequestQuantity int,
@NSSDeal varchar(5))
AS
BEGIN

-- We need the SpecialDealID from the Special Deals Table
-- First, declare a variable to hold the ID
DECLARE @GetAccountID int

-- Get the SpecialDealID from the Special Deals  Table from the ID provided by the user and store it in @GetAccountID
SELECT @GetAccountID = SpecialDealID FROM SpecialDeals
WHERE SpecialDeals.SpecialDealID IS NOT NULL
AND SpecialDeals.SpecialDealID = @SpecialDealID

--Now we can add the row using an insert statement
INSERT INTO
Request(RequestID,SpecialDealID,NonStandardNumber,RequestPerLine,TotalCreditValue,DeviceName,DevicePriceAfterDiscount,MarketingCampaign,RequestQuantity,NSSDeal)
VALUES
(@RequestID,@GetAccountID,@NonStandardNumber,@RequestPerLine,@TotalCreditValue,@DeviceName,@DevicePriceAfterDiscount,@MarketingCampaign,@RequestQuantity,
@NSSDeal)

--Now return the @@identity so the calling code knows where
-- the data ended up
RETURN SCOPE_IDENTITY()
END
GO

--End of creating the stored procedure

-- View all the values in the Request Table

SELECT * FROM Request

/*--------------------------------------------------------------- DATA QUESTIONS --------------------------------------------------------------------*/

/* Data Question 1

1)  What percent of accounts churn?
```

```
*/

GO

-- Create a view of all accounts that Churned

CREATE VIEW AccountsChurned AS (
SELECT
hist_account.AccountId
FROM hist_account
WHERE hist_account.Churn = 'Churn'
)


GO

-- Create a view of all accounts that did not churn

CREATE VIEW AccountsNoChurn AS (
SELECT
hist_account.AccountId
FROM hist_account
WHERE hist_account.Churn = 'No Churn'
)

SELECT * FROM AccountsNoChurn
SELECT * FROM AccountsChurned

GO

-- Query to get the percent of the accounts that churned

SELECT
CAST(CAST(a.Total_Churned AS decimal (12,4)) /  CAST(a.Total_Accounts AS decimal (12,4)) AS decimal(12,2))*100 as Percent_Churned_Data_Question_1
FROM
(
SELECT
(SELECT COUNT(*) FROM AccountsChurned) as Total_Churned,
COUNT(DISTINCT hist_account.AccountId) as Total_Accounts
FROM hist_account
) a
```

```
GO


/* Data Question 1 Answer

1)  50.00%

*/


/* Data Question 2

2)  Of those accounts that churn, what is the average subsidy?

*/

--Query to get the average subsidy for all the accounts that churned

SELECT
CAST(AVG(Subsidy.DeviceMargin) as decimal (12,2)) as Average_Subsidy_Data_Question_2
FROM Subsidy
WHERE Subsidy.AccountId IN (SELECT AccountId FROM AccountsChurned)
-- Check

GO

--Creating a view of those accounts that churned and took a subsidy
CREATE VIEW ChurnSubsidy AS (
SELECT
b.AccountId,
b.Churn,
DeviceMargin as Subsidy
FROM Subsidy a
INNER JOIN hist_account b on a.AccountId = b.AccountId

)

GO

--Query to confirm the average susbidy amount for those accounts that churned

SELECT
CAST(AVG(a.Subsidy) as decimal (12,2)) as Average_Subsidy_Check
```

```sql
FROM
(
SELECT
AccountId,
Churn,
Subsidy
FROM ChurnSubsidy
WHERE Churn = 'Churn'
-- 5 accounts that churn with a subsidy
) a

-- Check complete. $120.00

/* Data Question 2 Answer

2)  $120.00

*/

/* Data Question 3

3)  Of those accounts that churn, what is the average credit received?

*/

-- Query to get the average credit recieved across the accounts that churned

SELECT
CAST(AVG(Adjustments.AdjustmentTotal) as decimal (12,2)) as Average_Credit_Data_Question_3
FROM Adjustments
WHERE Adjustments.AccountId IN (SELECT AccountId FROM AccountsChurned)
-- Check

GO

-- Creating a view that selects all the accounts that churned and received a credit

CREATE VIEW ChurnAdjustment AS (
SELECT
b.AccountId,
b.Churn,
a.AdjustmentTotal as Credit
FROM Adjustments a
```

```
INNER JOIN hist_account b on a.AccountId = b.AccountId
)

GO

-- Performing a query to validate the intial query in terms of the average credit

SELECT
CAST(AVG(a.Credit) as decimal (12,2)) as Average_Credit_Check
FROM
(
SELECT
AccountId,
Churn,
Credit
FROM ChurnAdjustment
WHERE Churn = 'Churn'
-- 5 accounts that churn with a credit
) a

-- Check complete. $1,072

SELECT * FROM ChurnAdjustment

/* Data Question 3 Answer

3)  $1,072

*/

/* Data Question 4

4) Of those accounts that churn, what percent of them are NSS Deals?

*/

-- Query to select those accounts that churn and of those accounts that churned, what percent of them were designated as an NSS Deal?
SELECT
CAST(CAST(COUNT(a.NSSDeal) as decimal (12,2)) /  CAST(COUNT(a.AccountId) as decimal (12,2)) as decimal (12,2))*100 as percent_of_churn_NSSDeal_Data_Question_4
FROM
(SELECT
a.AccountId,
b.SalesforceAccountID,
b.SalesforceAccountName,
c.SalesDivision,
c.OwnerTeam,
c.ApprovalStatus,
c.SpecialDealID,
d.RequestID,
d.NonStandardNumber,
d.DeviceName,
d.NSSDeal
FROM AccountsChurned a
LEFT OUTER JOIN Salesforce b on b.AccountId = a.AccountId
LEFT OUTER JOIN SpecialDeals c on c.AccountName = b.SalesforceAccountName
LEFT OUTER JOIN Request d on d.SpecialDealID = c.SpecialDealID
) a

/* Data Question 4 Answer

4)  40.00%

*/

/* Data Question 5

5) What is the difference in the average Net CLV of those accounts that churn from the average NET CLV of those accounts that do not churn?

*/


GO

-- Creating a view to retreive the average NET CLV of those accounts that churn
CREATE VIEW NetCLVChurn AS (
SELECT
AVG(a.NetCLV) as Average_CLV
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'Churn'

)

SELECT * FROM NetCLVChurn

GO
```

```
-- Creating a view to retreive the average NET CLV of those accounts that do not churn
CREATE VIEW NetCLVNoChurn AS (
SELECT
AVG(a.NetCLV) as Average_CLV
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'No Churn'

)

GO

-- Query to get the difference in Net CLVs (accounts that churn vs. accounts that do not churn)
SELECT
SUM(a.Average_CLV) as Difference_Avg_CLV_Data_Question_5
FROM
(
SELECT
Average_CLV as Average_CLV
FROM NetCLVChurn
UNION
SELECT
Average_CLV*-1 as Average_CLV
FROM NetCLVNoChurn
) a

/* Data Question 5 Answer

5)  56

*/

/* Data Question 6

6)  What is the difference in the average Tenure of those accounts that churn from the average Tenure of those accounts that do not churn?

*/

GO

--Creating a view to retreive the average tenure of those accounts that churn
CREATE VIEW AvgTenureChurn AS (
SELECT
AVG(a.Tenure60) as Average_Tenure
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'Churn'

)

GO

--Creating a view to retreive the average tenure of those accounts that do not churn
CREATE VIEW AvgTenureNoChurn AS (
SELECT
AVG(a.Tenure60) as Average_Tenure
FROM UnitEconomics a
INNER JOIN hist_account b on b.AccountId = a.AccountId
WHERE b.Churn = 'No Churn'

)

GO

--Query to determine the difference in average tenure between those accounts that churn and those accounts that do not churn
SELECT
SUM(a.Average_Tenure) as Difference_Avg_Tenure_Data_Question_6
FROM
(
SELECT
Average_Tenure as Average_Tenure
FROM AvgTenureChurn
UNION
SELECT
Average_Tenure*-1 as Average_Tenure
FROM AvgTenureNoChurn
) a

GO

-- Average tenure is lower by 1 momth for those accounts that churn

SELECT * FROM AvgTenureChurn
SELECT * FROM AvgTenureNoChurn
```

```sql
/* Data Question 6 Answer

6)  Average tenure is lower by 1 momth for those accounts that churn

*/


/*---------------------------------------------------------------- Query witH New Data --------------------------------------------------------------*/

-- Query to get all selected information regarding the new account that was added through microsoft access
SELECT
a.AccountId,
a.Churn,
b.ARPU,
b.CCPU,
b.Margin,
b.UpgradeExpense,
c.AdjustmentDate,
c.AdjustmentReason,
c.AdjustmentTotal,
d.DeviceMargin,
d.SkuModel,
e.CustomerName,
e.Segment,
f.AutoPay,
f.SalesforceAccountName,
f.AddALine,
g.PotentialLines,
g.SpecialDealID,
g.SalesDivision,
h.NSSDeal,
h.MarketingCampaign,
h.RequestQuantity,
h.RequestID
FROM hist_account a
INNER JOIN UnitEconomics b on b.AccountId = a.AccountId
INNER JOIN Adjustments c on c.AccountId = a.AccountId
INNER JOIN Subsidy d on d.AccountId = a.AccountId
INNER JOIN CustomerProfile e on e.AccountId = a.AccountId
LEFT OUTER JOIN Salesforce f on f.AccountId = a.AccountId
LEFT OUTER JOIN SpecialDeals g on g.AccountName = f.SalesforceAccountName
LEFT OUTER JOIN Request h on h.SpecialDealID = g.SpecialDealID
WHERE a.AccountId = '444444'
```

Database Design & Management with Microsoft Access and Microsoft SQL Server Reflection

-- This is new account I added through Microsoft Access

3/21/2020

***Note: All Data has been edited, randomized, or made up for the Security of the firm**

This project is a critical first step in establishing the best framework for management to answer critical data questions regarding special approvals. Also, this database helps resolve the issue of disparate resources and the lack of interconnectivity between different databases. Having data in Salesforce and Teradata is a risk in itself. It is even riskier trying to manually join these datasets to answer highly critical and sensitive data questions for senior management. The databases have different logical designs, different security protocols, different stored procedures, and some data is manually entered without any rule, esp. in Salesforce. Salesforce lacks security and logical design because any sales rep can enter whatever data he or she wants to. This is one of the reasons it is impossible to join Teradata and Salesforce together. This database unifies both databases with the same overall logical and conceptual designs. Both databases will now have stored procedures to ensure accurate data is entered into the database. Finally, as witnessed above, this database is able to answer very important questions for management with ONE query. A member of the team does not have to go query Teradata, then query Salesforce, then try to figure out how to unite the results to accurately answer the data questions.

Lucas Daniel Zarzeczny
T-MOBILE