Sublime Customer Relations

Parsa Bazargani

Summary/Table of Contents:

Sublime Customer Relations Management System

1. Introduction

Basic functions of Sublime CRM

1. Customer data management
2. Lead and opportunity management
3. Sales Forecasting

2. Business Rules

- A pre-built E2E CRM solution to give your business a head start
- Customize your marketing campaigns
- Customer lifecycle funnel tracking
- Voice of the customer

Business Background

Founded in Northern California near Silicon Valley.

Focused on creating an easy experience for both the customer and the company via several simple methods.

Created by individuals whose past informed them there was a need for improvement in the industry.

CRM system is now coveted and sought after by many companies throughout California.

Business Rules

1. Customers must have a payment type linked to their account
2. Customer can submit support tickets
3. Companies can create leads and campaigns at ease.
4. Companies can list items to sell

Business Requirements

For this project, we will focus on 2 types of users:

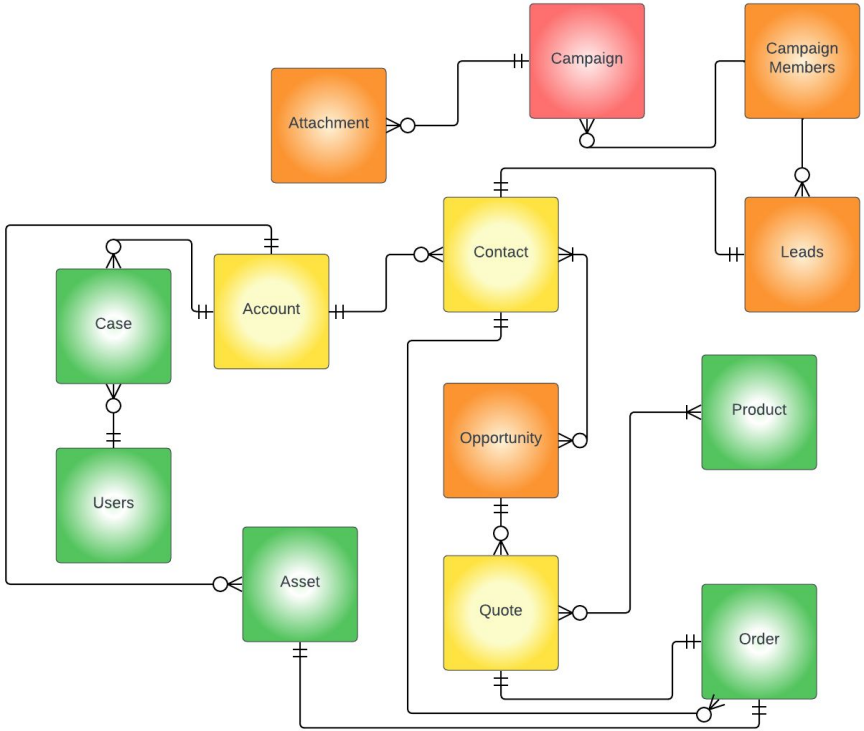
Buyers - Can purchase items through ‘orders’ with their credit card, and update their credit card at any time.

Businesses - Create campaigns for products, listing assets for a certain price.

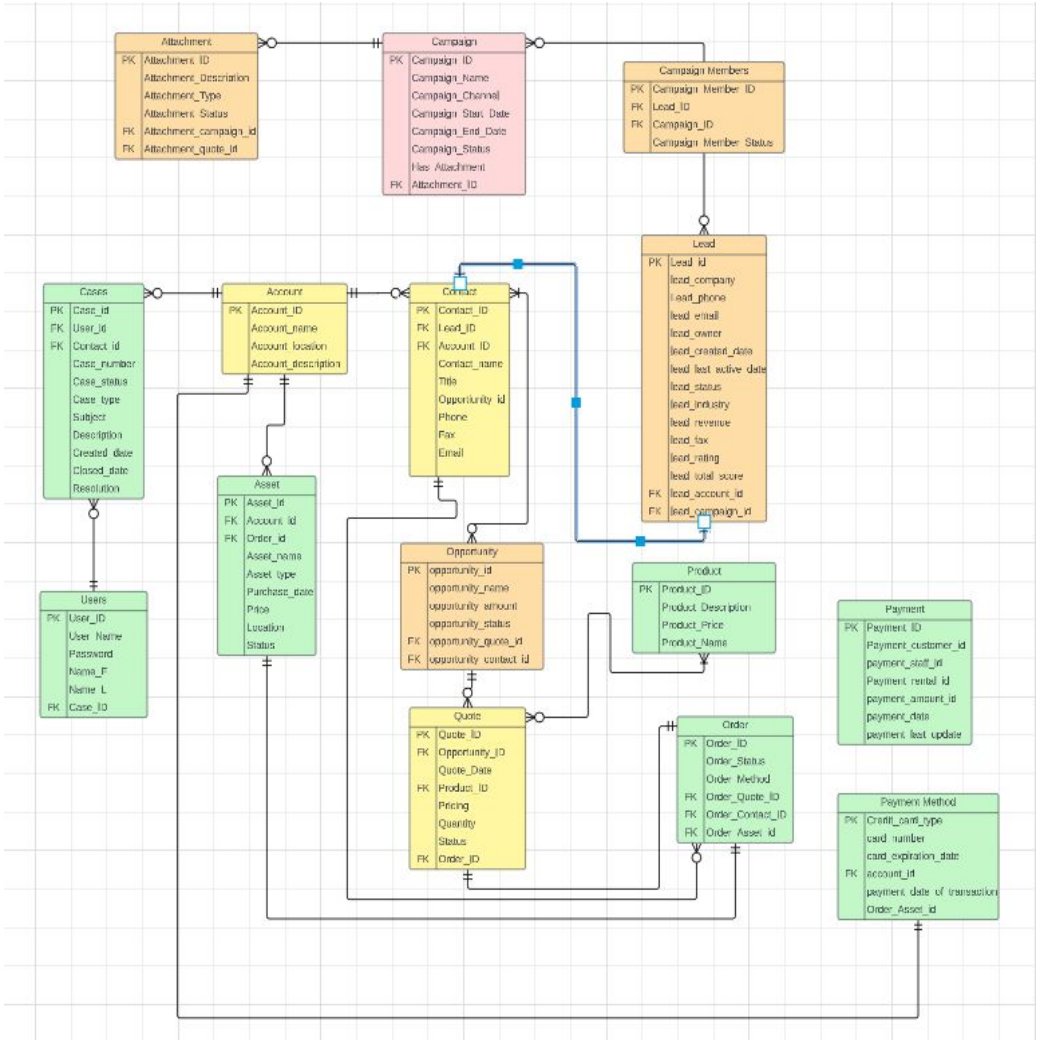
3. Operation Procedures

1. Customer Security
   - 1.1. Stay up to date with data protection regulations, such as CCPA (California Consumer Privacy Act)
   - 1.2. Implement security measures to exclude hackers, cyber-terrorists
2. System Maintenance, Keeping it up to the standard
   - 2.1. Ensure the CRM systems we provide are working at the highest level
   - 2.2. Testing new methods, features, attributes prior to sending to clients
3. Thorough Employee Training
   - 3.1. Our employees become experts in CRM
     - 3.1.1. Offering nonstop guidance and resources to customers
   - 3.2. Clients can contact our company 24/7, responses in less than 30 minutes

4. ER Conceptual Model- ER Model

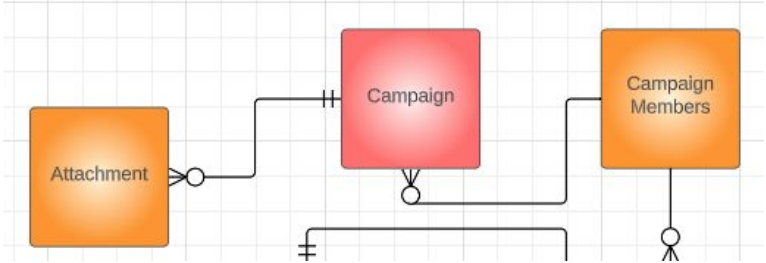


Continued ER Conceptual Model- Full ERD Detailed



Campaign Table/Implementation

```
CREATE TABLE [dbo].[Campaign](
    [campaign_id] [int] NOT NULL,
    [campaign_name] [varchar](25) NOT NULL,
    [campaign_channel] [varchar](20) NOT NULL,
    [campaign_start_date] [date] NOT NULL,
    [campaign_end_date] [date] NOT NULL,
    [campaign_status] [varchar](25) NOT NULL,
    [has_attachment] [char](1) NOT NULL,
    [attachment_id] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [campaign_id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Campaign]  WITH CHECK ADD  CONSTRAINT [attachment_id] FOREIGN KEY([attachment_id])
REFERENCES [dbo].[Attachment] ([attachment_id])
GO

ALTER TABLE [dbo].[Campaign] CHECK CONSTRAINT [attachment_id]
GO
```



	campaign_id	campaign_name	campaign_channel	campaign_start_date	campaign_end_date	campaign_status	has_attachment	attachment_id
1	1	online winter sale	email	2023-01-01	2023-03-01	inactive	y	1
2	2	online spring sale	email	2023-04-01	2023-06-01	inactive	y	2
3	3	online spring sale	mail	2023-06-01	2023-08-01	active	y	3
4	4	online summer sale 1	email	2023-07-01	2023-08-01	active	y	4
5	5	online summer sale 2	email	2023-07-01	2023-08-01	active	y	5
6	6	online summer sale 3	email	2023-07-01	2023-08-01	active	y	6
7	7	online summer sale 4	email	2023-07-01	2023-08-01	active	y	7
8	8	summer sale 1	mail	2023-07-01	2023-08-01	active	y	8
9	9	summer sale 2	mail	2023-07-01	2023-08-01	active	y	9
10	10	summer sale 3	mail	2023-07-01	2023-08-01	active	y	10

Views 1 - What's most popular time for campaigns?

```
-- view
select month(campaign_start_date) as campaign_month, count(distinct campaign_id) as num_campaign
from Campaign
group by month(campaign_start_date)
order by 2 desc
```

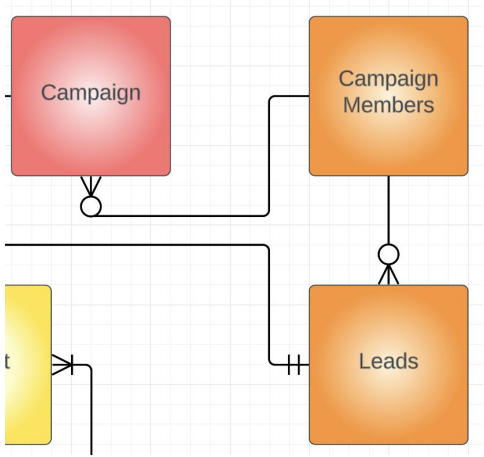
Results | Messages

	campaign_month	num_campaign
1	7	7
2	1	1
3	4	1
4	6	1

Campaign Members' Table and Implementation

```
CREATE TABLE Campaign_Member(
Campaign_member_id Int NOT NULL Primary Key,
Campaign_member_status VARCHAR(50) NOT NULL,
lead_id int NOT NULL Foreign Key,
campaign_id NOT NULL Foreign Key);

ALTER TABLE Campaign_Member ADD FOREIGN KEY
(campaign_id) REFERENCES Campaign (campaign_id);
ALTER TABLE Campaign_Member ADD FOREIGN KEY
(lead_id) REFERENCES Lead (lead_id);
```

Local View

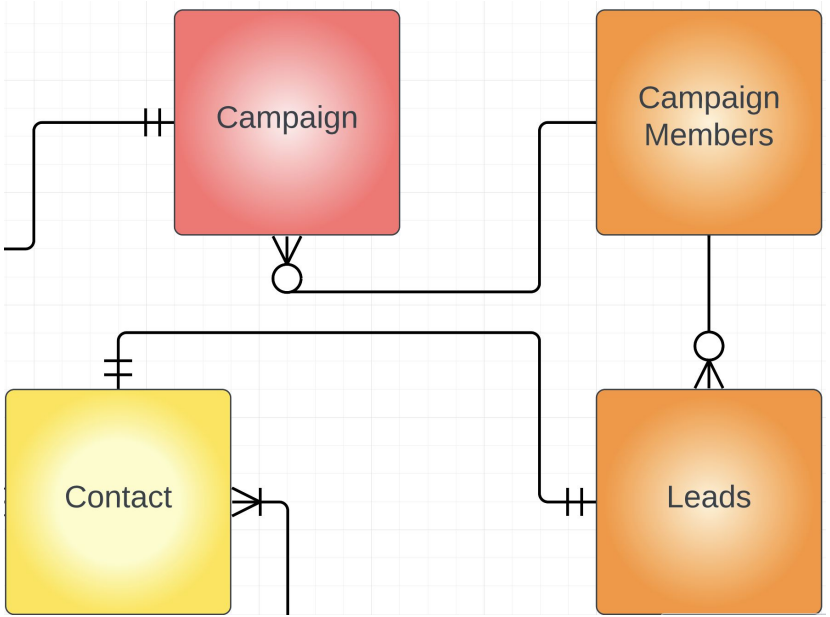


Results | Messages

	Campaign_member_id	Campaign_member_status	Lead_id	Campaign_id
1	1	active	1	1
2	2	active	2	2
3	3	inactive	3	3
4	4	active	4	4

Lead Table and Implementation

```
CREATE TABLE Lead (
lead_id INT NOT NULL PRIMARY KEY,
lead_company VARCHAR(255),
lead_email VARCHAR(25),
lead_phone VARCHAR(50),
lead_owner VARCHAR(255),
lead_created_date DATE ,
lead_last_active_date DATE ,
lead_status VARCHAR(25),
lead_industry VARCHAR(255),
lead_revenue DECIMAL(12,2) ,
lead_fax VARCHAR(25),
lead_rating VARCHAR(255),
lead_total_score INT,

CONSTRAINT CHECK_STATUS CHECK (lead_status LIKE
'Open%' OR lead_status LIKE 'Converted%' OR
lead_status LIKE 'Qualified')
);
```



A local view of Lead

	lead_id	lead_company	lead_email	lead_phone	lead_owner	lead_created_date	lead_last_active_date	lead_status	lead_industry	lead_revenue	lead_fax	lead_rating	lead_total_score
1	31	Johnson Inc.	john.doe@acme.com	555-1234	Jane Smith	2023-05-12	2023-07-01	Qualified	Agriculture	250000.00	-7654	good	9
2	40	Harris Enterprises	harris.jessica@acme…	555-0123	Jessica Roberts	2023-05-20	2023-07-10	Open	Consulting	*NULL*	-7663	good	8
3	45	*NULL*	pqr@example.com	555-5555	John Smith	2023-07-01	2023-07-10	Converted	Finance	500000.00	-7667	good	8
4	14	ABC Corporation	*NULL*	555-4444	John Smith	2022-06-01	2022-07-15	*NULL*	Retail	1000000.00	*NULL*	good	7
5	24	White & Sons	*NULL*	555-3333	Sarah White	2023-12-01	*NULL*	Converted	Healthcare	1500000.00	-7665	good	7
6	29	King Corp.	king.james@acme.com	555-6666	James King	2024-07-01	*NULL*	Open	Technology	4000000.00	*NULL*	good	7

Lead Trigger

status	lead_industry	lead_revenue	lead_fax	lead_rating	lead_total_score
	Agriculture	NULL	NULL	okay	3
erted	NULL	NULL	NULL	okay	3
	NULL	750000.00	NULL	bad	2
	NULL	NULL	NULL	okay	3
	Healthcare	NULL	555-555-5555	okay	3
	Manufacturing	NULL	NULL	okay	3
	NULL	NULL	555-555-5555	okay	3
erted	Hospitality	NULL	555-555-5555	okay	3
	NULL	NULL	NULL	bad	2
ified	Consulting	NULL	NULL	okay	4
ified	NULL	250000.00	555-123-4567	good	6
erted	Technology	NULL	NULL	okay	5
	Finance	NULL	555-555-5555	okay	5
	Retail	1000000.00	NULL	good	7
erted	Healthcare	NULL	NULL	okay	5
	NULL	1500000.00	555-987-6543	okay	4
ified	NULL	1750000.00	555-123-4567	good	6
	Transportation	NULL	555-555-5555	okay	5
erted	NULL	2250000.00	NULL	okay	5
	NULL	2500000.00	555-555-5555	good	6
	Technology	NULL	NULL	okay	5
	Finance	NULL	NULL	okay	5
erted	Retail	NULL	-7664	good	6
erted	Healthcare	1500000.00	-7665	good	7
ified	Manufacturing	2000000.00	NULL	good	6
ified	NULL	2500000.00	NULL	good	6
	NULL	3000000.00	NULL	good	6
erted	NULL	3500000.00	-7666	okay	5

```sql
CREATE TRIGGER CountNullColumnsTrigger
ON Lead
AFTER INSERT
AS
BEGIN
    UPDATE l
    SET l.lead_total_score = (
        SELECT COUNT(lead_company) + COUNT(lead_email)
              + COUNT(lead_phone) +COUNT(lead_owner)
             + COUNT(lead_created_date) + COUNT(lead_last_active_date)
             + COUNT(lead_industry) + COUNT(lead_revenue)
             + COUNT(lead_fax)
        FROM inserted
    )
    FROM Lead l

    UPDATE l
    SET l.lead_rating = (
        SELECT CASE
        WHEN l.lead_total_score BETWEEN 6 AND 9 THEN 'good'
        WHEN l.lead_total_score BETWEEN 3 AND 6 THEN 'okay'
        ELSE 'bad'
    END
    )
    FROM Lead l

    INNER JOIN inserted i ON l.lead_id = i.lead_id
    WHERE l.lead_total_score IS NULL;
END;
```

Lead View: Strongest Candidates

```
CREATE VIEW strong_leads_view AS

SELECT [cis55_50].[dbo].[lead].[lead_rating],
  [cis55_50].[dbo].[lead].[lead_company],
  [cis55_50].[dbo].[lead].[lead_owner],
  [cis55_50].[dbo].[lead].[lead_email]
FROM lead
WHERE lead_last_active_date >= DATEADD(month, -3, GETDATE())
WHERE lead_status = 'Open'
END;
```

lead_rating	lead_company	lead_owner	lead_email
good	Harris Enterprises	Jessica Roberts	harris.jessica@acme.com
good	XYZ Corporation	John Doe	xyz@example.com
good	*NULL*	John Doe	smith.john@acme.com
okay	Walker Industries	*NULL*	walker.jill@acme.com
okay	PQR Corporation	*NULL*	*NULL*
okay	LMN Corp.	*NULL*	lmn@example.com
okay	*NULL*	*NULL*	*NULL*
bad	*NULL*	*NULL*	roberts.amy@acme.com

Followed by a rating order:

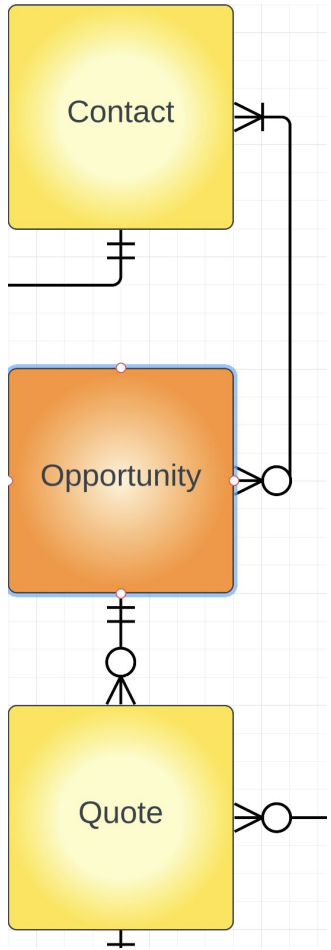
```
select *
from [strong_leads_view]
ORDER BY CASE lead_rating
    WHEN 'good' THEN 1
    WHEN 'okay' THEN 2
    ELSE 3
END;
```

Opportunity Table and Implementation

```
CREATE TABLE Opportunity (
opportunity_id int NOT NULL PRIMARY KEY,
opportunity_name VARCHAR(255),
opportunity_account VARCHAR(255),
opportunity_amount DECIMAL(25,2),
opportunity_status VARCHAR(25),

CONSTRAINT CHECK_STATUS CHECK
  (opportunity_status LIKE 'In Progress%'
  OR opportunity_status LIKE 'Done%')
CONSTRAINT Quote FOREIGN KEY (Quote_id)
   REFERENCES Quote(quote_id)
);
```

opportunity_id	opportunity_name	opportunity_amount	*opportunity_status*	quote_id
2	Shopify Workforce Development	350000099.00	Done	2
4	Mobile App Development Project	750000.00	Done	4
1	Government NNCI CHIPS Act Funding	25000000.00	In Progress	1
3	E-commerce Integration Solution	1000000.00	In Progress	3
5	Data Analytics Platform Upgrade	5000000.00	In Progress	5

A local view of Opportunity



Opportunity View: Quick Overview

```
SELECT [cis55_50].[dbo].[Opportunity].[opportunity_name],
        [cis55_50].[dbo].[Opportunity].[opportunity_amount],
        [cis55_50].[dbo].[Account].[Account_name],
        [cis55_50].[dbo].[Account].[Account_description]
    FROM [cis55_50].[dbo].[Opportunity]
JOIN [cis55_50].[dbo].[Contact] ON opportunity.opportunity_id = Contact.contact_id
JOIN [cis55_50].[dbo].[Account] ON Account.account_id = contact.contact_id;
```

Followed by a rating order:

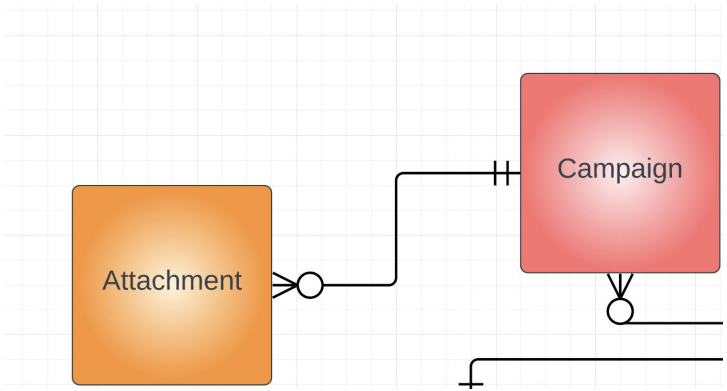
```
SELECT * FROM
opportunity_overview
ORDER BY CASE lead_rating
    WHEN 'good' THEN 1
    WHEN 'okay' THEN 2
    ELSE 3
END;
```

opportunity_name	opportunity_amount	Account_name	Account_description	lead_rating
Government NNCI CHIPS Act Funding	25000000.00	Account A	Description B	okay
Shopify Workforce Development	350000099.00	Account B	Description B	okay
Mobile App Development Project	750000.00	Account D	Description D	okay
Data Analytics Platform Upgrade	5000000.00	Account E	Description E	okay
E-commerce Integration Solution	1000000.00	Account C	Description C	bad

Attachment Table and Implementation

```
CREATE TABLE Attachment(
attachment_id INT NOT NULL PRIMARY KEY,
attachment_description VARCHAR(255) NOT NULL,
attachment_type VARCHAR(255) NOT NULL,
attachment_status VARCHAR(255) NOT NULL,

CONSTRAINT CHECK_STATUS CHECK (attachment_status
 LIKE 'READY%' OR attachment_status LIKE 'SENT%')
);
```

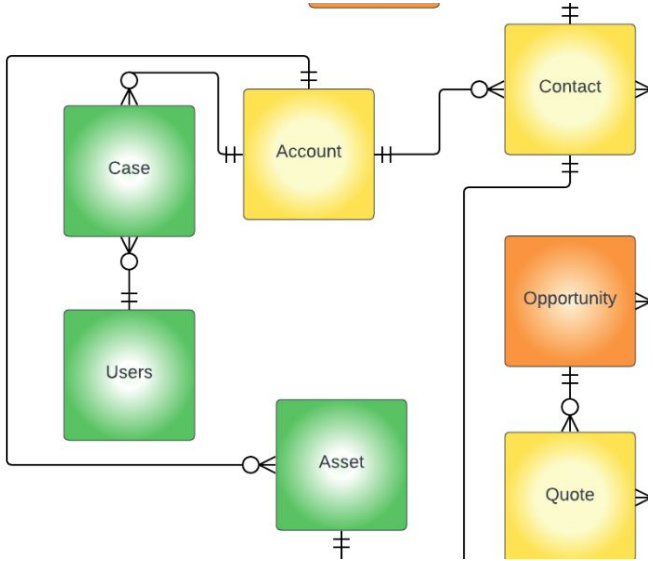
A local view of Attachment



attachment_id	attachment_description	attachment_type	attachment_status	attachment_quote_id
1	contract agreement	DOCX	SENT	3
2	presentation slides	PPTX	READY	4
3	contract agreement	DOCX	SENT	3
4	presentation slides	PPTX	READY	4
5	spreadsheet analysis	XLSX	SENT	5
6	image of product	JPG	READY	6
7	video demonstration	MP4	SENT	7
8	audio recording	WAV	READY	8
9	design mockup	PSD	SENT	9
10	data analysis report	PDF	READY	10
11	logo design	AI	SENT	11
12	code snippet	TXT	READY	12

Account Table and Implementation

```
CREATE TABLE Account (

account_id INT NOT NULL PRIMARY KEY,

account_name VARCHAR(255) NULL,

account_location VARCHAR(255) NULL,

account_description VARCHAR(500) NULL

);
```



	account_id	Account_name	Account_location	Account_description
1	1	Account A	Boston	Description B
2	2	Account B	Boston	Description B
3	3	Account C	Boston	Description C
4	4	Account D	Boston	Description D
5	5	Account E	Boston	Description E

Contact Table and Implementation

```
Create table Contact (

Contact_id int not null primary key,

Account_id int,

Lead_id int,

Contact_name varchar(225),

Title varchar(225),

Opportunity_id int,

Phone varchar(100),

Fax varchar(100),

Email varchar(225),

Foreign key (Account_id) references dbo.Account(account_id),

Foreign key (Lead_id) references dbo.Lead(lead_id));
```

Results | Messages

	Contact_id	Account_id	Lead_id	Contact_Name	Title	Opportunity_id	Phone	Fax	Email
1	1	1	1	Jane Smith	Operation Manager	1	555-1234	-7045	john.doe@acme.com
2	2	2	2	John Doe	Bussiness Relation Manager	2	555-5678	-7655	smith.jane@acme.com
3	3	3	3	Mark Johnson	Bussiness Relation Manager	3	555-9876	-7656	miller.jane@acme.com
4	4	4	4	Sarah Miller	Bussiness Relation Manager	4	555-5432	-7657	anderson.sam@acme.com
5	5	5	5	Chris Anderson	Bussiness Relation Manager	5	555-7654	-7658	thomas.lisa@acme.com

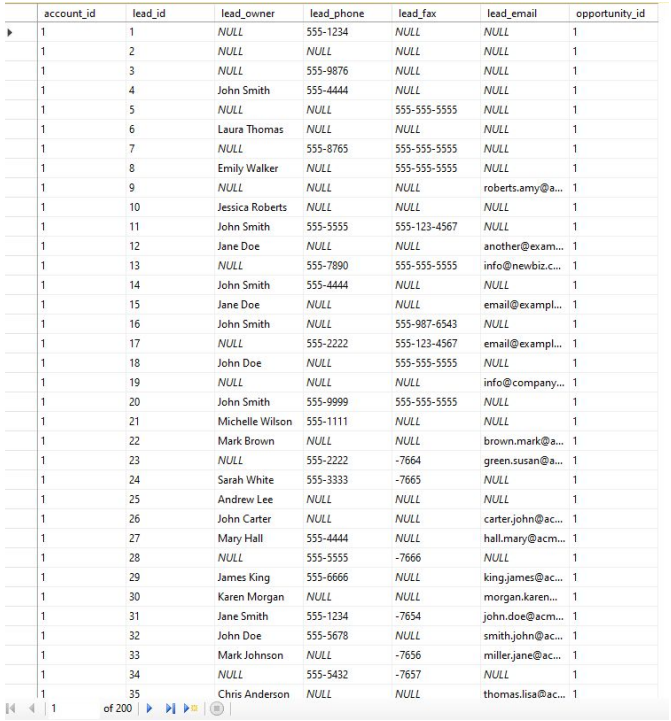
Contact Table View

Create view Contact_View

As select account_id, lead_id,

lead_owner, lead_phone, lead_fax, lead_email, opportunity_id

from [cis55_50].[dbo].[Account], [cis55_50].[dbo].[Lead], [cis55_50].[dbo].[Opportunity];

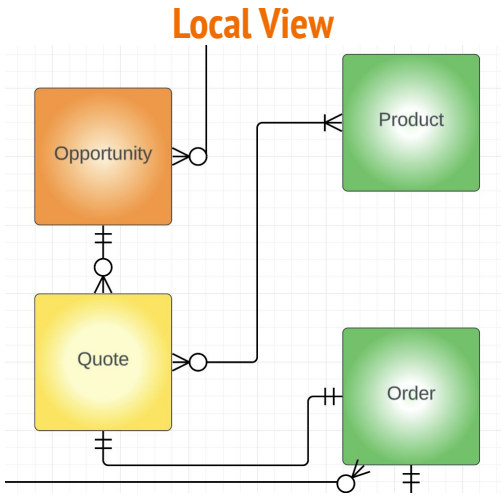


account_id	lead_id	lead_owner	lead_phone	lead_fax	lead_email	opportunity_id
1	1	NULL	555-1234	NULL	NULL	1
1	2	NULL	NULL	NULL	NULL	1
1	3	NULL	555-9876	NULL	NULL	1
1	4	John Smith	555-4444	NULL	NULL	1
1	5	NULL	NULL	555-555-5555	NULL	1
1	6	Laura Thomas	NULL	NULL	NULL	1
1	7	NULL	555-8765	555-555-5555	NULL	1
1	8	Emily Walker	NULL	555-555-5555	NULL	1
1	9	NULL	NULL	NULL	roberts.amy@a...	1
1	10	Jessica Roberts	NULL	NULL	NULL	1
1	11	John Smith	555-5555	555-123-4567	NULL	1
1	12	Jane Doe	NULL	NULL	another@exam...	1
1	13	NULL	555-7890	555-555-5555	info@newbiz.c...	1
1	14	John Smith	555-4444	NULL	NULL	1
1	15	Jane Doe	NULL	NULL	email@exampl...	1
1	16	John Smith	NULL	555-987-6543	NULL	1
1	17	NULL	555-2222	555-123-4567	email@exampl...	1
1	18	John Doe	NULL	555-555-5555	NULL	1
1	19	NULL	NULL	NULL	info@company...	1
1	20	John Smith	555-9999	555-555-5555	NULL	1
1	21	Michelle Wilson	555-1111	NULL	NULL	1
1	22	Mark Brown	NULL	NULL	brown.mark@a...	1
1	23	NULL	555-2222	-7664	green.susan@a...	1
1	24	Sarah White	555-3333	-7665	NULL	1
1	25	Andrew Lee	NULL	NULL	NULL	1
1	26	John Carter	NULL	NULL	carter.john@ac...	1
1	27	Mary Hall	555-4444	NULL	hall.mary@acm...	1
1	28	NULL	555-5555	-7666	NULL	1
1	29	James King	555-6666	NULL	king.james@ac...	1
1	30	Karen Morgan	NULL	NULL	morgan.karen...	1
1	31	Jane Smith	555-1234	-7654	john.doe@acm...	1
1	32	John Doe	555-5678	NULL	smith.john@ac...	1
1	33	Mark Johnson	NULL	-7656	miller.jane@ac...	1
1	34	NULL	555-5432	-7657	NULL	1
1	35	Chris Anderson	NULL	NULL	thomas.lisa@ac...	1

1 of 200

Quote Table and Implementation

```
CREATE TABLE Quote(
quote_id INT NOT NULL Primary Key,
opportunity_id INT NULL Foreign Key,
order_ID INT NULL Foreign Key,
product_id INT NULL Foreign Key;
pricing decimal(10, 2) NULL,
quantity INT NULL,
status VARCHAR(10) NULL,
CONSTRAINT CHK_STATUS CHECK (status in
('Drafting','Sent', 'Revision', 'Accepted', 'Rejected'))
quote_date DATE NULL);
```



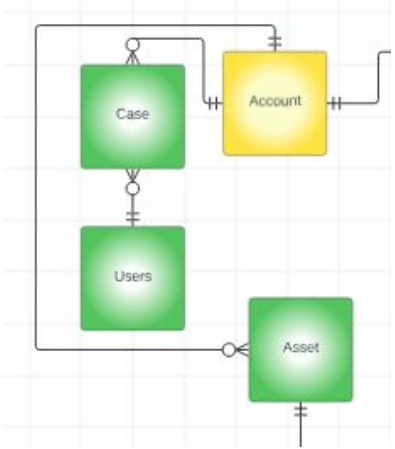

```
ALTER TABLE Quote
ADD FOREIGN KEY (opportunity_id)
REFERENCES Opportunity
(opportunity_id);
```

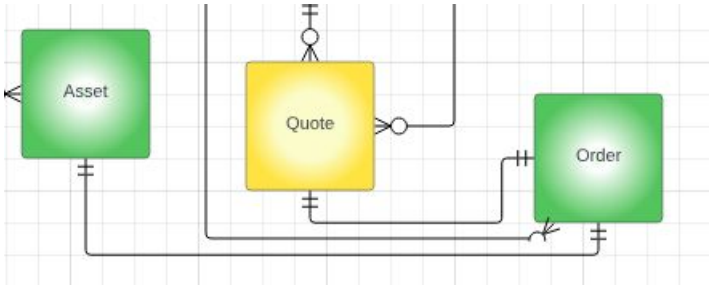
```
ALTER TABLE Quote
ADD FOREIGN KEY (product_id)
REFERENCES Product (product_id);
```

```
ALTER TABLE Quote
ADD FOREIGN KEY (order_id)
REFERENCES Order (order_id);
```

	quote_id	opportunity_id	order_id	product_id	pricing	quantity	status	quote_date
1	1	1	1	1	67917.38	1	sent	2023-06-29
2	2	2	2	2	15646.83	2	accepted	2022-08-21
3	3	3	3	3	21535.02	3	rejected	2022-11-13
4	4	4	4	4	86753.09	4	revision	2023-07-08
5	5	5	5	5	67899982.12	5	drafting	2023-07-06

Asset Table and Implementation






```
CREATE TABLE Asset (
    Asset_id INT NOT NULL PRIMARY KEY,
    Contact_id INT,
    Case_id INT,
    Product_id INT,
    Asset_name VARCHAR(255) NOT NULL,
    Asset_type VARCHAR(50) NOT NULL,
    Purchase_date DATE NOT NULL,
    Price DECIMAL(10,2) NOT NULL,
    Location VARCHAR(255),
    Status VARCHAR(50) NOT NULL,
);
```

```
ALTER TABLE Asset
ADD CONSTRAINT FK_Asset_Order
FOREIGN KEY (order_ID) REFERENCES Orders(order_ID);
```

```
ALTER TABLE Asset
ADD CONSTRAINT FK_Asset_Orders
FOREIGN KEY (order_ID) REFERENCES Orders(order_ID);
```

	Asset_id	order_ID	account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	1	Macbook	Laptop	2023-06-30	1200.00	Los Angeles	Active
2	101	2	2	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive
3	102	3	3	Dell XPS	Laptop	2023-06-20	0.00	San Francisco	sold
4	103	4	4	Samsung TV	Television	2023-06-15	0.00	Houston	Sold
5	104	5	5	HP Pavilion	Laptop	2023-06-10	1850.00	Chicago	In Repair
6	105	6	6	Canon EOS R	Camera	2023-06-05	2500.00	New York	Active

Asset Views

Assets with “Active” Status

```
CREATE VIEW ActiveAssets AS SELECT * FROM
[cis55_50].[dbo].[Asset] WHERE Status = 'Active';

SELECT * FROM ActiveAssets;
```

	Asset_id	order_id	Account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	1200.00	Los Angeles	Active
2	102	3	102	Dell XPS	Laptop	2023-06-20	1500.00	San Francisco	Active
3	105	6	105	Canon EOS R	Camera	2023-06-05	2500.00	New York	Active

To view Mobile Phones and Laptops

```
CREATE VIEW LaptopMobileAssets AS
SELECT * FROM
[cis55_50].[dbo].[Asset] WHERE
Asset_type IN ('Laptop', 'Mobile Phone');

SELECT * FROM
LaptopMobileAssets;
```

	Asset_id	order_id	Account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	1200.00	Los Angeles	Active
2	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive
3	102	3	102	Dell XPS	Laptop	2023-06-20	1500.00	San Francisco	Active
4	104	5	104	HP Pavilion	Laptop	2023-06-10	1850.00	Chicago	In Repair
5	109	10	109	Samsung Galaxy S21	Mobile Phone	2023-05-16	1150.00	Atlanta	In Repair
6	114	15	114	Sony Xperia	Mobile Phone	2023-04-21	900.00	Miami	Inactive

Views cont'd.

```
1  CREATE VIEW ExpensiveAsset2 AS
2  SELECT *
3  FROM [cis55_50].[dbo].[Asset]
4  WHERE Price > 1000.00;
```

	Asset_id	order_id	Account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	1200.00	Los Angeles	Active
2	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive
3	102	3	102	Dell XPS	Laptop	2023-06-20	1500.00	San Francisco	Active
4	104	5	104	HP Pavilion	Laptop	2023-06-10	1850.00	Chicago	In Repair
5	105	6	105	Canon EOS R	Camera	2023-06-05	2500.00	New York	Active
6	108	9	108	LG OLED TV	Television	2023-05-21	3000.00	Denver	Active
7	109	10	109	Samsung Galaxy S21	Mobile Phone	2023-05-16	1150.00	Atlanta	In Repair
8	110	11	110	Microsoft Surface Pro	Tablet	2023-05-11	1500.00	San Diego	Active

```
8  CREATE VIEW ExpensiveAsset3 AS
9  SELECT *
10 FROM [cis55_50].[dbo].[Asset]
11 WHERE Price > 1000.00 AND Status = 'inactive';
12
13 Select * From ExpensiveAsset3
```

	Asset_id	order_ID	account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive

Triggers

	Asset_id	order_ID	account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	1200.00	Los Angeles	Active
2	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive

If the status of any asset were to switch to “In Repair” I set the price to add $50.00.
This is for maintenance fee.

```
CREATE TRIGGER UpdateAssetPrices
ON Asset
AFTER UPDATE
AS
BEGIN
    IF UPDATE(Status)
    BEGIN
        IF EXISTS (SELECT 1 FROM INSERTED i JOIN DELETED d ON i.Asset_id = d.Asset_id WHERE i.Status = 'In Repair')
        BEGIN
            UPDATE Asset
            SET Price = Price + 50.00
            WHERE Asset_id IN (SELECT Asset_id FROM INSERTED WHERE Status = 'In Repair')
        END
        END
END;
```

```
UPDATE Asset
SET Status = 'In repair'
WHERE Asset_id = '100'

select * from Asset
```

	Asset_id	order_ID	account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	1250.00	Los Angeles	In repair
2	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive

Triggers cont'd.

```sql
CREATE TRIGGER StatusSoldPrice0
ON Asset
AFTER UPDATE
AS
BEGIN
    IF UPDATE(Status)
    BEGIN
        UPDATE a
        SET a.Price = CASE
            WHEN i.Status = 'Sold' THEN 0.00
            WHEN d.Status = 'Sold' THEN 1200.00
            ELSE a.Price
        END
        FROM Asset AS a
        INNER JOIN INSERTED AS i ON a.Asset_id = i.Asset_id
        INNER JOIN DELETED AS d ON a.Asset_id = d.Asset_id
        WHERE i.Status <> d.Status;
    END
END;
```

```sql
3  update Asset
4  set status = 'sold'
5  where account_id = 102
```

100 %

Results | Messages

	Asset_id	order_ID	account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	1200.00	Los Angeles	Active
2	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive
3	102	3	102	Dell XPS	Laptop	2023-06-20	0.00	San Francisco	sold

Stored Procedure Activates a Trigger

```
CREATE PROCEDURE UpdateAssetStatusss
    @Asset_id INT,
    @New_Status VARCHAR(50)
AS
BEGIN
    UPDATE Asset
    SET Status = @New_Status
    WHERE Asset_id = @Asset_id;
END;
```

SQLQuery8.sql - 209...5_50 (cis55_50 (59))* SQLQuery7.sql - 209...5_50 (cis55_50 (58))* SQLQuery6.sql - 209...5_50 (cis55_50

```
exec UpdateAssetStatusss
    @Asset_id = 100,
    @New_Status = 'Sold';

Select * from Asset
```

100 %

Results Messages

	Asset_id	order_ID	account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	0.00	Los Angeles	Sold
2	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive

```
exec UpdateAssetStatusss
    @Asset_id = 100,
    @New_Status = 'Active';

    Select * from Asset
```

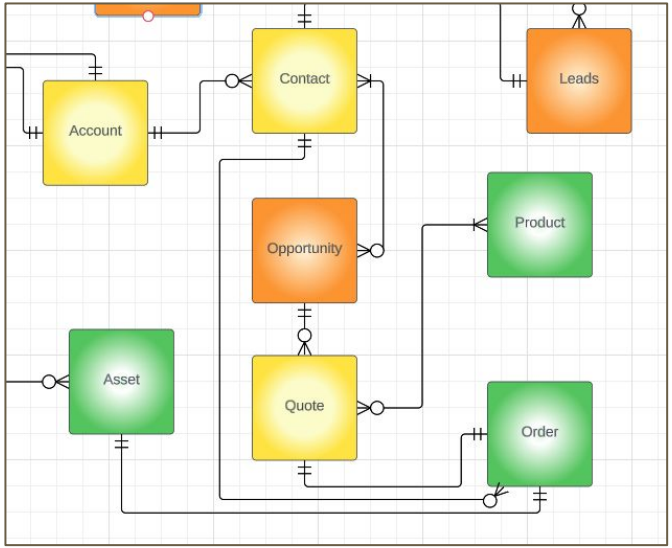
100 %

Results Messages

	Asset_id	order_ID	account_id	Asset_name	Asset_type	Purchase_date	Price	Location	Status
1	100	1	100	Macbook	Laptop	2023-06-30	1200.00	Los Angeles	Active
2	101	2	101	iPhone	Mobile Phone	2023-06-25	1200.00	Miami	Inactive

Order Table and Implementation

```
CREATE TABLE Orders(
    order_ID INT NOT NULL PRIMARY KEY,
    order_status VARCHAR(255) NULL,
    order_method VARCHAR(255) NULL,
    order_QuoteID INT NULL,
    order_Contact_id INT NULL,
    order_Asset_id INT NULL
)
```

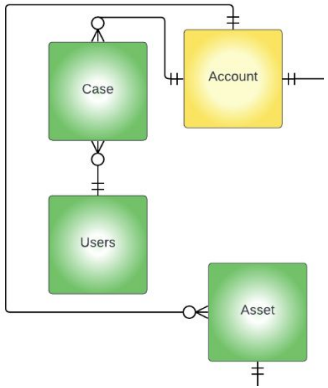


	order_ID	order_status	order_method	order_quote_id	order_Contact_id	order_Asset_id
1	1	Received	Phone	1	1	100
2	2	Delivered	Email	2	2	101
3	3	In Progress	Website	3	3	102
4	4	Shipping	Phone	4	4	103

Users Table and Implementation

```
CREATE TABLE [User] (
user_id INT(50) NOT NULL Primary Key,
case_id INT(50) NOT NULL,
user_name VARCHAR(50) NULL,
pass_word VARCHAR(50) NULL,
name_f VARCHAR(50) NULL,
name_l VARCHAR(50) NULL
);
```

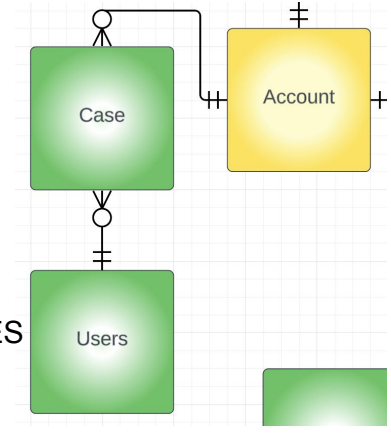
user_id	case_id	user_name	pass_word	name_f	name_l
1	101	johnsmith	password123	John	Smith
2	102	janedoe	password456	Jane	Doe
3	103	mikesmith	password789	Mike	Smith



Case Table and Implementation

ALTER TABLE Cases
ADD FOREIGN KEY
(User_id) REFERENCES
Users (User_id);

ALTER TABLE Cases
ADD FOREIGN KEY
(Account_id) REFERENCES
Account (Account_id);




```
CREATE TABLE Cases (
    Case_id INT NOT NULL PRIMARY KEY,
    User_id INT,
    Account_id INT,
    Case_number VARCHAR(50) NOT NULL,
    Case_status VARCHAR(50) NOT NULL,
    Case_type VARCHAR(50) NOT NULL,
    Subject VARCHAR(255) NOT NULL,
    Description TEXT NOT NULL,
    Created_date DATETIME NOT NULL,
    Closed_date DATETIME,
    Resolution TEXT,
    CONSTRAINT CHK_ClosedDate CHECK (Closed_date >= Created_date),
    CONSTRAINT UK_CaseNumber UNIQUE (Case_number)
);
```

	Case_id	User_id	Account_id	Case_number	Case_status	Case_type	Subject	Description	Created_date	Closed_date	Resolution
1	1	1	1	CASE001	Open	Complaint	Product Defec…	I received a …	2023-06-01 10…	*NULL*	*NULL*
2	2	2	2	CASE002	Closed	Enquiry	Product Infor…	My product is…	2023-05-03 15…	2023-06-03 11…	Issue Resolve…
3	3	3	3	CASE003	Open	Ticket	Service Reque…	I need assist…	2023-06-03 14…	*NULL*	*NULL*
4	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

CASES - TRIGGER

```
UPDATE Cases
SET Case_status = 'Closed'
WHERE Case_id = 1;
```

```
CREATE TRIGGER SetClosedDate
ON Cases
AFTER UPDATE
AS
BEGIN
   IF UPDATE(Case_status) AND EXISTS (SELECT * FROM inserted WHERE
Case_status = 'Closed')
   BEGIN
      UPDATE Cases
      SET Closed_date = GETDATE()
      FROM Cases
      INNER JOIN inserted ON Cases.Case_id = inserted.Case_id
      WHERE Cases.Case_id IN (SELECT Case_id FROM inserted WHERE
Case_status = 'Closed')
   END
END;
```

	Case_id	User_id	Account_id	Case_number	Case_status	Case_type	Subject	Description	Created_date	Closed_date	Resolution
1	1	1	1	CASE001	Open	Complaint	Product Defec…	I received a …	2023-06-01 10…	*NULL*	*NULL*
2	2	2	2	CASE002	Closed	Enquiry	Product Infor…	My product is…	2023-05-03 15…	2023-06-03 11…	Issue Resolve…
3	3	3	3	CASE003	Open	Ticket	Service Reque…	I need assist…	2023-06-03 14…	*NULL*	*NULL*
4	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

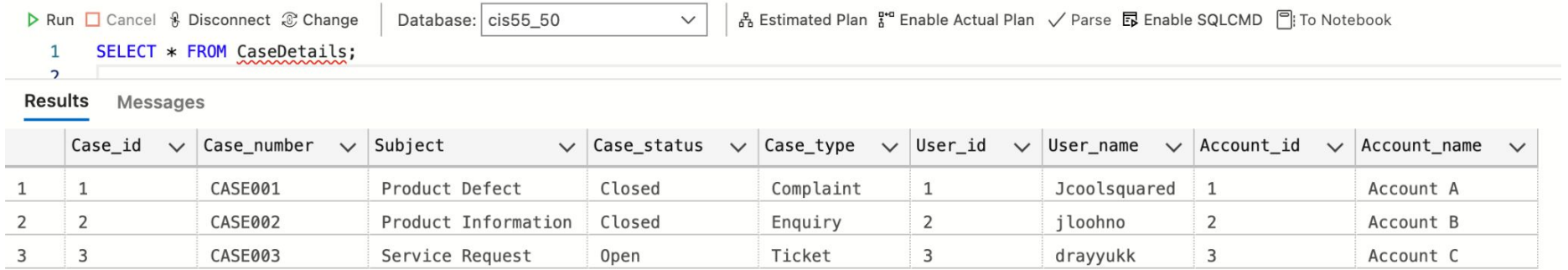
Results Messages

se_id	User_id	Account_id	Case_number	Case_status	Case_type	Subject	Description	Created_date	Closed_date
	1	1	CASE001	Closed	Complaint	Product Defect	I received a faulty product	2023-06-01 10:00:00.000	2023-07-11
	2	2	CASE002	Closed	Enquiry	Product Information	My product is broken	2023-05-03 15:35:05.000	2023-06-03
	3	3	CASE003	Open	Ticket	Service Request	I need assistance with a service issue	2023-06-03 14:45:00.000	*NULL*

CASES - VIEW

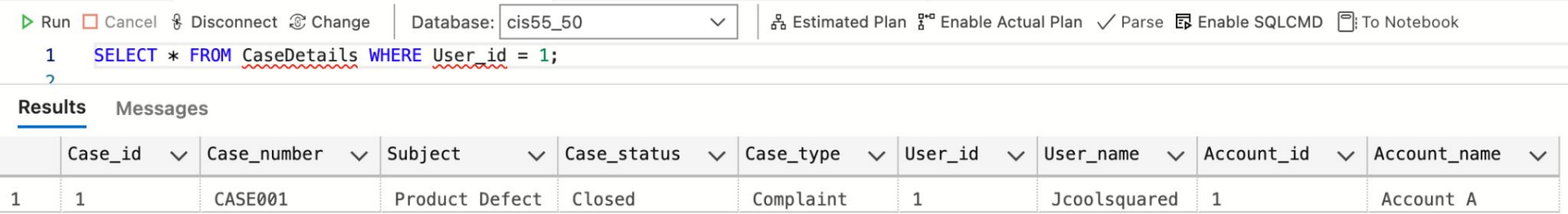
```
CREATE VIEW CaseDetails AS
SELECT C.Case_id, C.Case_number, C.Subject,
C.Case_status, C.Case_type, U.User_id, U.User_name,
CO.Account_id, CO.Account_name
FROM Cases C
JOIN Users U ON C.User_id = U.User_id
JOIN Account CO ON C.Account_id = CO.Account_id;
```

SELECT * FROM CaseDetails;



	Case_id	Case_number	Subject	Case_status	Case_type	User_id	User_name	Account_id	Account_name
1	1	CASE001	Product Defect	Closed	Complaint	1	Jcoolsquared	1	Account A
2	2	CASE002	Product Information	Closed	Enquiry	2	jloohno	2	Account B
3	3	CASE003	Service Request	Open	Ticket	3	drayyukk	3	Account C

SELECT * FROM CaseDetails WHERE User_id = 1;



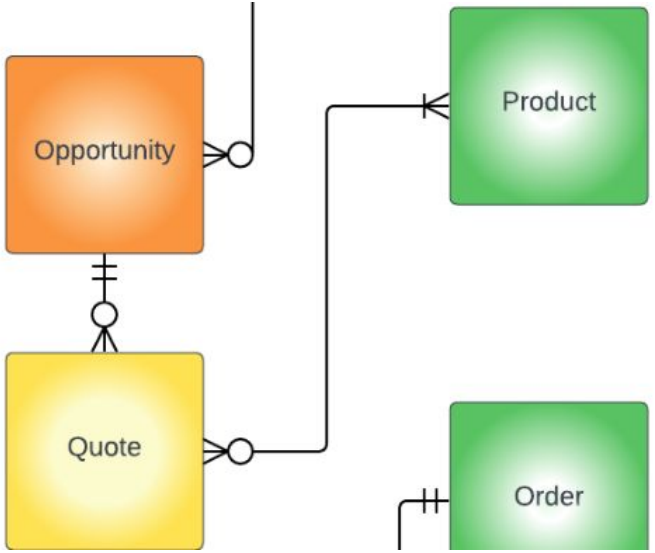
	Case_id	Case_number	Subject	Case_status	Case_type	User_id	User_name	Account_id	Account_name
1	1	CASE001	Product Defect	Closed	Complaint	1	Jcoolsquared	1	Account A

Product Table and Implementation

CREATE TABLE Product (

product_id int not null primary key,
product_name varchar(255) null,
product_price int null,
product_description varchar(500) null

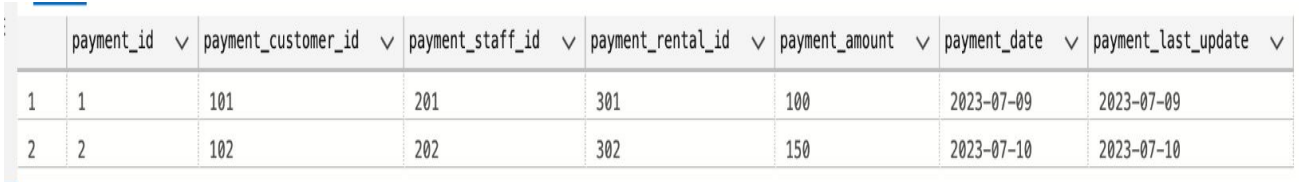
)



	product_id	product_name	product_price	product_description
1	1	Product A	10	Description A
2	2	Product B	10	Description B
3	3	Product C	10	Description C
4	4	Product D	10	Description D
5	5	Product E	10	Description E

Payment Table and Implementation

```
CREATE TABLE
CREATE TABLE payments (
payment_id INT NOT NULL PRIMARY
KEY,
payment_customer_id INT NOT NULL,
payment_staff_id INT NOT NULL,
payment_rental_id INT NOT NULL,
payment_amount INT NOT NULL,
payment_date DATE NOT NULL,
payment_last_update DATE,
);
```



	payment_id	payment_customer_id	payment_staff_id	payment_rental_id	payment_amount	payment_date	payment_last_update
1	1	101	201	301	100	2023-07-09	2023-07-09
2	2	102	202	302	150	2023-07-10	2023-07-10

Payment_Method Table and Implementation

Create table:

```
Create table Payment_Method (
Credit_Card_type varchar(100) not NULL,
Credit_Number int not NULL,
Card_Expiration_Date DATE not NULL,
Account_id int not null,
payment_dateDATE not NULL,
payment_id INT NOT NULL);
```

```
ALTER TABLE Payment_Method ADD FOREIGN KEY
(Account_id) REFERENCES Account (Account_id)
ALTER TABLE Payment_Method ADD FOREIGN KEY
(payment_date) REFERENCES Payments
(payment_date)
ALTER TABLE Payment_Method ADD FOREIGN KEY
(payment_id) REFERENCES Payments (payment_id)
```

select * from Payment_method

	Credit_Card_type	Credit_Number	Card_Expiration_Date	Account_id	payment_date	payment_id
1	Amex	53612312	2025-07-05	2	2023-06-04	1
2	Master	24231238	2023-12-11	3	2023-07-02	2
3	Master	261724123	2027-05-05	1	2023-07-13	1
4	Discovery	66651423	2026-01-03	4	2023-07-09	2
5	Visa	40392134	2027-04-06	1	2023-06-02	1

6. Conclusion

Learned several things throughout this process:

- Practical implementation of Sequel into the workplace
- Developing an approach to a business
- Benefits of ER Diagrams as well as entity relationships with one another
- Thank you to Professor Samir Magid and thank you for listening!