OPEN-ENDED LAB
DBMS (SE-204)

SalesTree

Kashif Ali SE-22074
Abdul Moiz SE-22073
Abdul Rauf Kaka SE-22094

Submitted To: Engr. Shiza Riaz Memon, Lecturer, Department of Software Engineering

Department of Software Engineering
NED University of Engineering & Technology

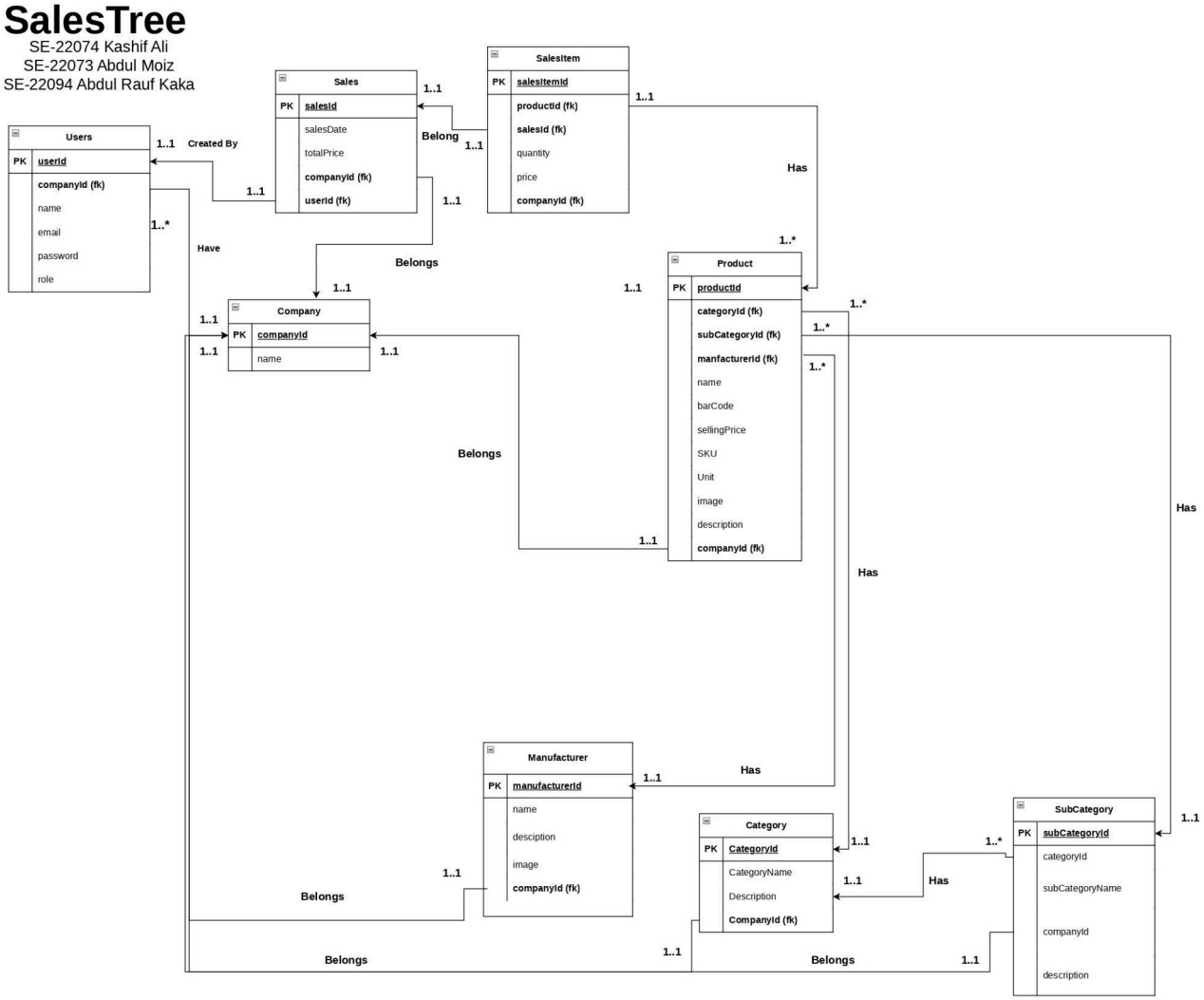
1. Introduction

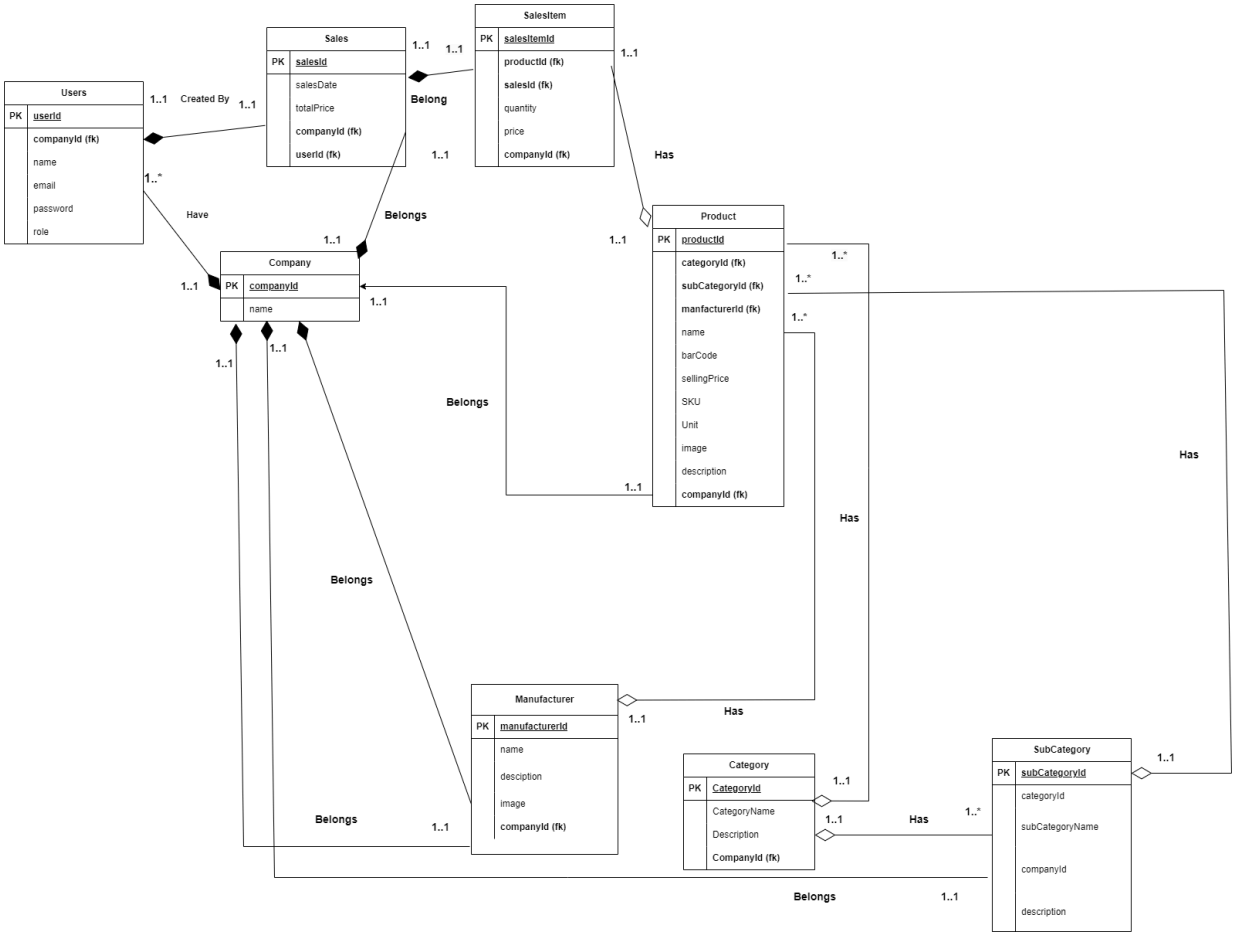
SalesTree is the ultimate platform for all POS needs, designed to streamline inventory management and transaction processes across various industries. With its intuitive interface and comprehensive features, SalesTree enables businesses to operate more efficiently and cost-effectively, empowering users to optimize their operations seamlessly.

The features of this system are:

- User-friendly interface
- Role-based access control
- Robust inventory management
- Sales management tools
- User management functionalities

2. Data Model (ERD & EERD)





3. Normalization

1NF

ID	user_id	created_by	role	status	created_at	updated_at	is_delete
1	2	1	Admin	Inactive	24-06-2024	24-06-2024	F
2	3	1	Sales Associate	Active	24-06-2024	24-06-2024	F
3	4	2	Sales Associate	Active	24-06-2024	24-06-2024	F

Notice the problem with data redundancy. If you, for some reasons, decide to change the role name from **“Sales Associate”** to **“User”**, you must find all uses of the original role name in the employee table and update them to the new role name. Performing these updates is not only more work than necessary, but it can also lead to inconsistent data states if some instances are updated, and others are not. You can avoid this pitfall by separating the role into a new table and creating a relationship between employee and role.

Role Table

Id	role_name	created_at	updated_at	is_deleted

Employee Table

ID	user_id	created_by	role_id	status	created_at	updated_at	is_delete

2NF

The given tables are already in 2NF,no changes required

3NF

The given table already satisfies the conditions of 3NF, no changes required.

4. Database Schema

a. Creation of Database

Create database salestree;

b. Creation of Tables

i. User Table

```
CREATE TABLE users (
   ID serial PRIMARY KEY,
   name varchar(255) NOT NULL,
   profile_image varchar(255),
   email varchar(255) UNIQUE NOT NULL,
   company_id integer NOT NULL,
   password varchar(255) NOT NULL,
   address TEXT, -- Changed type to TEXT
   currency varchar(10),
   timezone varchar(50),
   created_at timestamp default current_timestamp,
   updated_at timestamp default current_timestamp,
   is_delete boolean default false,
   FOREIGN KEY (company_id) REFERENCES company (ID) ON
DELETE CASCADE
);
```

ii. Employee Table

```
CREATE TABLE employees (
   ID serial PRIMARY KEY,
   user_id INT,
   created_by INT,
   role varchar(20) CHECK (role IN ('admin', 'sales associate')) NOT
NULL default 'sales associate',
   status varchar(20) CHECK (status IN ('active','inactive')) NOT NULL
default 'active',
   created_at timestamp default current_timestamp,
   updated_at timestamp default current_timestamp,
   is_delete boolean default false,
   FOREIGN KEY (user_id) REFERENCES users (ID) ON DELETE
CASCADE,
   FOREIGN KEY (created_by) REFERENCES users (ID) ON DELETE
CASCADE
);
```

iii. Company Table

```
CREATE TABLE company (
   ID serial PRIMARY KEY,
   name varchar(30) UNIQUE NOT NULL ,
   created_at timestamp default current_timestamp,
   updated_at timestamp default current_timestamp,
   is_delete boolean default false
);
```

iv. Category Table

```
CREATE TABLE category (
   ID serial PRIMARY KEY,
   name varchar(255) NOT NULL,
   description TEXT NOT NULL,
   company_id integer NOT NULL,
   created_user INT NOT NULL,
   created_at timestamp default current_timestamp,
   updated_at timestamp default current_timestamp,
   is_delete boolean default false,
   FOREIGN KEY (created_user) REFERENCES users (ID) ON DELETE
CASCADE,
   FOREIGN KEY (company_id) REFERENCES company (ID) ON
DELETE CASCADE
);
```

v. Sub Category Table

```
CREATE TABLE sub_category (
   ID serial PRIMARY KEY,
   name varchar(255) NOT NULL,
   description TEXT NOT NULL,
   created_user INT NOT NULL, -- Enclosed user in double quotes
   company_id INT NOT NULL,
   category INT NOT NULL,
   FOREIGN KEY (created_user) REFERENCES users (ID) ON DELETE
CASCADE, -- Enclosed user in double quotes
   FOREIGN KEY (company_id) REFERENCES company (ID) ON
DELETE CASCADE,
   FOREIGN KEY (category) REFERENCES category (ID) ON DELETE
CASCADE,
   created_at timestamp default current_timestamp,
```

```
    updated_at timestamp default current_timestamp,
    is_delete boolean default false
);
```

vi. Product Table

```
CREATE TABLE product (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT NOT NULL,
    company_id INTEGER NOT NULL,
    category_id INTEGER NOT NULL,
    manufacture_id INTEGER NOT NULL,
    sku VARCHAR(100) NOT NULL,
    unit VARCHAR(100) NOT NULL,
    selling_price DECIMAL(20,2) NOT NULL,
    barcode VARCHAR(100) NOT NULL,
    image VARCHAR(255) NOT NULL,
    sub_category_id INTEGER NOT NULL,
    FOREIGN KEY (company_id) REFERENCES company (ID) ON
DELETE CASCADE,
    FOREIGN KEY (category_id) REFERENCES category (ID) ON
DELETE CASCADE,
    FOREIGN KEY (manufacture_id) REFERENCES manufacture (ID) ON
DELETE CASCADE,
    FOREIGN KEY (sub_category_id) REFERENCES sub_category (ID)
ON DELETE CASCADE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    is_delete BOOLEAN DEFAULT FALSE,
);
```

vii. Manufacture Table

```
CREATE TABLE manufacture(
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT NOT NULL,
    company_id INT NOT NULL,
    created_user INT NOT NULL,
    FOREIGN KEY (created_user) REFERENCES users (ID) ON DELETE
CASCADE,
    FOREIGN KEY (company_id) REFERENCES company (ID) ON
DELETE CASCADE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    is_delete BOOLEAN DEFAULT FALSE,
    image VARCHAR(255)
);
```

viii. Sales Table

```
CREATE TABLE sales (
    id SERIAL PRIMARY KEY,
    customer_name VARCHAR(255) NOT NULL,
    created_user INT NOT NULL,
    total_cost INT NOT NULL,
    company_id INTEGER NOT NULL,
    FOREIGN KEY (company_id) REFERENCES company (ID) ON
DELETE CASCADE,
    FOREIGN KEY (created_user) REFERENCES users (ID) ON DELETE
CASCADE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    is_delete BOOLEAN DEFAULT FALSE,
    customer_phoneno VARCHAR(255) NOT NULL,
    cash_received INT NOT NULL,
    discount INT NOT NULL,
    cash_refund INT NOT NULL,
    payment_method VARCHAR(255) NOT NULL
);
```

ix. Sales Item Table

```
CREATE TABLE sales_item (
    id SERIAL PRIMARY KEY,
    sales_id INT NOT NULL,
    product_id INT NOT NULL,
    created_user INT NOT NULL,
    company_id INTEGER NOT NULL,
    FOREIGN KEY (company_id) REFERENCES company (ID) ON
DELETE CASCADE,
    cost INT NOT NULL,
    quantity INT NOT NULL,
    FOREIGN KEY (created_user) REFERENCES users (ID) ON DELETE
CASCADE,
    FOREIGN KEY (sales_id) REFERENCES sales (ID) ON DELETE
CASCADE,
    FOREIGN KEY (product_id) REFERENCES product (ID) ON DELETE
CASCADE,
```

```
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    is_delete BOOLEAN DEFAULT FALSE
);
```

5. Sample Data

a. Company Table

```
1  INSERT INTO public.company(
2      name)
3      VALUES ('NED');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 94 msec.
```

b. Category Table

```
1  INSERT INTO public.category(
2      name, description , company_id,created_user)
3      VALUES ('Clothing', 'This is Clothing Category',13,1);
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 99 msec.
```

c. SubCategory Table

```
INSERT INTO public.sub_category(
     name, description, created_user, company_id, category)
    VALUES ('Men Clothing', 'This is Men Clothing', 1, 13, 7);
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 71 msec.
```

d. Manufacture Table

```
INSERT INTO public.manufacture(
     name, description, company_id, created_user)
    VALUES ('Khaadi', 'Most Popular Clothing Brand', 13, 1);
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 63 msec.
```

e. Product Table

```
INSERT INTO public.product(
    name, description, company_id, category_id, manufacture_id, sku, selling_price, barcode, sub_category_id, unit,image)
    VALUES ('White Kurta', 'This is Cotton Kurta', 13, 7, 4, 100, 200,'01123456',9,100,'null');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 56 msec.
```

f. Sales Table

```
INSERT INTO public.sales(
    customer_name, created_user, total_cost, customer_phoneno, cash_received, discount, cash_refund, payment_method, company_id)
    VALUES ('Alex', 1, 100, '0332467333', 200, 100, 100,'CHEQUE', 7);
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 281 msec.
```

6. Functional Queries

a. Company Table

i. Select

```
SELECT id, name, created_at, updated_at, is_delete
    FROM public.company;
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (30)	created_at timestamp without time zone	updated_at timestamp without time zone	is_delete boolean
1	1	Vs	2024-03-30 18:42:37.676849	2024-03-30 18:42:37.676849	false
2	2	Vs2	2024-03-30 19:38:25.649317	2024-03-30 19:38:25.649317	false
3	3	Example Organization	2024-03-30 19:41:21.903596	2024-03-30 19:41:21.903596	false

ii. Update

```
UPDATE public.company
    SET  name='New Ned'
    WHERE id = 7;
```

Data Output | Messages | Notifications

```
UPDATE 1

Query returned successfully in 65 msec.
```

iii. Delete

```
DELETE FROM public.company
    WHERE id = 7;
```

Data Output | Messages | Notifications

```
DELETE 1

Query returned successfully in 80 msec.
```

b. Category Table

i. Select

```
SELECT id, name, description, company_id, created_user, created_at, updated_at, is_delete
    FROM public.category;
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (255)	description text	company_id integer	created_user integer	created_at timestamp without time zone	updated_at timestamp without time zone	is_delete boolean
1	2	categor1	asd	9	2	2024-04-07 16:12:53.147783	2024-04-07 16:12:53.147783	false
2	3	electronics	electronic devices	9	2	2024-04-07 16:13:24.39804	2024-04-07 16:13:24.39804	false
3	1	123	123	9	2	2024-04-07 08:04:07.746572	2024-04-07 08:04:07.746572	true
4	7	Clothing	This is Clothing Category	13	1	2024-07-02 23:49:10.687678	2024-07-02 23:49:10.687678	false

ii. Update

```
UPDATE public.category
    SET name='New Clothing'
    WHERE id =7 ;
```

Data Output | Messages | Notifications

```
UPDATE 1

Query returned successfully in 89 msec.
```

iii. Delete

```
1  DELETE FROM public.category
2      WHERE id=7;
```

Data Output | Messages | Notifications

```
DELETE 1

Query returned successfully in 238 msec.
```

c. Sub Category Table

i. Select

```
1  SELECT id, name, description, created_user, company_id, category, created_at, updated_at, is_delete
2      FROM public.sub_category;
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (255)	description text	created_user integer	company_id integer	category integer	created_at timestamp without time zone	updated_at timestamp without time zone	is_delete boolean
1	5	television	led	2	9	3	2024-04-16 22:33:27.278726	2024-04-16 22:33:27.278726	false
2	7	mens watches	apple watches	2	9	3	2024-04-16 23:23:45.689545	2024-04-16 23:23:45.689545	true
3	6	watches	smart watche	2	9	3	2024-04-16 23:18:43.69134	2024-04-16 23:18:43.69134	true
4	8	watches	smart watch	2	9	3	2024-04-16 23:54:51.791614	2024-04-16 23:54:51.791614	false

ii. Update

```
1  UPDATE public.sub_category
2      SET  name='Women Clothing'
3      WHERE id = 8;
```

Data Output | Messages | Notifications

```
UPDATE 1

Query returned successfully in 125 msec.
```

iii. Delete

```sql
DELETE FROM public.sub_category
    WHERE id = 8;
```

Data Output | Messages | Notifications

```
DELETE 1

Query returned successfully in 258 msec.
```

d. Product Table

i. Select

```sql
SELECT product.name, product.description , product.barcode , product.selling_price AS price ,
product.unit, product.sku, product.image AS file,
    UPPER(category.name) AS category,
    UPPER(sub_category.name) AS sub_category,
    UPPER(manufacture.name) AS manufacture
    FROM product
    JOIN category ON category.id = product.category_id
    JOIN sub_category ON sub_category.id = product.sub_category_id
    JOIN manufacture ON manufacture.id = product.manufacture_id
    WHERE product.company_id = 9
    AND product.id =1
```

Data Output | Messages | Notifications

	name character varying (255)	description text	barcode character varying (100)	price numeric (20,2)	unit character varying (100)	sku character varying (100)	file character varying (255)	category text	sub_category text	manufacture text
1	oled display	Very Good	100020	100.00	100	PKKAS	public/uploads/product/football_hhj9c.png	ELECTRONICS	TELEVISION	SAMSUNG

ii. Update

```sql
UPDATE public.product
    SET  name='OLED Dispaly TV'
    WHERE id = 1;
```

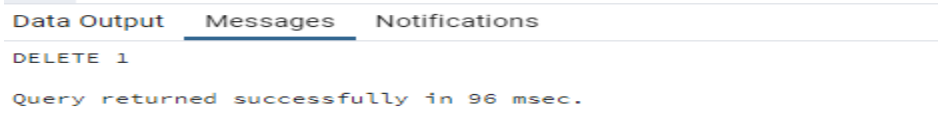
Data Output | Messages | Notifications

```
UPDATE 1

Query returned successfully in 113 msec.
```

iii. Delete

```
1  DELETE FROM public.product
2      WHERE id = 1;
```



7. Documentation

The Database consists of following tables that store data accordingly. Here is an overview of schema:

1. **Category**

- PK CategoryId
- CategoryName
- Description
- CompanyId (fk)

2. **SubCategory**

- PK subCategoryId
- categoryId
- subCategoryName
- companyId
- description

3. **Manufacturer**

- PK manufacturerId
- name

- description
- image
- companyId (fk)

4. **Product**

- PK productId
- categoryId (fk)
- subCategoryId (fk)
- manufacturerId (fk)
- name
- barCode
- sellingPrice
- SKU
- Unit
- image
- description
- companyId (fk)

5. **Sales**

- PK salesId
- salesDate
- totalPrice
- companyId (fk)
- userId (fk)

6. **SalesItem**

- PK salesItemId
- productId (fk)
- salesId (fk)
- quantity
- price
- companyId (fk)

7. **Company**

- PK companyId
- name

8. **Users**

- PK userId
- companyId (fk)
- name
- email
- password
- role

Relationship Between Entities:

The relationships mentioned in the document are as follows:

- **Category** has a one-to-many (1..*) relationship with **SubCategory** (One Category can have many SubCategories).
- **SubCategory** has a many-to-one (*..1) relationship with **Category** (Many SubCategories belong to one Category).
- **Manufacturer** has a one-to-many (1..*) relationship with **Product** (One Manufacturer can produce many Products).
- **Product** has a many-to-one (*..1) relationship with **Category** (Many Products belong to one Category).
- **Product** has a many-to-one (*..1) relationship with **SubCategory** (Many Products belong to one SubCategory).
- **Product** has a many-to-one (*..1) relationship with **Manufacturer** (Many Products are produced by one Manufacturer).
- **Sales** has a one-to-many (1..*) relationship with **SalesItem** (One Sale can have many SalesItems).
- **SalesItem** has a many-to-one (*..1) relationship with **Product** (Many SalesItems can refer to one Product).
- **SalesItem** has a many-to-one (*..1) relationship with **Sales** (Many SalesItems belong to one Sale).
- **Company** has a one-to-many (1..*) relationship with **Category** (One Company can have many Categories).
- **Company** has a one-to-many (1..*) relationship with **SubCategory** (One Company can have many SubCategories).
- **Company** has a one-to-many (1..*) relationship with **Manufacturer** (One Company can have many Manufacturers).

- **Company** has a one-to-many (1..*) relationship with **Product** (One Company can have many Products).
- **Company** has a one-to-many (1..*) relationship with **Sales** (One Company can have many Sales).
- **Company** has a one-to-many (1..*) relationship with **SalesItem** (One Company can have many SalesItems).
- **Company** has a one-to-many (1..*) relationship with **Users** (One Company can have many Users).