How to work with tables

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

Dominik Egarter
Data Engineering Enthusiast

General SQL statements

- Create a table
- Insert data into a table
- Update fields in a table
- Drop a table
- Delete the content of a table
- Change the structure of a table

Creating a table

General structure:

```
CREATE TABLE Person(
    ID INT NOT NULL,
      Name CHAR(32)
     );
```

General data types (more information):

- INT representing numbers
- CHAR representing a string

[1] https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-2017

Insert and update a table

Inserting data:

```
INSERT INTO ___ VALUES (___, ___);
```

```
INSERT INTO Person VALUES ('1', 'Smith');
```

Updating data:

```
UPDATE ___
    SET ___ = ___
    WHERE ___ = ___;
```

```
UPDATE Person
    SET Name = 'Anderson'
    WHERE ID = 1;
```

Delete and drop a table

Delete the rows of a table:

```
DELETE FROM ___
    WHERE ___ = ___;
```

```
DELETE FROM Person
    WHERE ID = 1;
```

Drop a table:

```
DROP TABLE ___
```

```
DROP TABLE Person
```

Change a table structure

Add a column:

```
ALTER TABLE ___
    ADD ___;
```

```
ALTER TABLE Person
    ADD new;
```

Delete a column:

```
ALTER TABLE ___
    DROP COLUMN ___
```

```
ALTER TABLE Person
    DROP COLUMN old
```

Let's practice!

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

Working with relational data models

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

Dominik Egarter
Data Engineering Enthusiast

Basics about relational data models

> The relational database model is the most widely used database model, which is the standard in database development.

A relational data model consists of:

- Tables
- Attributes
- Relations
- Relational algebra

Tables and attributes

Properties:

- Every table has a name (e.g., **Personal_Data**)
- Each column describes an attribute (e.g., **ID, Name, Birthday**)
- Each row consists of data

ID	Name	Birthday
1	Adam Smith	1.3.1978
2	Anna Jones	23.8.1991
3	Paul Williams	2.5.1954
4	Jessica Anderson	2.5.1954

Create relations

A relation is created by:

- **primary key**
- **foreign key**

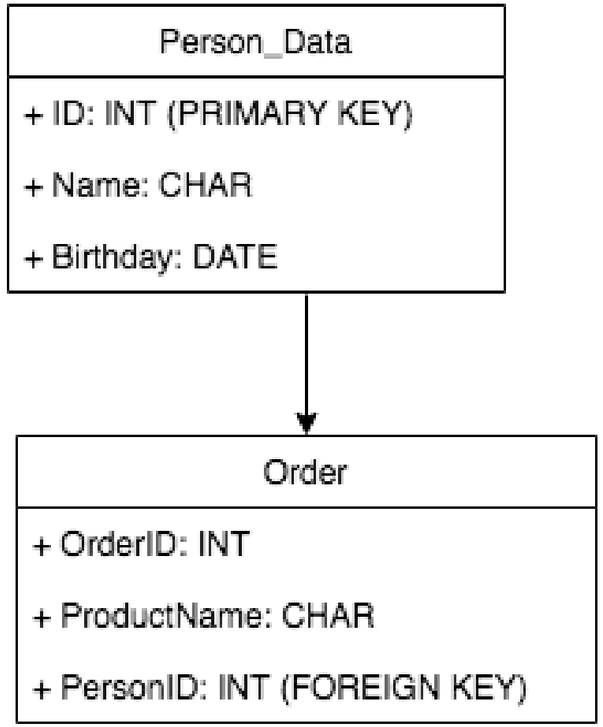
Properties of **primary keys**:

- unique
- each row has a primary key

Properties of **foreign keys**:

- primary key of another table

Example: Order history



Define primary and foreign keys

Primary key:

```
fieldName fieldType NOT NULL PRIMARY KEY,
```

e.g. for table `Person_Data` : `ID INT NOT NULL PRIMARY KEY`

Foreign key:

```
fieldName fieldType FOREIGN KEY REFERENCES tableName(primaryKey)
```

e.g. `newID INT FOREIGN KEY REFERENCES Person_Data(ID)`

Relational algebra

> Relational algebra is a formal language for relational databases and makes it possible to form a new relation from two or more relations.

Examples:

- SELECT
- UNION
- DIFFERENCE
- JOIN

Let's practice!

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

Working with hierarchical data models

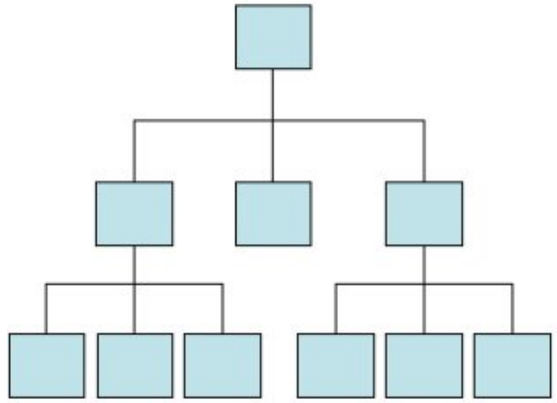
HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

Dominik Egarter
Data Engineering Enthusiast

The hierarchical data model

Properties of hierarchical data models:

- Represented as a tree structure
- Has one root element
- Each child record has one parent record



Advantages:

- Simple to understand
- Fast to select

Disadvantages:

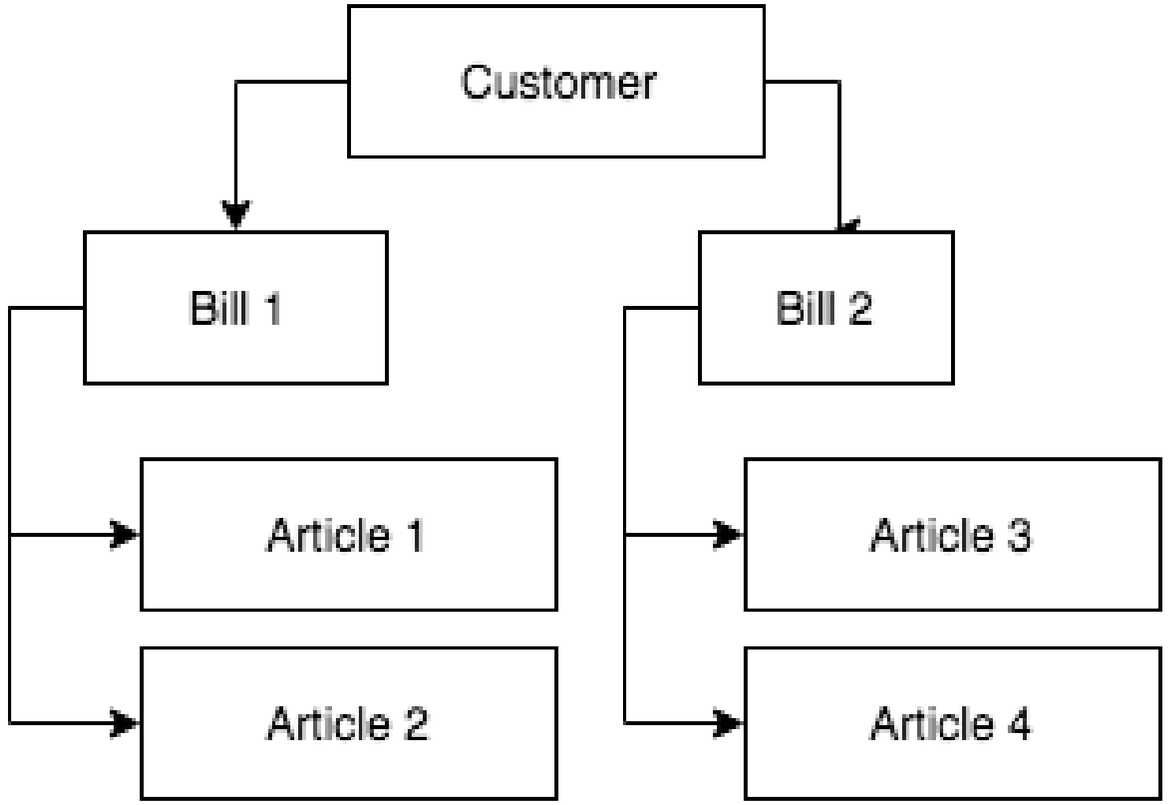
- Rigidly constructed
- Complicated to change structure

Example of hierarchical data model

Customer-bill-article relation:

> One customer can have several bills and each bill can have several articles

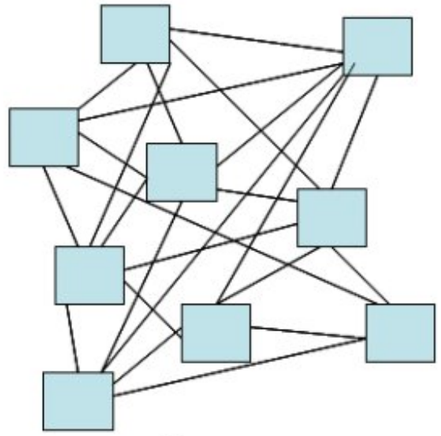
```
CREATE TABLE Customer (
    ID INT NOT NULL);
CREATE TABLE Bill (
    BillID INT NOT NULL,
      CustomerID INT);
CREATE TABLE Article (
    ArticleID INT NOT NULL,
      BillID INT);
```



The networked data model

Properties of networked data models:

- Similar to hierarchical data models
- many-to-many relation
- Many search paths exists



Advantages:

- No strict hierarchy
- Many solution paths
- Many real-world examples

Disadvantage:

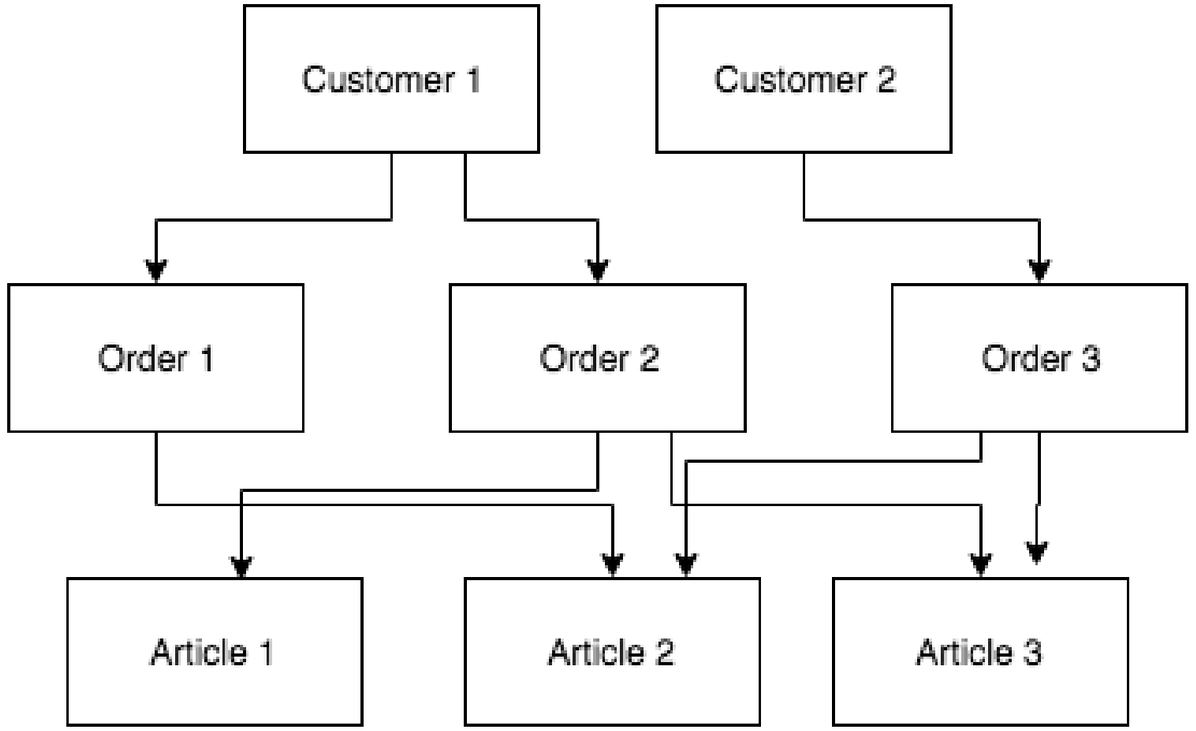
- Clarity decreases for large data models

Example of networked data models

Customer-order-article relation:

> Many customers can have several orders and each order can have several articles.

```
CREATE TABLE Customer (
    ID INT NOT NULL);
CREATE TABLE Order (
    OrderID INT NOT NULL,
      CustomerID INT);
CREATE TABLE Article (
    ArticleID INT NOT NULL,
      OrderID INT);
```



Let's practice!

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER