Example 1: Composite Relationship Between Two Tables

This example demonstrates how to model a **composite foreign key relationship** between two tables using Go struct annotations and a repository pattern.

📄 Tables: `Order` and `OrderItem`

🔗 `Order` Entity

```
type Order struct {
    _       DbField[any]    `db:"table(orders)"`
    OrderId DbField[uint64] `db:"primaryKey"`
    Version DbField[int]    `db:"primaryKey"`
    Note    DbField[string] `db:"length(200)"`
}
```

- Composite primary key: (`OrderId`, `Version`)

🔗 `OrderItem` Entity

```
type OrderItem struct {
    _        DbField[any]    `db:"table(order_items)"`
    Id       DbField[uint64] `db:"primaryKey;autoIncrement"`
    OrderId  DbField[uint64] `db:"index(order_ref_idx)"`
    Version  DbField[int]    `db:"index(order_ref_idx)"`
    Product  DbField[string] `db:"length(100)"`
    Quantity DbField[int]
}
```

- Composite index: (`OrderId`, `Version`) with shared name `order_ref_idx` to mark relationship.

💼 Repository Definition

```
type OrderRepository struct {
    TenantDb
    Orders     Queryable[Order]
    OrderItems Queryable[OrderItem]
}
```

🔦Declare Relationship in Init()

```
func (r *OrderRepository) Init() {
    r.NewRelationship().
        From(r.Orders.OrderId, r.Orders.Version).
        To(r.OrderItems.OrderId, r.OrderItems.Version)
}
```

📃Optional SQL Output (Manual or Dialect-based)

```
ALTER TABLE order_items
ADD CONSTRAINT fk_order_items_orders
FOREIGN KEY (order_id, version)
REFERENCES orders(order_id, version);
```

This would be generated automatically if `GenerateForeignKeyConstraintsSql()` is implemented.

🔗Summary

Table	Key	Description
`orders`	(`order_id`, `version`)	Composite primary key
`order_items`	(`order_id`, `version`)	Composite foreign key index

The `Init()` method is where relationships can be declared for automatic SQL generation, validation, or query planner use.