Entity and Mapping Translation

EFCore (C#)

```
[Table("Customers", Schema = "Sales")]
public class Customer {
    [Key]
    public int CustomerID { get; set; }
    public required string CustomerName { get; set; }
    public DateTime AccountOpenedDate { get; set; }
    public decimal? CreditLimit { get; set; }
    public List<CustomerTransaction> Transactions {
    get; set; } = []; }
```

```
[Table("CustomerTransactions", Schema = "Sales")]
public class CustomerTransaction {
    [Key]
    public int CustomerTransactionID { get; set; }
    [ForeignKey(nameof(Customer))]
    public int CustomerID { get; set; }
    public DateTime TransactionDate { get; set; }
    public decimal TransactionAmount { get; set; } }
```

AbstractEntityBuilder

```
.addSchema(Sales)
.addTable(Customers)
.addPrimaryKey(CustomerID, int)
.addProperty(CustomerName, string)
.addProperty(AccountOpenedDate, DateTime)
.addProperty(CreditLimit, decimal, optional)
.buildClass();
```

```
.addSchema(Sales)
.addTable(CustomersTransaction)
.addPrimaryKey(CustomerTransactionID, int)
.addForeignKey(CustomerID, int)
.addProperty(TransactionDate, DateTime)
.addProperty(TransactionAmount, decimal)
.buildClass();
```

Dapper (C#)
runtime: 182.58 ms
memory: 40,924.88 kB

```
public class Customer {
    public int CustomerID { get; set; }
    public required string CustomerName { get; set; }
    public DateTime AccountOpenedDate { get; set; }
    public decimal? CreditLimit { get; set; }
    public List<CustomerTransaction> Transactions {
    get; set; } = []; }
```

```
public class CustomerTransaction {
    public int CustomerTransactionID { get; set; }
    public int CustomerID { get; set; }
    public DateTime TransactionDate { get; set; }
    public decimal TransactionAmount { get; set; } }
```

NHibernate (C#)
runtime: TODO ms
memory: TODO kB

NHibernate entity

NHibernate mapping

NHibernate entity

NHibernate mapping

DBMS

DBMS

Customer	CustomerTransaction
CustomerID	CustomerTransactionID
CustomerName	CustomerID
AccountOpenedDate	TransactionDate
CreditLimit	TransactionAmount

DOPOCITAVA HODNOTY
PRO NHIBERNATE,
PROTOZE JE VSTUPNI
EFCORE NEUVAZUJE

Query Translation

EFCore (C#)
runtime: 2,474.37 ms
memory: 175,866.23 kB

```
using var ctx = contextFactory.CreateDbContext();
var result = ctx.Customers
    .Include(c => c.Transactions)
    .OrderBy(c => c.CustomerID)
    .ToList();
```

EFCore Q2

AbstractQueryBuilder

```
.from(Customers)
.join(Customers, CustomerTransactionID,
    LEFT_JOIN, 1.CustomerID=2.CustomerID)
.orderBy(Customers, {CustomerId}, ASC)
.project(*)
.build();
```

```
.from(Customers)
.join(Customers, CustomerTransactionID,
    LEFT_JOIN, 1.CustomerID=2.Custom...
.orderBy(Customers, {CustomerId}, ASC)
.project(*)
.build();
```

```
string sql = "SELECT c.*, ct.* FROM Customers c
LEFT JOIN CustomerTransactions ct
ON c.CustomerID= ct.CustomerID
ORDER BY c.CustomerID";

var customers = new Dictionary<int, Customer>();
connection.Query<Customer, CustomerTransaction,
Customer>(
    sql, (customer, transaction) => {
        if (!customers.TryGetValue(customer.CustomerID,
        out var existing)) {
            existing = customer;
            customers.Add(customer.CustomerID, existing); }
```

Dapper Q2

NHibernate Q1

NHibernate Q2

What-if analysis

EFCore (C#) Q1
runtime: 2,474.37 ms
memory: 175,866.23 kB

```
using var ctx = contextFactory.CreateDbContext();
var result = ctx.Customers
    .Include(c => c.Transactions)
    .OrderBy(c => c.CustomerID)
    .ToList();
```

EFCore (C#) Q2
runtime: 2,474.37 ms
memory: 175,866.23 kB

EFCore Q2

Dapper (C#) Q1
runtime: 2,474.37 ms
memory: 175,866.23 kB

```
string sql = "SELECT c.*, ct.* FROM Customers c
LEFT JOIN CustomerTransactions ct
ON c.CustomerID= ct.CustomerID
ORDER BY c.CustomerID";

var customers = new Dictionary<int, Customer>();
connection.Query<Customer, CustomerTransaction,
Customer>(
```

Dapper (C#) Q2
runtime: 2,474.37 ms
memory: 175,866.23 kB

Dapper Q2

```
var result = customers.Values.ToList(); }
```

NHibernate (C#) Q1
runtime: 2,474.37 ms
memory: 175,866.23 kB

NHibernate Q1 vše obalené do test suite

NHibernate (C#) Q2
runtime: 2,474.37 ms
memory: 175,866.23 kB
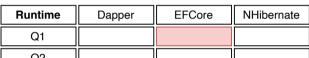
NHibernate Q2 obalené do testsuite

ORM Advisor

Constraints:
MAX_MEM= 128 MB
NUMBER OF FRAMEWORKS = 1

ILP Solver
(e.g., Gurobi)

ENUMERATION AND HIGHLIGHT WHEN
IT DOES NOT MEET CONSTRAINTS

Runtime	Dapper	EFCore	NHibernate
Q1			
Q2			

Memory	Dapper	EFCore	NHibernate
Q1			
Q2			
SUM			

Dapper (C#)
runtime: 182.58 ms
memory: 40,924.88 kB