1

Primary keys are in **bold** font.

- **branch**: `b_name | b_city | assets`
- **loan-branch**: `b_name | l_name`
- **loan**: `l_num | amount`
- **payment**: `l_num | p_num | p_date | p_amount`
- **borrower**: `l_num | c_id`
- **customer**: `c_id | c_name | c_street | c_city`
- **customer-banker**: `c_id | e_id | type`
- **employee**: `e_id | e_name | tel_num | start_date`
- **dependant**: `e_id | d_name`
works-for [manager_id, worker_id]

depositor [c_id, a_num, access_date]

account [a_num, balance]

savings-account [a_num, interest_rate]

checking-account [a_num, overdraft_amount]

2

1. Relational Calculus:

{\{t|\exists s \in Seller(s[Name] = t[Name] \land s[FeedbackRating] = 100\% \land s[Country] = "Denmark"\}}

Datalog:

\[ query(N) : \neg \text{seller}(I, N, E, 100, "Denmark"). \]

\[ \text{?query}(N). \]

2. Relational Algebra:

\[ r1 \leftarrow \sigma_{\text{Category Name} = "Oil Painting"}(Seller \bowtie Sell \bowtie Item \bowtie Belong) \]

\[ r2 \leftarrow \sigma_{\text{Category Name} \neq "Oil Painting"}(Seller \bowtie Sell \bowtie Item \bowtie Belong) \]

\[ \text{result} \leftarrow \pi_{\text{Seller ID}}(r1 - r2) \]

3. Relational Algebra:

\[ PaymentOption \div \pi_{\text{Option}}(PaymentOption) \]

Relational Calculus:

\[ \{t|\exists r \in Seller(r[\text{Seller ID}] = t[\text{Seller ID}] \land \]
\((\forall u \in \text{PaymentOption} \Rightarrow \\
\exists s \in \text{PaymentOption}(t[SellerID] = s[SellerID] \land u[\text{option}] = s[\text{option}]))\)

4. Relational Algebra:

\[ r \leftarrow \sigma_{\text{CategoryName} = "Oil Painting"}(\text{Seller} \bowtie \text{Sell} \bowtie \text{Belong}) \]

\[ \text{result} \leftarrow \pi_{\text{SellerID}}(\sigma_r.\text{InitialPrice} > x.\text{InitialPrice} (\rho_x(\sigma_{\text{Country} = "Denmark"}(r)) \times r)) \]

SQL:

```sql
SELECT SELL.SELLERID
FROM SELL, ITEM, BELONG
WHERE SELL.ITEMID = ITEM.ITEMID AND BELONG.ITEMID = ITEM.ITEMID AND
    BELONG.CATEGORYNAME = "Oil Painting" AND ITEM.INITIALPRICE > SOME
    (SELECT ITEM.INITIALPRICE
     FROM ITEM, BELONG, SELL
     WHERE ITEM.ITEMID = BELONG.ITEMID AND SELL.SELLERID = SELL.SELLERID AND SEL.L.ITEMID = ITEM.ITEMID
     AND
     BELONG.CATEGORYNAME = "Oil Painting" AND SELL.COUNTRY = "Denmark")
```

5. SQL:

```sql
SELECT BID.BIDDERID, ITEM.ITEMID, MAX(ITEM.BIDVALUE)
FROM BID, ITEM, BELONG
WHERE BID.ITEMID = ITEM.ITEMID AND ITEM.ITEMID = BELONG.ITEMID AND
    BELONG.CATEGORYNAME = "Oil Painting"
GROUP BY ITEM.ITEMID
```

QBE:

<table>
<thead>
<tr>
<th>Belong</th>
<th>ItemID</th>
<th>CategoryName</th>
<th>Bid</th>
<th>ItemID</th>
<th>BidValue</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>&quot;Oil Painting&quot;</td>
<td>P</td>
<td>P.x</td>
<td>P.x - y</td>
</tr>
</tbody>
</table>

6. QBE:

<table>
<thead>
<tr>
<th>Bidder</th>
<th>BidderID</th>
<th>Name</th>
<th>Bid</th>
<th>BidderID</th>
<th>ItemID</th>
<th>BidValue</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>_b</td>
<td>&quot;Jane Wu&quot;</td>
<td>_b</td>
<td>_b</td>
<td>_i</td>
<td>P.x2 - y2</td>
</tr>
</tbody>
</table>

7. Datalog:

\[
\text{hierarchy}(C, S) : \neg \text{subcategory}(C, S).
\]

\[
\text{hierarchy}(C, S) : \neg \text{hierarchy}(C, X), \text{subcategory}(X, S).
\]

\[
?\text{hierarchy}("Art", Y).
\]
Relational Algebra:
This query cannot be performed in relational algebra as it would involve an unbounded number of joins in order to find paths of any length. A recursive operator would be needed, but such an operator does not belong to the "classic" relational algebra we studied.