APPENDIX

A1: Statistics of the generated instances (Table : NumOfRows)

20K Products :	
Vendor : 203;	Product : 20, 000
Product1 : 10, 000;	Product2 : 10, 000
Person : 10210;	Review : 200000; ReviewC : 200, 000
ProductFeature : 10, 519;	ProductFeatureProduct : 423, 222
ProductFeatureProduct1 : 212, 342;	ProductFeatureProduct2 : 210, 880
ProductType : 329;	ProductTypeProduct : 80, 000
ProductTypeProduct1 : 40, 000;	ProductTypeProduct2 : 40, 000
Offer : 400, 000;	Producer : 405

200K Prooducts :	
Vendor : 2, 027;	Product : 200, 000
Product1 : 100, 000;	Product2 : 100, 000
Person : 102, 596;	Review : 2, 000, 000; ReviewC : 2, 000, 000
ProductFeature : 47, 884;	ProductFeatureProduct : 3, 885, 664
ProductFeatureProduct1 : 1, 941, 862;	ProductFeatureProduct2 : 1, 943, 802
ProductType : 2, 011;	ProductTypeProduct : 1, 000, 000
ProductTypeProduct1 : 500, 000;	ProductTypeProduct2 : 500, 000
Offer : 400, 000;	Producer : 3, 956

2M Products :	
Vendor : 19, 969;	Product : 2, 000, 000
Product1 : 1, 000, 000;	Product2 : 1, 000, 000
Person : 1, 024, 622;	Review : 1, 9999, 800; ReviewC : 19, 999, 800
ProductFeature : 94, 259;	ProductFeatureProduct : 3, 8571, 526
ProductFeatureProduct1 : 19, 285, 093;	ProductFeatureProduct2 : 19, 286, 433
ProductType : 3, 949;	ProductTypeProduct : 10, 000, 000
ProductTypeProduct1 : 5, 000, 000;	ProductTypeProduct2 : 5, 000, 000
Offer : 40, 000, 000;	Producer : 39, 530

A2: Pre-computed hints

Next we show the source hints used for the experiment. The top block contains the empty joins hints, the middle block contains the data redundancy hints, and the bottom one contains the materialized views. Here, T_i denotes T is a table/relation in the VDB $\mathbb{\Sigma}$ for data source S_i, and $\mathbb{D} \in \{s, m, l\}$ denotes one of the three instances considered in Appendix A1.

$\mathsf{EFJ1} : \mathsf{Product1}_1 \bowtie_{\mathsf{Product1}_1.nr=\mathsf{Product2}_2.nr} \mathsf{Product2}_2 =_{\mathbb{D}} \emptyset$
$\mathsf{EFJ2} : \mathsf{ProductFeatureProduct1}_1 \bowtie_{\mathsf{ProductFeatureProduct1}_1.product\mathsf{Product2}_2.nr} \mathsf{Product2}_2 =_{\mathbb{D}} \emptyset$
$\mathsf{EFJ3} : \mathsf{Product1}_1 \bowtie_{\mathsf{Product1}_1.nr\mathsf{ProductFeatureProduct2}_1.product} \mathsf{ProductFeatureProduct2}_1 =_{\mathbb{D}} \emptyset$
$\mathsf{DRS} : \mathsf{ReviewC}_1(\ldots) \equiv_{\mathbb{D}} \mathsf{Review}_5(\ldots)$

$op1(\ldots) \leftarrow \pi_{\ldots}(\mathsf{Offer}_4 \bowtie_{\mathsf{Offer}_4.product=\mathsf{Product1}_1.nr} \mathsf{Product1}_1)$
$op2(\ldots) \leftarrow \pi_{\ldots}(\mathsf{Offer}_4 \bowtie_{\mathsf{Offer}_4.product=\mathsf{Product2}_2.nr} \mathsf{Product2}_2)$
$pfpf1(\ldots) \leftarrow \pi_{\ldots}(\mathsf{ProductFeature}_3 \bowtie_{\mathsf{ProductFeature}_3.nr=\mathsf{ProductFeatureProduct1}_1.productfeature} \mathsf{ProductFeatureProduct1}_1)$
$pfpf2(\ldots) \leftarrow \pi_{\ldots}(\mathsf{ProductFeature}_3 \bowtie_{\mathsf{ProductFeature}_3.nr=\mathsf{ProductFeatureProduct2}_2.productfeature} \mathsf{ProductFeatureProduct2}_2)$
$ppd1(\ldots) \leftarrow \pi_{\ldots}(\mathsf{Product1}_1 \bowtie_{\mathsf{Product1}_1.nr=\mathsf{Producer}_4.product} \mathsf{Producer}_4)$
$ppd2(\ldots) \leftarrow \pi_{\ldots}(\mathsf{Product2}_2 \bowtie_{\mathsf{Product2}_2.nr=\mathsf{Producer}_4.product} \mathsf{Producer}_4)$

The statics for the materialized views, over each of the three data instances, are listed in the table below:

20K Products :		200K Prooducts :		2M Products :	
op1 : 200, 067;	*op2* : 199, 933	*op1* : 2, 001, 215;	*op2* : 1, 998, 785	*op1* : 20, 005, 677;	*op2* : 19, 994, 323
pfpf1 : 212, 342;	*pfpf2* : 210, 880	*pfpf1* : 1, 941, 862;	*pfpf2* : 1943802	*pfpf1* : 19, 285, 093;	*pfpf2* : 19, 286, 433
ppd1 : 10, 000;	*ppd2* : 10, 000	*ppd1* : 100, 000;	*ppd2* : 100, 000	*ppd1* : 1, 000, 000;	*ppd2* : 1, 000, 000

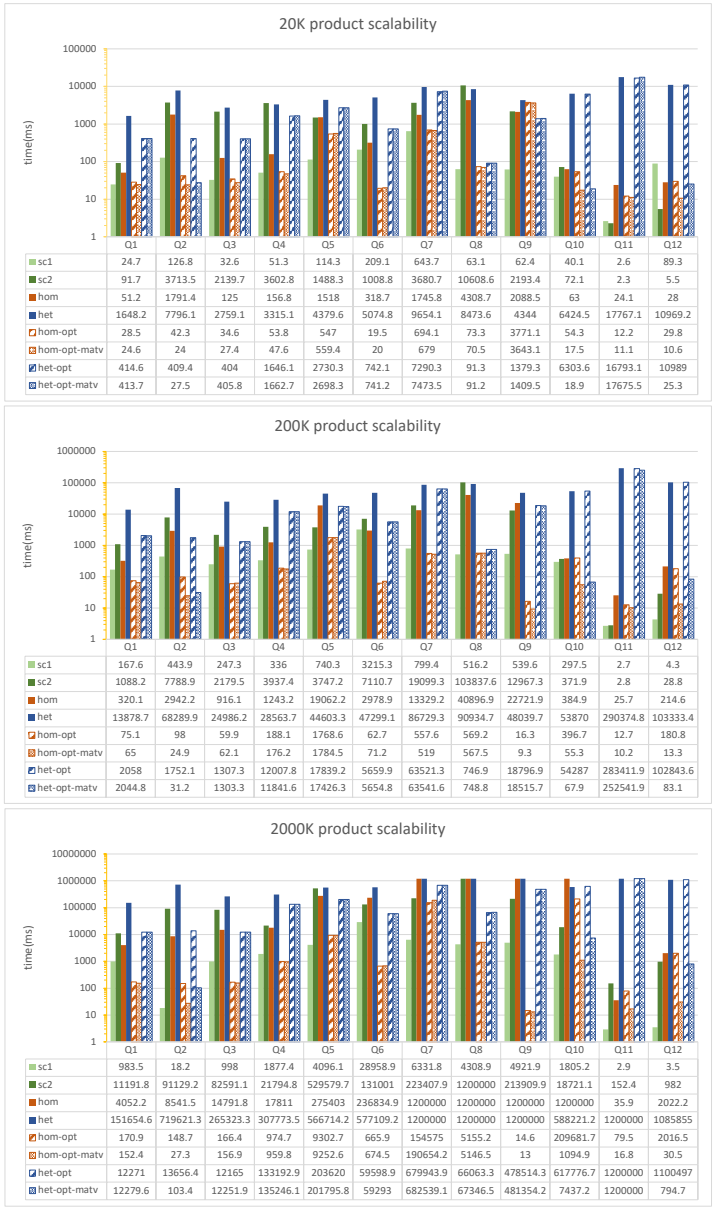
A3: Query evaluation

A4: Analysis of the experimental results

In this section, we analyze the impact hints-based optimizations over the queries in our experiment. Next, we use q_1–q_{12} to denote the translation from $Q1$–$Q12$ performed by Ontop (hence, with classical OBDA optimizations) in the federation situation.

Empty federated joins. The pre-computed empty-federated joins have an impact on queries q_1-q_6. For instance, Ontop translation q_3, containing 5 terminal federated joins (joins of unions) and 6 union operations, is optimized into a query without federated joins and a single union. Similarly, translation q_4, containing 8 terminal federated joins and 11 union operations, is optimized into a query without federated joins and 3 unions.

These optimizations pave the road to other optimizations that can be applied later, for instance w.r.t. materialized views. For instance, consider query q_2, containing 17 terminal federated joins and 14 union operations. Such translation is optimized into a query with 6 federated



20K product scalability

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
sc1	24.7	126.8	32.6	51.3	114.3	209.1	643.7	63.1	62.4	40.1	2.6	89.3
sc2	91.7	3713.5	2139.7	3602.8	1488.3	1008.8	3680.7	10608.6	2193.4	72.1	2.3	5.5
hom	51.2	1791.4	125	156.8	1518	318.7	1745.8	4308.7	2088.5	63	24.1	28
het	1648.2	7796.1	2759.1	3315.1	4379.6	5074.8	9654.1	8473.6	4344	6424.5	17767.1	10969.2
hom-opt	28.5	42.3	34.6	53.8	547	19.5	694.1	73.3	3771.1	54.3	12.2	29.8
hom-opt-matv	24.6	24	27.4	47.6	559.4	20	679	70.5	3643.1	17.5	11.1	10.6
het-opt	414.6	409.4	404	1646.1	2730.3	742.1	7290.3	91.3	1379.3	6303.6	16793.1	10989
het-opt-matv	413.7	27.5	405.8	1662.7	2698.3	741.2	7473.5	91.2	1409.5	18.9	17675.5	25.3

200K product scalability

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
sc1	167.6	443.9	247.3	336	740.3	3215.3	799.4	516.2	539.6	297.5	2.7	4.3
sc2	1088.2	7788.9	2179.5	3937.4	3747.2	7110.7	19099.3	103837.6	12967.3	371.9	2.8	28.8
hom	320.1	2942.2	916.1	1243.2	19062.2	2978.9	13329.2	40896.9	22721.9	384.9	25.7	214.6
het	13878.7	68289.9	24986.2	28563.7	44603.3	47299.1	86729.3	90934.7	48039.7	53870	290374.8	103333.4
hom-opt	75.1	98	59.9	188.1	1768.6	62.7	557.6	569.2	16.3	396.7	12.7	180.8
hom-opt-matv	65	24.9	62.1	176.2	1784.5	71.2	519	567.5	9.3	55.3	10.2	13.3
het-opt	2058	1752.1	1307.3	12007.8	17839.2	5659.9	63521.3	746.9	18796.9	54287	283411.9	102843.6
het-opt-matv	2044.8	31.2	1303.3	11841.6	17426.3	5654.8	63541.6	748.8	18515.7	67.9	252541.9	83.1

2000K product scalability

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
sc1	983.5	18.2	998	1877.4	4096.1	28958.9	6331.8	4308.9	4921.9	1805.2	2.9	3.5
sc2	11191.8	91129.2	82591.1	21794.8	529579.7	131001	223407.9	1200000	213909.9	18721.1	152.4	982
hom	4052.2	8541.5	14791.8	17811	275403	236834.9	1200000	1200000	1200000	1200000	35.9	2022.2
het	151654.6	719621.3	265323.3	307773.5	566714.2	577109.2	1200000	1200000	1200000	588221.2	1200000	1085855
hom-opt	170.9	148.7	166.4	974.7	9302.7	665.9	154575	5155.2	14.6	209681.7	79.5	2016.5
hom-opt-matv	152.4	27.3	156.9	959.8	9252.6	674.5	190654.2	5146.5	13	1094.9	16.8	30.5
het-opt	12271	13656.4	12165	133192.9	203620	59598.9	679943.9	66063.3	478514.3	617776.7	1200000	1100497
het-opt-matv	12279.6	103.4	12251.9	135246.1	201795.8	59293	682539.1	67346.5	481354.2	7437.2	1200000	794.7

Figure 6: The figure reports the results (time in ms, average of 10 executions) of the SQL evaluation, over the three data instances 20K, 200K and 2M products. The timeout for each execution is set to 1200 seconds.

joins and a single union. Such optimized query, that we here call q'_2, has the following structure:

$$(\mathrm{u}_1(\mathrm{ProductFeature}_3) \bowtie_{\theta_1} \mathrm{u}_2(\mathrm{ProductFeatureProduct}_1) \bowtie_{\theta_2} \mathrm{u}_3(\mathrm{Product1}_1) \bowtie_{\theta_3} \mathrm{u}_4(\mathrm{Producer}_4)) \cup$$
$$(\mathrm{v}_1(\mathrm{ProductFeature}_3) \bowtie_{\varphi_1} \mathrm{v}_2(\mathrm{ProductFeatureProduct2}_2) \bowtie_{\varphi_2} \mathrm{v}_3(\mathrm{Product2}_2) \bowtie_{\varphi_3} \mathrm{v}_4(\mathrm{Producer}_4))$$

We will see in the remainder of this section that this query can be further optimized by exploiting the pre-computed materialized views.

These optimizations lead to a considerable reduction of the query evaluation time. For the 200K instance and the hom configuration, the evaluation times of q_2, q_3 and q_4 go from 2942.2 ms, 916.1 ms and 1243.2 ms to 98 ms, 59.9 ms and 188.1 ms, respectively. Note that these executions are even faster than those for the "easiest" configuration, sc1.

Data redundancy. The pre-computed data redundancy hint DRS has an impact on translations q_6–q_9. After optimizing with respect to this hint, in fact, the evaluation of these three queries becomes as fast as for the sc1 situation. This improvement the hint allows for the

removal of several redundant operations. Consider, for instance, SQL query q_9, containing 5 terminal federated joins and 4 union operations. Optimizing such translation with respect to data redundancy leads to a query without federated joins and unions.

Materialized views. The pre-computed materialized views have an impact on translations q_2, q_{10} and q_{12}. For instance, after applying the optimizations based on empty federated joins hints, query q_2' above is further simplified into

$$q_2'' : (\mathsf{u}(S^{\mathsf{M}}.ppd1) \bowtie_{\theta} \mathsf{v}(S^{\mathsf{M}}.pfpf1)) \cup (\mathsf{u}'(S^{\mathsf{M}}.ppd2) \bowtie_{\varphi} \mathsf{v}'(S^{\mathsf{M}}.pfpf2))$$

Observe that, for such query, no computation of federated joins or access to inefficient sources is performed. This simplification results, for the 2M instance and for the hom and het settings, in a further reduction of the evaluation times respectively from 8541.5 ms and 719621.3 ms to 27.3 ms and 103.4 ms.

Query q_{10}, instead, is transformed from the following algebra expression:

$$((\mathsf{u}_1(\mathsf{Product1}_1) \bowtie_{\theta_1} \mathsf{v}_1(\mathsf{Offer}_4) \bowtie_{\varphi_1} \mathsf{w}_1(\mathsf{Vendor}_4)))\cup$$
$$((\mathsf{u}_2(\mathsf{Product2}_2) \bowtie_{\theta_2} \mathsf{v}_2(\mathsf{Offer}_4) \bowtie_{\varphi_2} \mathsf{w}_2(\mathsf{Vendor}_4)))$$

into the query

$$q_{10}'' : (\mathsf{u}(S^{\mathsf{M}}.op1) \bowtie_{\theta'} \mathsf{v}(\mathsf{Vendor}_4)) \cup (\mathsf{u}'(S^{\mathsf{M}}.op2) \bowtie_{\varphi'} \mathsf{v}'(\mathsf{Vendor}_4))$$

This simplification results, for the 2M instance and for the hom and het settings, in a reduction of the evaluation times respectively from 209681.7 ms and 617776.8 ms to 1094.9 ms and 7437.2 ms.