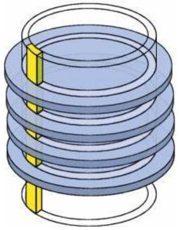
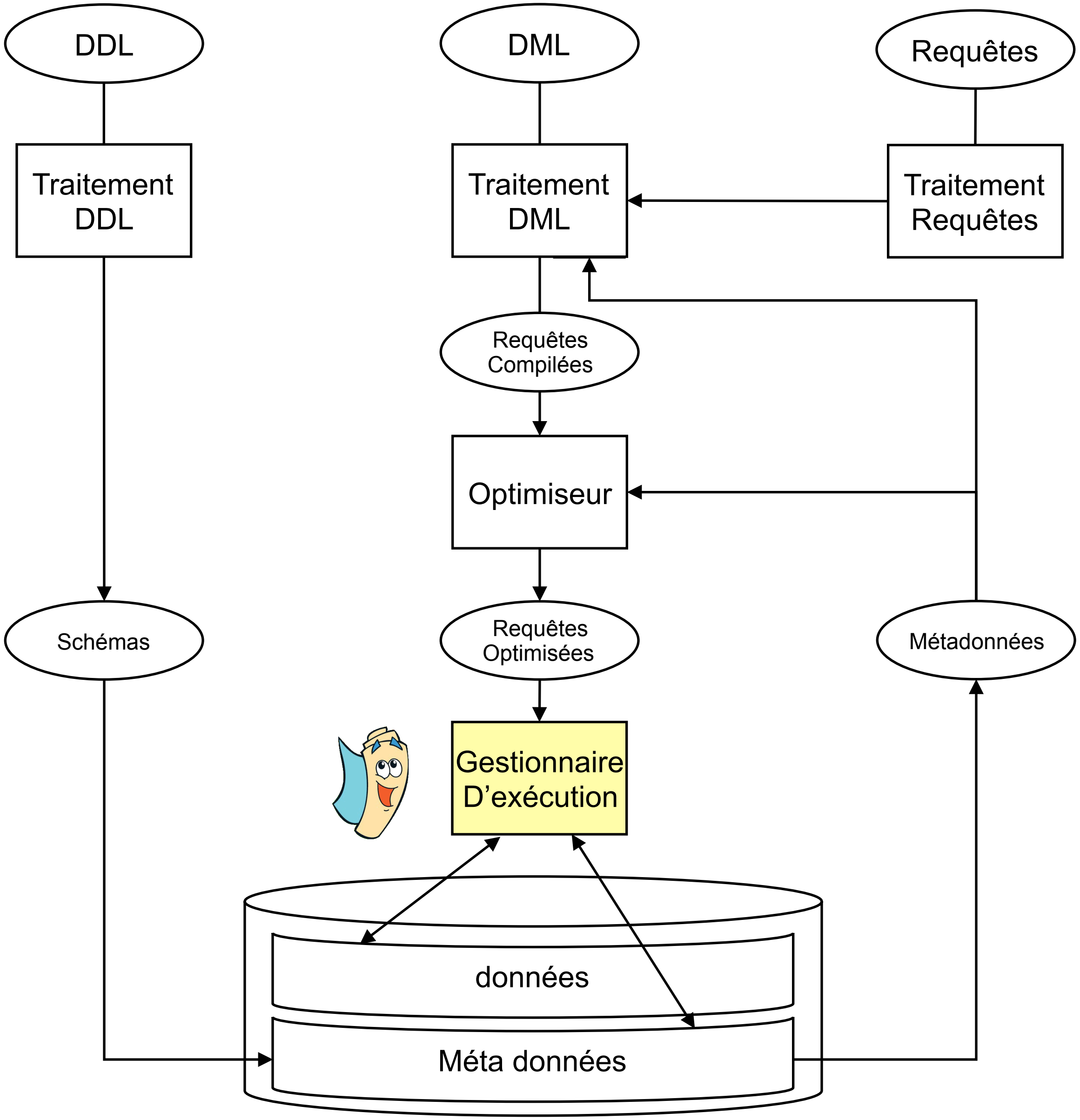
Conception Avancée de Bases de Données

Grace Hash Join

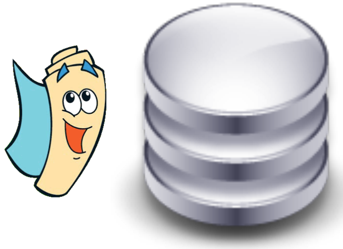


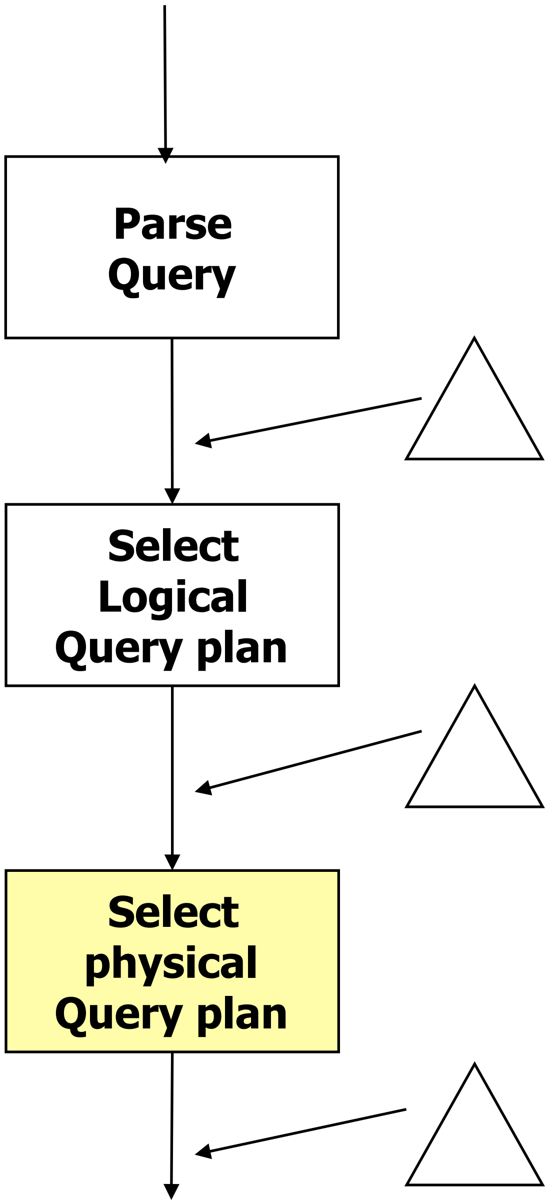


D'après C.J DATE

DDL : langage de définition des données; DML : langage de manipulation des données

From Ullman

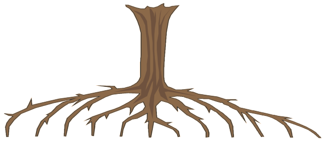




Query expression tree

Logical Query Plan tree

Physical Query Plan tree



Du modèle au code

Modèle

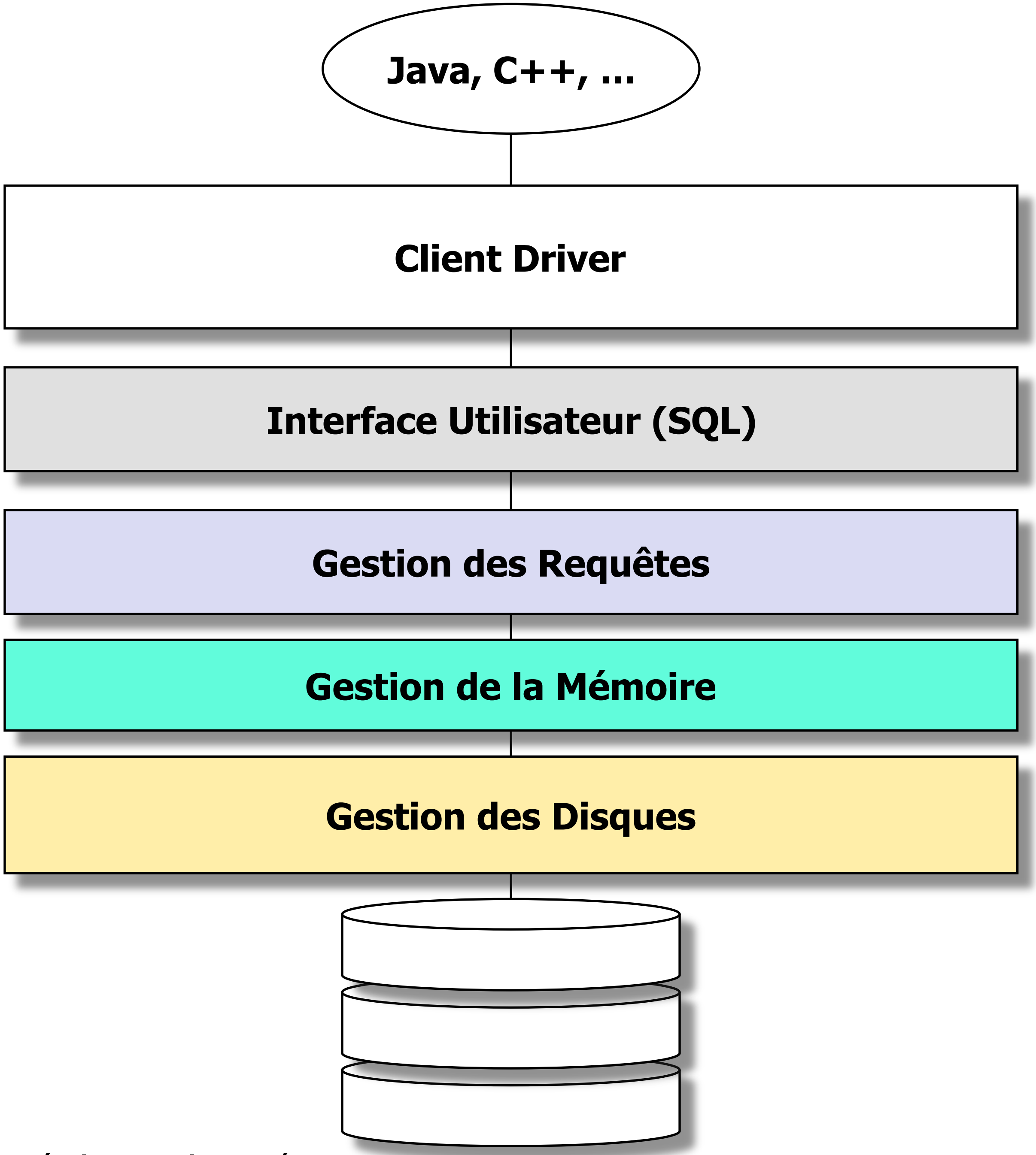
name1	owner1	name2	owner2
Fluffy	Harold	Buffy	Harold
Fluffy	Harold	Fang	Benny
Fluffy	Harold	Bowser	Diane
Claws	Gwen	Buffy	Harold
Claws	Gwen	Fang	Benny
Claws	Gwen	Bowser	Diane

Algèbre

σ owner1=owner2 (Cats $\otimes$ Dogs) = Cat $\bowtie$ Dogs

Logiciel

Java, C++,..



Java, C++, ...

Client Driver

Interface Utilisateur (SQL)

Gestion des Requêtes

Gestion de la Mémoire

Gestion des Disques

Grace Hash join

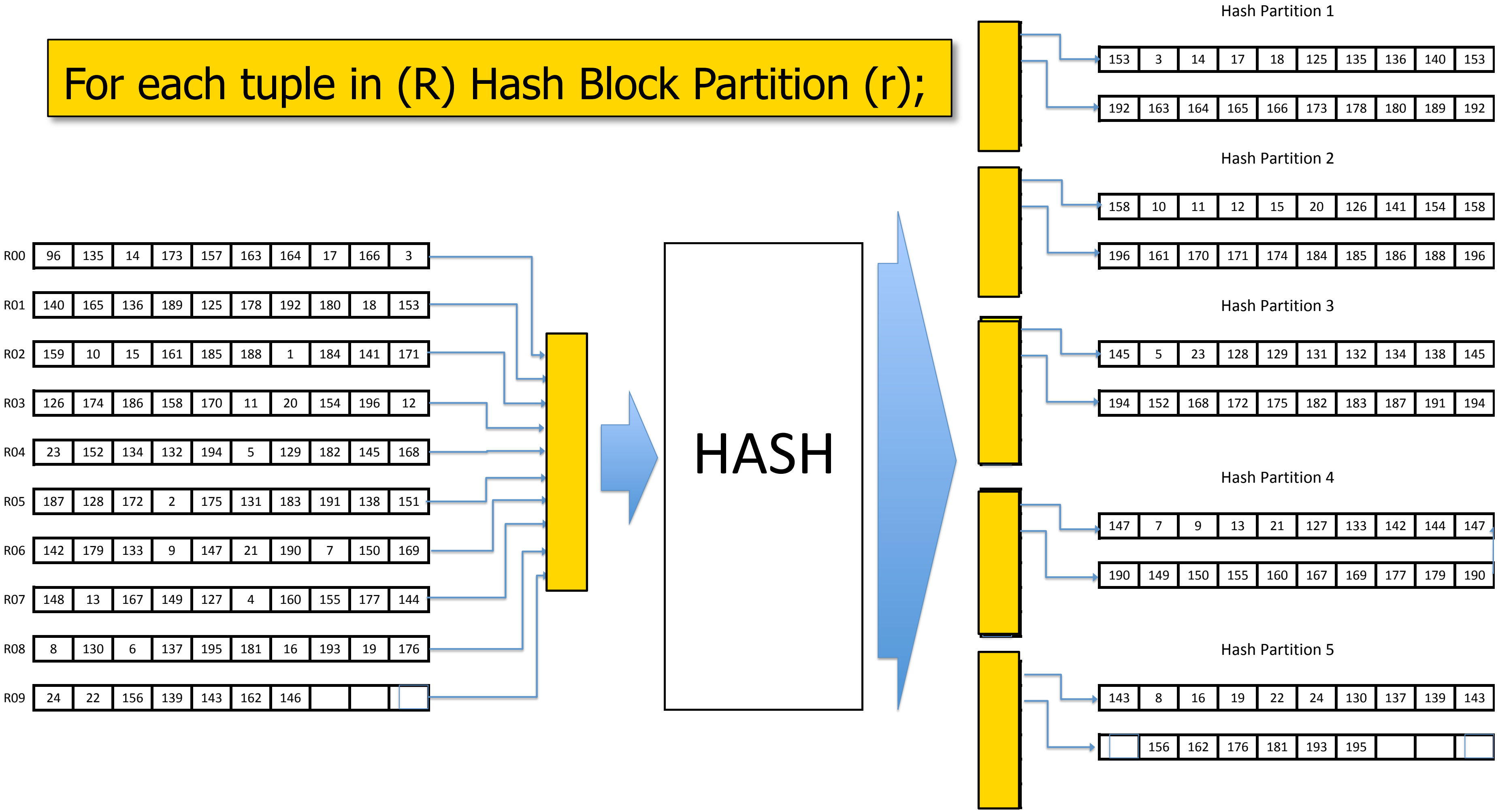
```
For each tuple in (R) Hash Block Partition (r);

For each tuple in (S) Hash Block Partition (s);

For each Hash Block Partition Number (Rhi) and (Shi)
        Nested Loop Join of Rhi and Shi
End for;
```

R Hash Block Partitioning

For each tuple in (R) Hash Block Partition (r);



R00	96	135	14	173	157	163	164	17	166	3
R01	140	165	136	189	125	178	192	180	18	153
R02	159	10	15	161	185	188	1	184	141	171
R03	126	174	186	158	170	11	20	154	196	12
R04	23	152	134	132	194	5	129	182	145	168
R05	187	128	172	2	175	131	183	191	138	151
R06	142	179	133	9	147	21	190	7	150	169
R07	148	13	167	149	127	4	160	155	177	144
R08	8	130	6	137	195	181	16	193	19	176
R09	24	22	156	139	143	162	146			

Hash Partition 1

153	3	14	17	18	125	135	136	140	153
192	163	164	165	166	173	178	180	189	192

Hash Partition 2

158	10	11	12	15	20	126	141	154	158
196	161	170	171	174	184	185	186	188	196

Hash Partition 3

145	5	23	128	129	131	132	134	138	145
194	152	168	172	175	182	183	187	191	194

Hash Partition 4

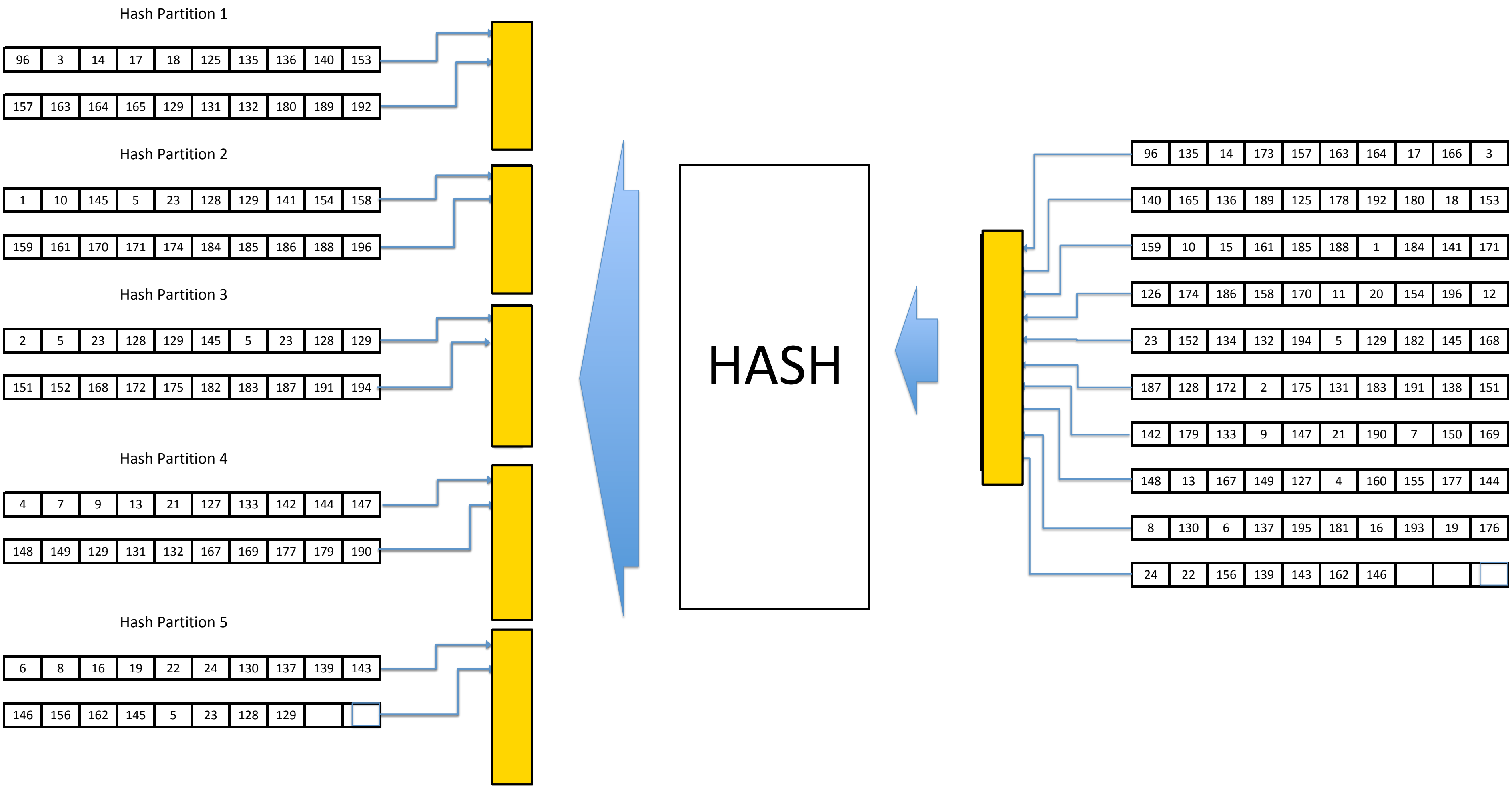
147	7	9	13	21	127	133	142	144	147
190	149	150	155	160	167	169	177	179	190

Hash Partition 5

143	8	16	19	22	24	130	137	139	143
	156	162	176	181	193	195			

S Hash Block Partitioning

For each tuple in (S) Hash Block Partition (s);



Hash Partition 1

96	3	14	17	18	125	135	136	140	153
157	163	164	165	129	131	132	180	189	192

Hash Partition 2

1	10	145	5	23	128	129	141	154	158
159	161	170	171	174	184	185	186	188	196

Hash Partition 3

2	5	23	128	129	145	5	23	128	129
151	152	168	172	175	182	183	187	191	194

Hash Partition 4

4	7	9	13	21	127	133	142	144	147
148	149	129	131	132	167	169	177	179	190

Hash Partition 5

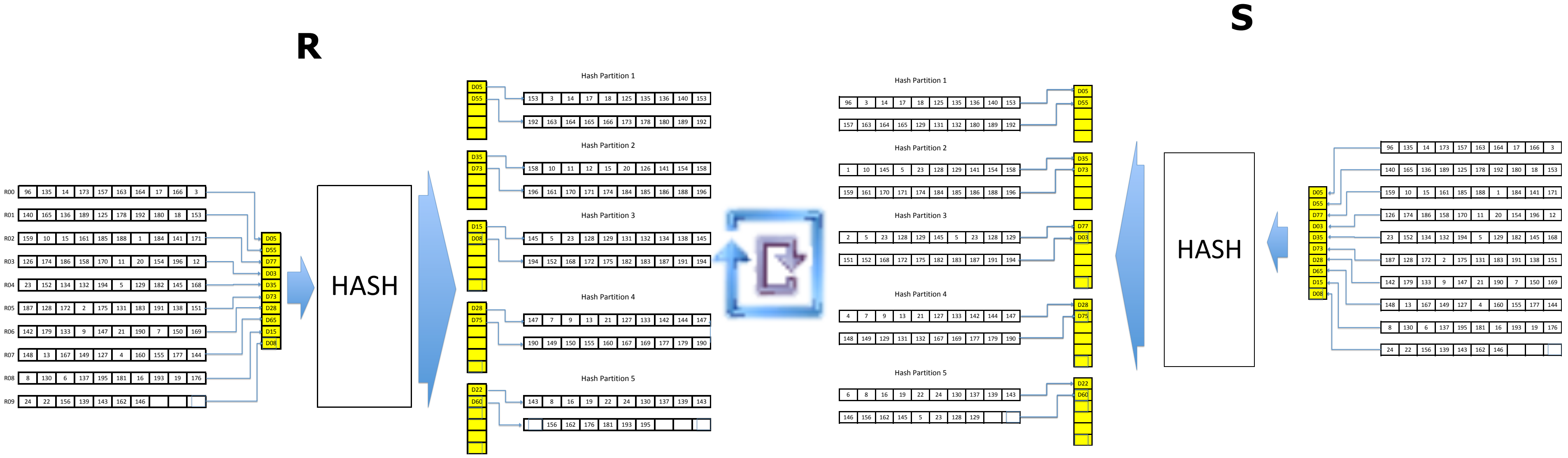
6	8	16	19	22	24	130	137	139	143
146	156	162	145	5	23	128	129		

HASH

96	135	14	173	157	163	164	17	166	3
140	165	136	189	125	178	192	180	18	153
159	10	15	161	185	188	1	184	141	171
126	174	186	158	170	11	20	154	196	12
23	152	134	132	194	5	129	182	145	168
187	128	172	2	175	131	183	191	138	151
142	179	133	9	147	21	190	7	150	169
148	13	167	149	127	4	160	155	177	144
8	130	6	137	195	181	16	193	19	176
24	22	156	139	143	162	146			

Emmanuel fuchs Conception Avancée de Bases de Données

R S Nested loop



R

R00	96	135	14	173	157	163	164	17	166	3
R01	140	165	136	189	125	178	192	180	18	153
R02	159	10	15	161	185	188	1	184	141	171
R03	126	174	186	158	170	11	20	154	196	12
R04	23	152	134	132	194	5	129	182	145	168
R05	187	128	172	2	175	131	183	191	138	151
R06	142	179	133	9	147	21	190	7	150	169
R07	148	13	167	149	127	4	160	155	177	144
R08	8	130	6	137	195	181	16	193	19	176
R09	24	22	156	139	143	162	146			

D05 D55 D77 D03 D35 D73 D28 D65 D15 D08

Hash Partition 1

D05	153	3	14	17	18	125	135	136	140	153
D55	192	163	164	165	166	173	178	180	189	192

Hash Partition 2

D35	158	10	11	12	15	20	126	141	154	158
D73	196	161	170	171	174	184	185	186	188	196

Hash Partition 3

D15	145	5	23	128	129	131	132	134	138	145
D08	194	152	168	172	175	182	183	187	191	194

Hash Partition 4

D28	147	7	9	13	21	127	133	142	144	147
D75	190	149	150	155	160	167	169	177	179	190

Hash Partition 5

D22	143	8	16	19	22	24	130	137	139	143
D60		156	162	176	181	193	195			

Hash Partition 1

96	3	14	17	18	125	135	136	140	153	D05
157	163	164	165	129	131	132	180	189	192	D55

Hash Partition 2

1	10	145	5	23	128	129	141	154	158	D35
159	161	170	171	174	184	185	186	188	196	D73

Hash Partition 3

2	5	23	128	129	145	5	23	128	129	D77
151	152	168	172	175	182	183	187	191	194	D03

Hash Partition 4

4	7	9	13	21	127	133	142	144	147	D28
148	149	129	131	132	167	169	177	179	190	D75

Hash Partition 5

6	8	16	19	22	24	130	137	139	143	D22
146	156	162	145	5	23	128	129			D60

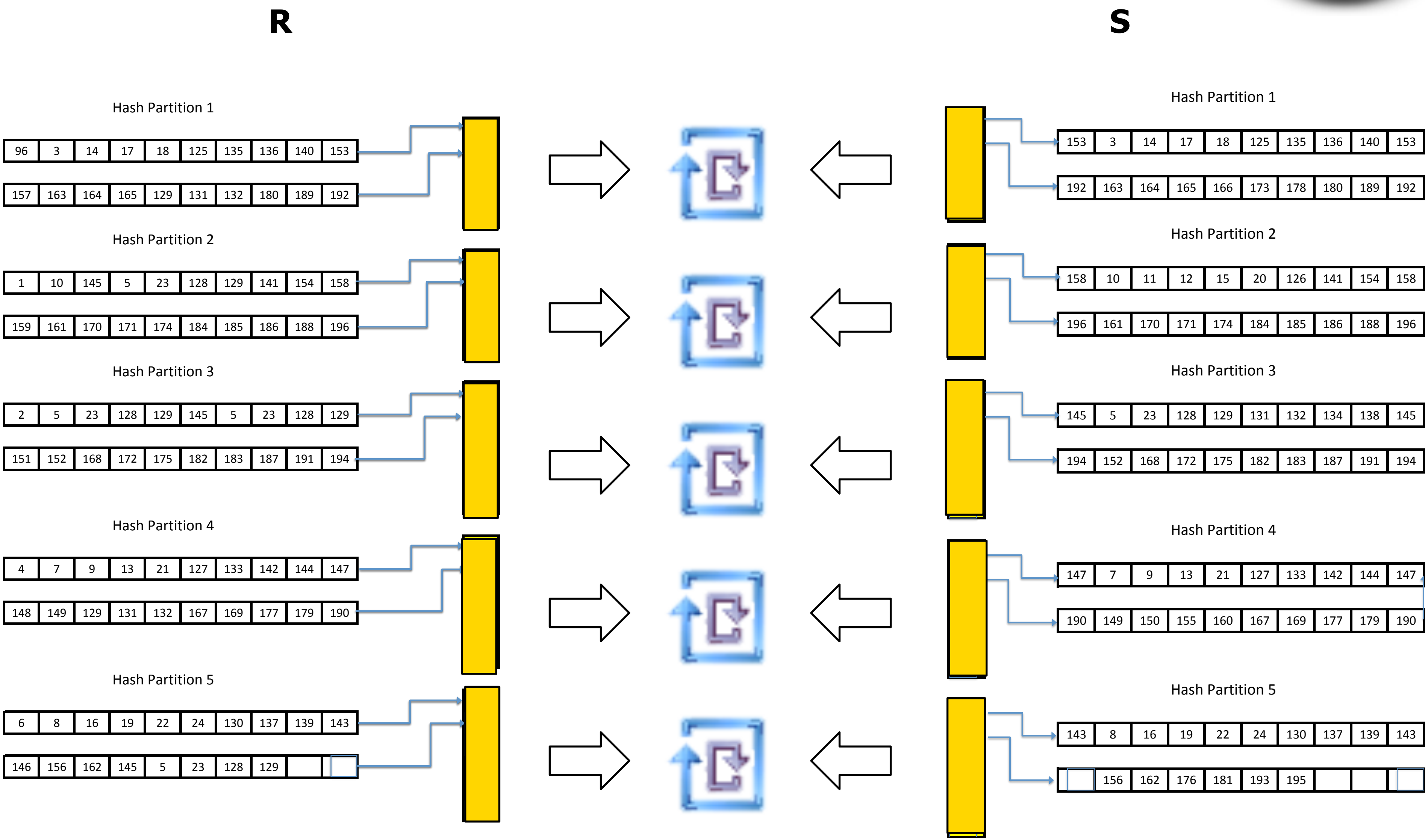
S

D05 D55 D77 D03 D35 D73 D28 D65 D15 D08

96	135	14	173	157	163	164	17	166	3
140	165	136	189	125	178	192	180	18	153
159	10	15	161	185	188	1	184	141	171
126	174	186	158	170	11	20	154	196	12
23	152	134	132	194	5	129	182	145	168
187	128	172	2	175	131	183	191	138	151
142	179	133	9	147	21	190	7	150	169
148	13	167	149	127	4	160	155	177	144
8	130	6	137	195	181	16	193	19	176
24	22	156	139	143	162	146			

```
For each Hash Block Partition Number (Rhi) and (Shi)
            Nested Loop Join of Rhi and Shi
End for;
```

R S



R

Hash Partition 1

96	3	14	17	18	125	135	136	140	153
157	163	164	165	129	131	132	180	189	192

Hash Partition 2

1	10	145	5	23	128	129	141	154	158
159	161	170	171	174	184	185	186	188	196

Hash Partition 3

2	5	23	128	129	145	5	23	128	129
151	152	168	172	175	182	183	187	191	194

Hash Partition 4

4	7	9	13	21	127	133	142	144	147
148	149	129	131	132	167	169	177	179	190

Hash Partition 5

6	8	16	19	22	24	130	137	139	143
146	156	162	145	5	23	128	129		

S

Hash Partition 1

153	3	14	17	18	125	135	136	140	153
192	163	164	165	166	173	178	180	189	192

Hash Partition 2

158	10	11	12	15	20	126	141	154	158
196	161	170	171	174	184	185	186	188	196

Hash Partition 3

145	5	23	128	129	131	132	134	138	145
194	152	168	172	175	182	183	187	191	194

Hash Partition 4

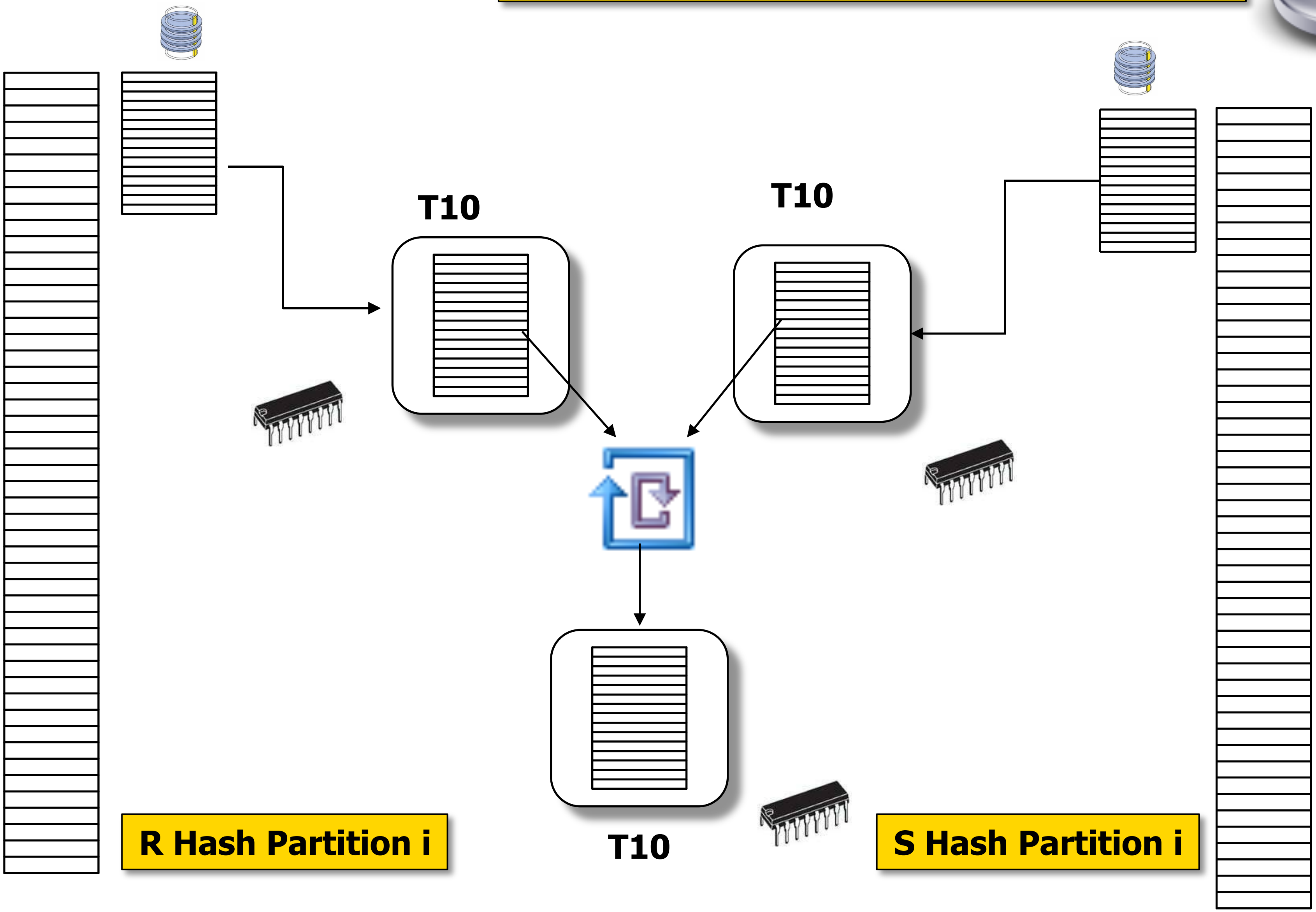
147	7	9	13	21	127	133	142	144	147
190	149	150	155	160	167	169	177	179	190

Hash Partition 5

143	8	16	19	22	24	130	137	139	143
	156	162	176	181	193	195			

Nested Loop Join

```
For each Hash Block Partition Number (Rhi) and (Shi)
        Nested Loop Join of Rhi and Shi
End for;
```



R Hash Partition i

S Hash Partition i