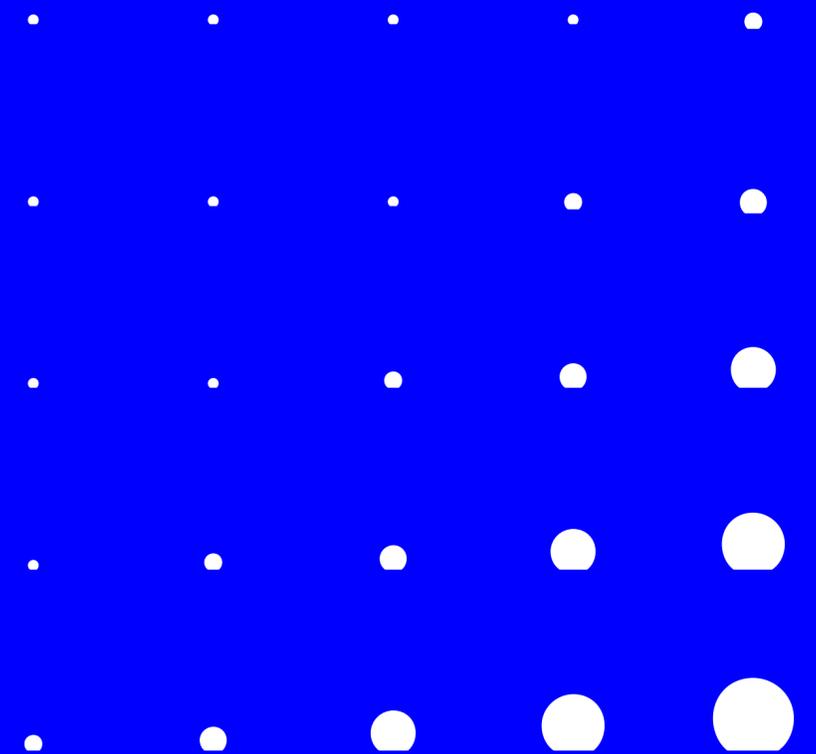


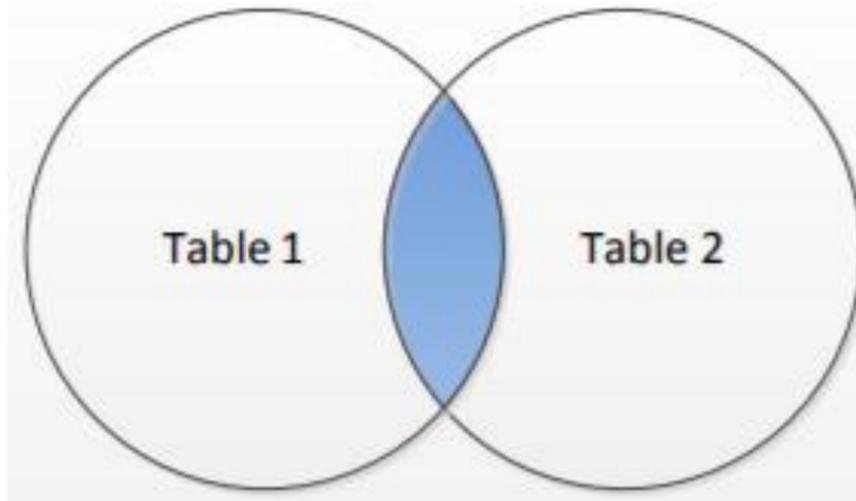
**SQL: DRL & Join**  
**Abfragen über mehrere Tabellen mittels JOIN**



# DRL – JOIN: Was ist ein Verbund?

- Um Redundanzen beim Speichern zu vermeiden, werden die Informationen auf verschiedene Tabellen verteilt.
  - Dies geschieht über den Vorgang der Normalisierung.
- Zur Erhaltung der logischen Zusammengehörigkeit werden dabei Fremdschlüssel-Beziehungen zwischen den Tabellen aufgebaut.
- Muss das Datenbanksystem nun eine Anfrage verarbeiten, bei der Informationen aus mehreren Tabellen benötigt werden, so muss man die einzelnen Datensätze der Tabellen wieder zusammenzuführen.
- Dadurch werden die ursprünglichen Informationen wiederhergestellt.
- Dies wird in SQL-Datenbanken mit Hilfe von SQL JOINS (Verbänden) umgesetzt.
- Mit einem SQL JOIN werden also mehrere Tabellen verknüpft, die in einer Beziehung zueinander stehen.

# DRL – INNER JOIN



```
SELECT *  
  FROM Table_1 t1  
 INNER JOIN Table_2 t2  
    ON t1.id = t2.fk;
```

# DRL – INNER JOIN: Beispiel

rechnungnummer	kundennummer	betrag	kartennummer
98765	ABX039	49.95	12345
98766	ABX039	12.95	NULL
98767	ABX040	79.95	12347
98768	ABX050	59.99	12347
98769	ABX050	29.99	12348
98770	ABX060	99.99	NULL

rechnungen

fk

kartennummer	firma	inhaber	ablaufdatum
12345	VISA	Max Mustermann	2017-05-01
12346	Mastercard	Katrin Musterfrau	2018-01-01
12347	American Express	John Doe	2015-02-01
12348	Diners Club	John Doe	2020-03-01

pk

kreditkarten

- Gesucht werden alle Rechnungen, die mit Kreditkarte beglichen wurden.  
Von diesen Kreditkarten hätte man auch gern die Daten:

**SELECT**

**r.rechnungnummer, r.kundennummer, r.betrag, r.kartennummer, k.firma, k.inhaber, k.ablaufdatum**

**FROM rechnungen r**

**INNER JOIN kreditkarten k**

**ON r.kartennummer = k.kartennummer;**

# DRL – INNER JOIN: Beispiel

rechnungnummer	kundennummer	betrag	kartennummer
98765	ABX039	49.95	12345
98766	ABX039	12.95	NULL
98767	ABX040	79.95	12347
98768	ABX050	59.99	12347
98769	ABX050	29.99	12348
98770	ABX060	99.99	NULL

rechnungen

fk

kartennummer	firma	inhaber	ablaufdatum
12345	VISA	Max Mustermann	2017-05-01
12346	Mastercard	Katrin Musterfrau	2018-01-01
12347	American Express	John Doe	2015-02-01
12348	Diners Club	John Doe	2020-03-01

pk

kreditkarten

- Gesucht werden alle Rechnungen, die mit Kreditkarte beglichen wurden. Von diesen Kreditkarten hätte man auch gern die Daten:

rechnungnummer	kundennummer	betrag	kartennummer	firma	inhaber	ablaufdatum
98765	ABX039	49.95	12345	VISA	Max Mustermann	2017-05-01
98767	ABX040	79.95	12347	American Express	John Doe	2015-02-01
98768	ABX050	59.99	12347	American Express	John Doe	2015-02-01
98769	ABX050	29.99	12348	Diners Club	John Doe	2020-03-01

## DRL – INNER JOIN: Beispiel

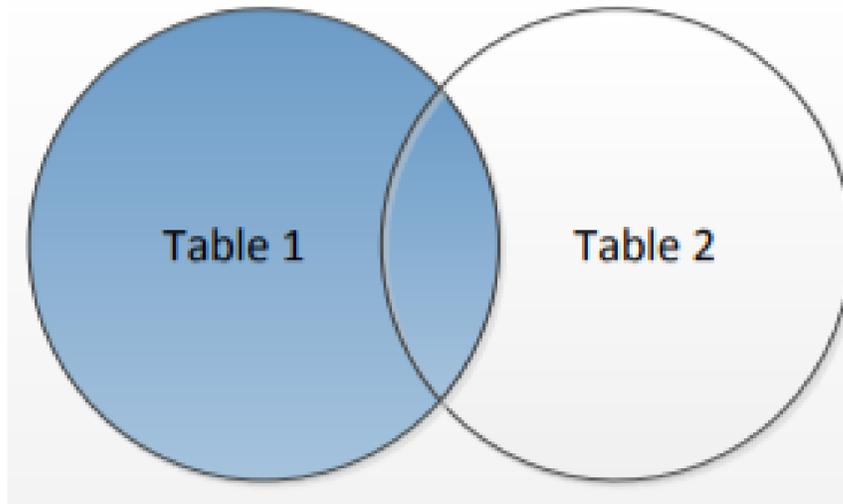
- Zunächst fällt auf, dass von ursprünglich 6 Rechnungen nun nur noch 4 Rechnungen übrig sind.
  - Der Grund liegt darin, dass zu den beiden Barzahlungseinträgen keine passende Kreditkarte gefunden werden konnte.
  - Damit ist das ON-Kriterium des Joins nicht erfüllt und der Datensatz entfällt.
- Analog dazu ist die Kreditkarte 12346 nicht im Ergebnis enthalten, da sie keiner Rechnung zugeordnet werden konnte.
- Es werden bei diesem INNER JOIN also
  - nicht unbedingt alle Rechnungen wiedergegeben und auch
  - nicht unbedingt alle Kreditkarten.

## DRL – INNER JOIN: Beispiel

- Die Spalte Kartenummer muss zusammen mit einem Tabellennamen angegeben werden, da sie durch den Join im Ergebnis doppelt vorhanden ist, einmal aus jeder Tabelle.
  - Das DBMS fordert daher zur Auflösung dieser Mehrdeutigkeit auf.
- Das obige Ergebnis stimmt zudem mit einer nicht-normalisierten Speicherung aller Kreditkartenzahlungen überein, allerdings mit den bekannten Redundanzen, hier z.B. die Kreditkartendaten der Kartenummer 12347...

12347	American Express	John Doe	2015-02-01
12347	American Express	John Doe	2015-02-01

# DRL – LEFT OUTER JOIN



```
SELECT *  
  FROM Table_1 t1  
 LEFT JOIN Table_2 t2  
   ON t1.id = t2.fk;
```

# DRL – LEFT OUTER JOIN: Beispiel

rechnungnummer	kundennummer	betrag	kartennummer
98765	ABX039	49.95	12345
98766	ABX039	12.95	NULL
98767	ABX040	79.95	12347
98768	ABX050	59.99	12347
98769	ABX050	29.99	12348
98770	ABX060	99.99	NULL

rechnungen

fk

kartennummer	firma	inhaber	ablaufdatum
12345	VISA	Max Mustermann	2017-05-01
12346	Mastercard	Katrin Musterfrau	2018-01-01
12347	American Express	John Doe	2015-02-01
12348	Diners Club	John Doe	2020-03-01

pk

kreditkarten

- Gesucht werden alle Rechnungen.  
Falls sie per Kreditkarte bezahlt wurden, so sollen die Kartendaten ebenfalls ausgegeben werden:

**SELECT**

**r.rechnungnummer, r.kundennummer, r.betrag, r.kartennummer, k.firma,  
k.inhaber, k.ablaufdatum**

**FROM rechnungen r**

**LEFT JOIN kreditkarten k**

**ON r.kartennummer = k.kartennummer**

# DRL – LEFT OUTER JOIN: Beispiel

rechnungnummer	kundennummer	betrag	kartennummer
98765	ABX039	49.95	12345
98766	ABX039	12.95	NULL
98767	ABX040	79.95	12347
98768	ABX050	59.99	12347
98769	ABX050	29.99	12348
98770	ABX060	99.99	NULL

rechnungen

fk

kartennummer	firma	inhaber	ablaufdatum
12345	VISA	Max Mustermann	2017-05-01
12346	Mastercard	Katrin Musterfrau	2018-01-01
12347	American Express	John Doe	2015-02-01
12348	Diners Club	John Doe	2020-03-01

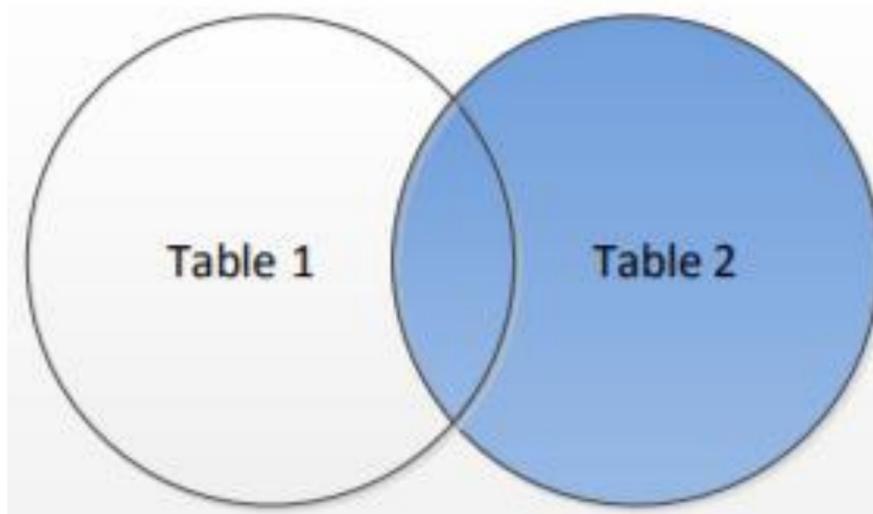
pk

kreditkarten

- Gesucht werden alle Rechnungen.  
Falls sie per Kreditkarte bezahlt wurden, so sollen die Kartendaten ebenfalls ausgegeben werden:

rechnungnummer	kundennummer	betrag	kartennummer	firma	inhaber	ablaufdatum
98765	ABX039	49.95	12345	VISA	Max Mustermann	2017-05-01
98767	ABX040	79.95	12347	American Express	John Doe	2015-02-01
98768	ABX050	59.99	12347	American Express	John Doe	2015-02-01
98769	ABX050	29.99	12348	Diners Club	John Doe	2020-03-01
98766	ABX039	12.95	NULL	NULL	NULL	NULL
98770	ABX060	99.99	NULL	NULL	NULL	NULL

# DRL – RIGHT OUTER JOIN



```
SELECT *  
  FROM Table_1 t1  
 RIGHT JOIN Table_2 t2  
   ON t1.id = t2.fk;
```

# DRL – RIGHT OUTER JOIN: Beispiel

rechnungnummer	kundennummer	betrag	kartennummer
98765	ABX039	49.95	12345
98766	ABX039	12.95	NULL
98767	ABX040	79.95	12347
98768	ABX050	59.99	12347
98769	ABX050	29.99	12348
98770	ABX060	99.99	NULL

rechnungen

fk

kartennummer	firma	inhaber	ablaufdatum
12345	VISA	Max Mustermann	2017-05-01
12346	Mastercard	Katrin Musterfrau	2018-01-01
12347	American Express	John Doe	2015-02-01
12348	Diners Club	John Doe	2020-03-01

pk

kreditkarten

- Gesucht werden alle Karteninformationen.  
Falls mit der entsprechenden Kreditkarte etwas bestellt wurde,  
so sollen die Rechnungsinformationen beigefügt werden :

**SELECT**

**r.rechnungnummer, r.kundennummer, r.betrag, r.kartennummer, k.firma,  
k.inhaber, k.ablaufdatum**

**FROM rechnungen r**

**RIGHT JOIN kreditkarten k**

**ON r.kartennummer = k.kartennummer**

# DRL – RIGHT OUTER JOIN: Beispiel

rechnungnummer	kundennummer	betrag	kartennummer
98765	ABX039	49.95	12345
98766	ABX039	12.95	NULL
98767	ABX040	79.95	12347
98768	ABX050	59.99	12347
98769	ABX050	29.99	12348
98770	ABX060	99.99	NULL

rechnungen

fk

kartennummer	firma	inhaber	ablaufdatum
12345	VISA	Max Mustermann	2017-05-01
12346	Mastercard	Katrin Musterfrau	2018-01-01
12347	American Express	John Doe	2015-02-01
12348	Diners Club	John Doe	2020-03-01

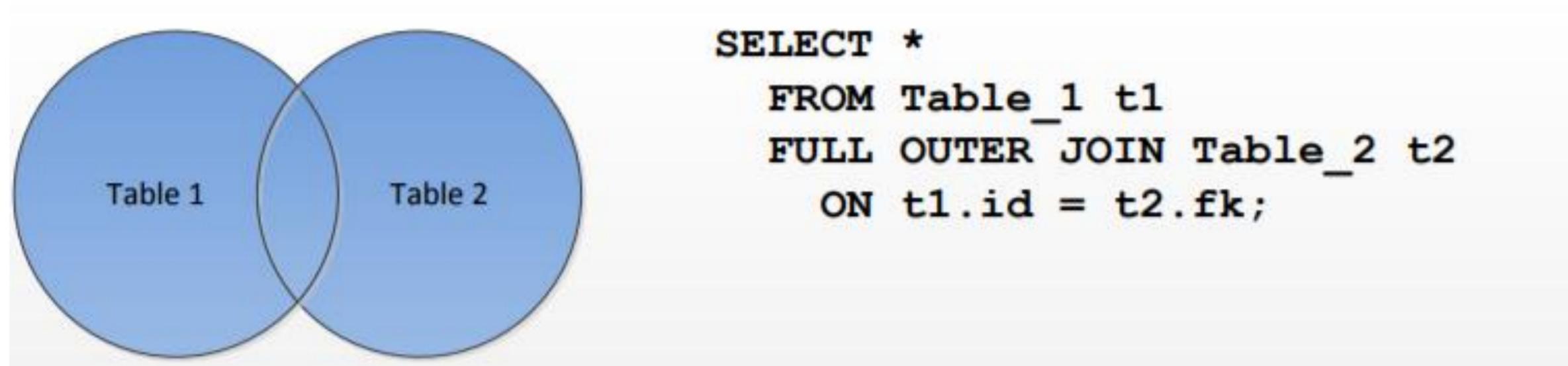
pk

kreditkarten

- Gesucht werden alle Karteninformationen. Falls mit der entsprechenden Kreditkarte etwas bestellt wurde, so sollen die Rechnungsinformationen beigefügt werden :

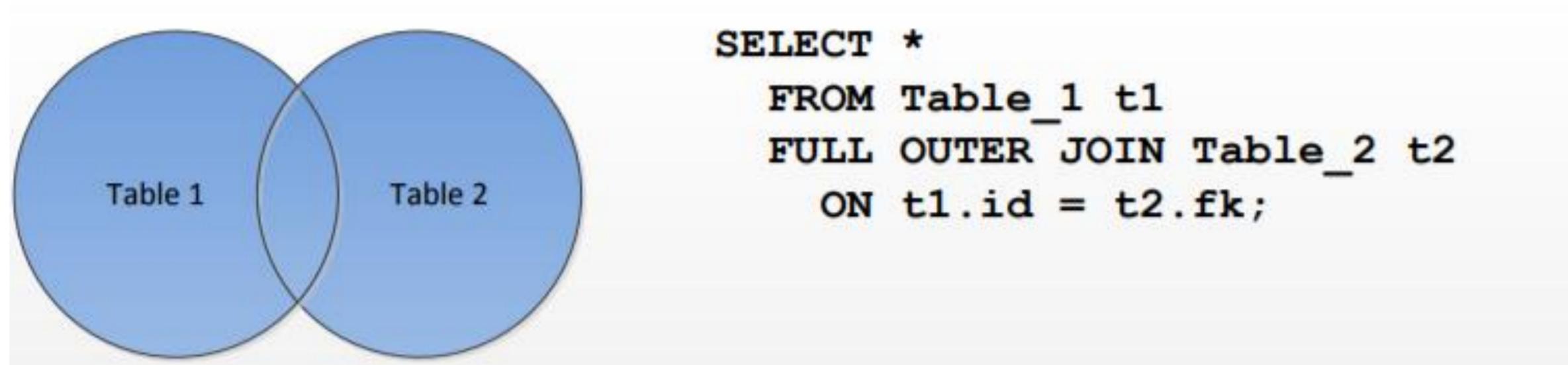
rechnungnummer	kundennummer	betrag	kartennummer	firma	inhaber	ablaufdatum
98765	ABX039	49.95	12345	VISA	Max Mustermann	2017-05-01
98767	ABX040	79.95	12347	American Express	John Doe	2015-02-01
98768	ABX050	59.99	12347	American Express	John Doe	2015-02-01
98769	ABX050	29.99	12348	Diners Club	John Doe	2020-03-01
NULL	NULL	NULL	NULL	Mastercard	Katrin Musterfrau	2018-01-01

# DRL – FULL OUTER JOIN



- Bei einem FULL OUTER JOIN handelt es sich um eine Kombination aus einem LEFT JOIN und einem RIGHT JOIN.
- Das Ziel dabei ist, die Datensätze beidseitig zu verknüpfen.

# DRL – FULL OUTER JOIN



- Leider unterstützt MariaDB kein FULL OUTER JOIN, daher muss man hier ein wenig tricksen, indem man per UNION das Ergebnis zweier Abfragen miteinander verknüpft.
- Diese beiden Abfragen sind eben
  - ein LEFT JOIN und
  - ein RIGHT JOIN,deren Ergebnisse dann mit einem UNION verknüpft werden.

# DRL – FULL OUTER JOIN: Beispiel in MariaDB

```
Tabelle 'hersteller'  
+-----+-----+  
| id | zulieferer |  
+-----+-----+  
| 1 | Hilti      |  
| 2 | Hoch & Tief |  
| 3 | Eisen-Karl |  
| 4 | Stahl AG  |  
| 5 | Gähm & Söhne |  
+-----+-----+
```

pk

```
Tabelle 'produkt'  
+-----+-----+-----+  
| id | produkt      | hs_link |  
+-----+-----+-----+  
| 1 | Schlagbohrer | 1       |  
| 2 | Zement        | 2       |  
| 3 | Kneifzange    | 3       |  
| 4 | Brecheisen    | 3       |  
| 5 | Hammer       | 7       |  
+-----+-----+-----+
```

fk

- Gesucht werden alle Zulieferer und alle Produkte.
- Werden Produkte von bestimmten Zulieferern geliefert, so möchte man dies erkennen.
- Liefern Zulieferer bestimmte Produkte, so möchte man dies auch erkennen...

# DRL – FULL OUTER JOIN: Beispiel in MariaDB

```
Tabelle 'hersteller'
```

id	zulieferer
1	Hilti
2	Hoch & Tief
3	Eisen-Karl
4	Stahl AG
5	Gähn & Söhne

pk

```
Tabelle 'produkt'
```

id	produkt	hs_link
1	Schlagbohrer	1
2	Zement	2
3	Kneifzange	3
4	Brecheisen	3
5	Hammer	7

fk

```
SELECT h.zulieferer, p.produkt  
FROM hersteller h  
LEFT JOIN produkt p ON (h.id = p.hs_link)
```

UNION

```
SELECT h.zulieferer, p.produkt  
FROM hersteller h  
RIGHT JOIN produkt p ON (h.id = p.hs_link);
```

# DRL – FULL OUTER JOIN: Beispiel in MariaDB

```
Tabelle 'hersteller'  
+-----+-----+  
| id | zulieferer |  
+-----+-----+  
| 1 | Hilti      |  
| 2 | Hoch & Tief |  
| 3 | Eisen-Karl |  
| 4 | Stahl AG  |  
| 5 | Gähm & Söhne |  
+-----+-----+
```

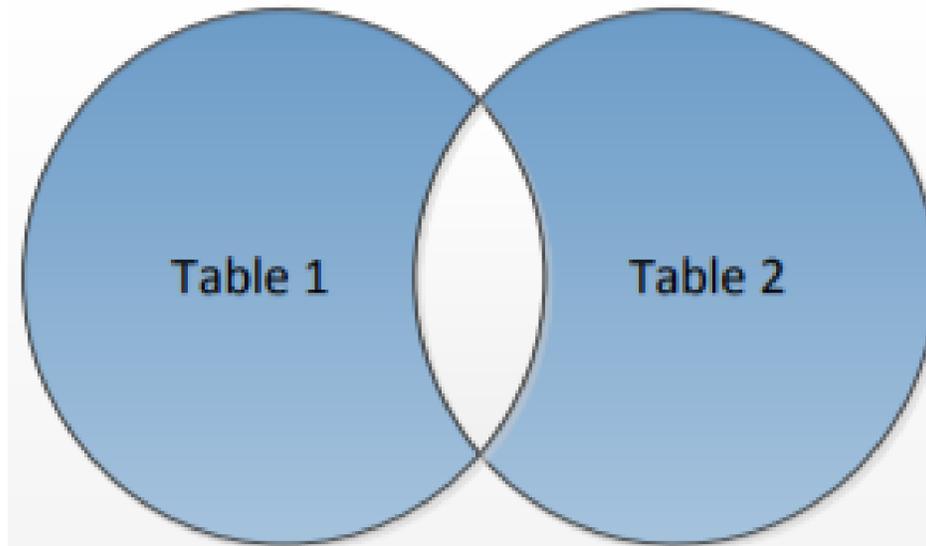
pk

```
Tabelle 'produkt'  
+-----+-----+-----+  
| id | produkt      | hs_link |  
+-----+-----+-----+  
| 1 | Schlagbohrer | 1       |  
| 2 | Zement        | 2       |  
| 3 | Kneifzange   | 3       |  
| 4 | Brecheisen   | 3       |  
| 5 | Hammer       | 7       |  
+-----+-----+-----+
```

fk

```
+-----+-----+  
| zulieferer | produkt |  
+-----+-----+  
| Hilti      | Schlagbohrer |  
| Hoch & Tief | Zement      |  
| Eisen-Karl | Kneifzange  |  
| Eisen-Karl | Brecheisen  |  
| Stahl AG   | NULL       |  
| Gähm & Söhne | NULL      |  
| NULL      | Hammer     |  
+-----+-----+
```

# DRL – FULL OUTER JOIN with exclusion



```
SELECT *  
  FROM Table_1 t1  
 FULL OUTER JOIN Table_2 t2  
   ON t1.id = t2.fk  
 WHERE t1.id is null  
        OR t2.fk is null;
```

# DRL – FULL OUTER JOIN with exclusion: Beispiel in MariaDB

```
Tabelle 'hersteller'
```

id	zulieferer
1	Hilti
2	Hoch & Tief
3	Eisen-Karl
4	Stahl AG
5	Gähn & Söhne

pk

```
Tabelle 'produkt'
```

id	produkt	hs_link
1	Schlagbohrer	1
2	Zement	2
3	Kneifzange	3
4	Brecheisen	3
5	Hammer	7

fk

- Gesucht werden alle Zulieferer, die keine Produkte anbieten  
sowie alle Produkte, die von keinem Zulieferer geliefert werden können...

```
SELECT h.zulieferer, p.produkt FROM hersteller h  
LEFT JOIN produkt p ON (h.id = p.hs_link)  
WHERE (p.hs_link IS NULL)
```

UNION

```
SELECT h.zulieferer, p.produkt FROM hersteller h  
RIGHT JOIN produkt p ON (h.id = p.hs_link)  
WHERE (h.id IS NULL)
```

# DRL – FULL OUTER JOIN with exclusion: Beispiel in MariaDB

```
Tabelle 'hersteller'  
+-----+-----+  
| id | zulieferer |  
+-----+-----+  
| 1 | Hilti      |  
| 2 | Hoch & Tief |  
| 3 | Eisen-Karl |  
| 4 | Stahl AG  |  
| 5 | Gähn & Söhne |  
+-----+-----+
```

pk

```
Tabelle 'produkt'  
+-----+-----+-----+  
| id | produkt      | hs_link |  
+-----+-----+-----+  
| 1 | Schlagbohrer | 1       |  
| 2 | Zement        | 2       |  
| 3 | Kneifzange    | 3       |  
| 4 | Brecheisen    | 3       |  
| 5 | Hammer       | 7       |  
+-----+-----+-----+
```

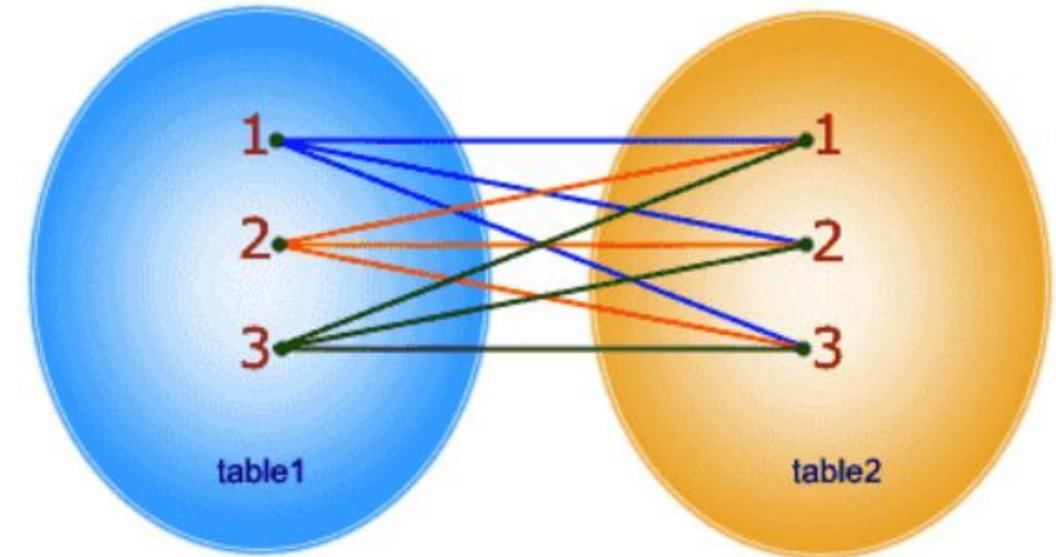
fk

- Gesucht werden alle Zulieferer, die keine Produkte anbieten sowie alle Produkte, die von keinem Zulieferer geliefert werden können...

zulieferer	produkt
Stahl AG	NULL
Gähn & Söhne	NULL
NULL	Hammer

# DRL – CROSS JOIN

- `SELECT * FROM table1  
CROSS JOIN table2;`
- CROSS JOINS, die keine WHERE-Klausel aufweisen, erzeugen das sogenannte kartesische Produkt aus den an dem JOIN beteiligten Tabellen.
- So entspricht die Größe des Resultsets eines kartesisches Produkts der Anzahl der Zeilen in der ersten Tabelle multipliziert mit der Anzahl der Zeilen in der zweiten Tabelle.
- Die Begriffe Kartesisches Produkt, Kreuzprodukt und Cross Join sind synonym.



# DRL – CROSS JOIN

## Beispiel

foods

ITEM_ID	ITEM_NAME	ITEM_UNIT	COMPANY_ID
1	Chex Mix	Pcs	16
6	Cheez-It	Pcs	15
2	BN Biscuit	Pcs	15
3	Mighty Munch	Pcs	17
4	Pot Rice	Pcs	15
5	Jaffa Cakes	Pcs	18
7	Salt n Shake	Pcs	

company

COMPANY_ID	COMPANY_NAME	COMPANY_CITY
18	Order All	Boston
15	Jack Hill Ltd	London
16	Akas Foods	Delhi
17	Foodies.	London
19	sip-n-Bite.	New York

**SELECT**

```
foods.item_name, foods.item_unit, company.company_name,  
company.company_city  
FROM foods, company;
```

... alternativ dazu, auch von MariaDB unterstützt:

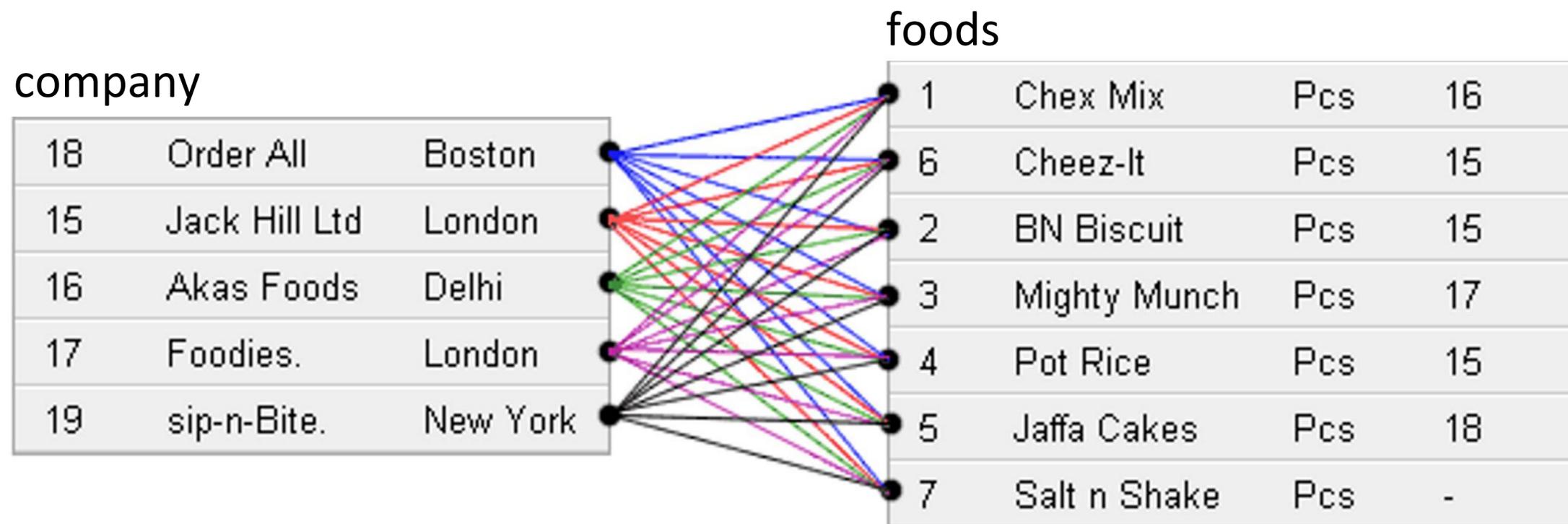
**SELECT**

```
foods.item_name, foods.item_unit, company.company_name,  
company.company_city  
FROM foods  
CROSS JOIN company;
```

# DRL – CROSS JOIN

Beispiel: Jeder mit jedem!

```
SELECT  
foods.item_name, foods.item_unit, company.company_name,  
company.company_city  
FROM foods  
CROSS JOIN company;
```



# DRL – CROSS JOIN

## Beispiel: Riesige Ergebnismengen!

ITEM_NAME	ITEM_UNIT	COMPANY_NAME	COMPANY_CITY
Chex Mix	Pcs	Order All	Boston
Cheez-It	Pcs	Order All	Boston
BN Biscuit	Pcs	Order All	Boston
Mighty Munch	Pcs	Order All	Boston
Pot Rice	Pcs	Order All	Boston
Jaffa Cakes	Pcs	Order All	Boston
Salt n Shake	Pcs	Order All	Boston
Chex Mix	Pcs	Jack Hill Ltd	London
Cheez-It	Pcs	Jack Hill Ltd	London
BN Biscuit	Pcs	Jack Hill Ltd	London
Mighty Munch	Pcs	Jack Hill Ltd	London
Pot Rice	Pcs	Jack Hill Ltd	London
Jaffa Cakes	Pcs	Jack Hill Ltd	London
Salt n Shake	Pcs	Jack Hill Ltd	London
Chex Mix	Pcs	Akas Foods	Delhi
Cheez-It	Pcs	Akas Foods	Delhi
BN Biscuit	Pcs	Akas Foods	Delhi
Mighty Munch	Pcs	Akas Foods	Delhi
Pot Rice	Pcs	Akas Foods	Delhi
Jaffa Cakes	Pcs	Akas Foods	Delhi
Salt n Shake	Pcs	Akas Foods	Delhi
Chex Mix	Pcs	Foodies.	London
.....			