S(tructured)Q(uery)L(anguage)
Verknüpfung von Tabellen

Welche Bestellungen hat Kunde ... aufgegeben?

Welche Kunden im Staat USA haben Waren im Wert von mindestens ... bestellt?

Welche Waren wurden nicht bestellt?

Relationales Datenbankmodell

- Ablage von Daten in Tabellen. Für jede abzubildende Objektgruppe wird eine Tabelle angelegt.
- Erstellungen von Beziehungen (Relationen) zwischen Tabellen. Objektgruppen interagieren miteinander. Zu einem Vorgang werden Details angezeigt.
- Abbildung eines Entity-Relationship-Modells.

Beziehungen zwischen Tabellen

- Verknüpfung zwischen zwei Tabellen.
- Darstellung von Beziehungen zwischen Elementen.
- Verweis mit Hilfe eines Schlüssels auf einen Datensatz in einer anderen Tabelle.

Syntax in einer SQL-Anweisung

```
SELECT [Feld], [Feld]
FROM [Tabelle]

INNER | LEFT OUTER JOIN [Tabelle]
ON [Primärschlüssel] = [Fremdschlüssel]

ORDER BY [Feld] ASC|DESC, [Feld] ASC|DESC;
```

Beispiel

```
SELECT
      "Track"."Name", "Track"."Composer",
      "InvoiceLine"."UnitPrice",
      "InvoiceLine"."Quantity"

FROM "Track"
INNER JOIN "InvoiceLine"
ON ("Track"."TrackId" = "InvoiceLine"."TrackId")

ORDER BY "Track"."Composer", "Track"."Name";
```

Auswahl der Datenfelder

```
SELECT
      "Track"."Name", "Track"."Composer",
      "InvoiceLine"."UnitPrice",
      "InvoiceLine"."Quantity"

FROM "Track"
INNER JOIN "InvoiceLine"
ON ("Track"."TrackId" = "InvoiceLine"."TrackId")

ORDER BY "Track"."Composer", "Track"."Name";
```

Erläuterung

- Dem Befehl `SELECT` folgt eine Liste von Feldern, die angezeigt werden sollen.
- Wenn die Felder in verschiedenen Tabellen definiert sind, werden qualifizierte Bezeichner `tabelle.feld` genutzt.

Auswahl der Master-Tabelle

```
SELECT
      "Track"."Name", "Track"."Composer",
      "InvoiceLine"."UnitPrice",
      "InvoiceLine"."Quantity"

FROM "Track"
INNER JOIN "InvoiceLine"
ON ("Track"."TrackId" = "InvoiceLine"."TrackId")

ORDER BY "Track"."Composer", "Track"."Name";
```

Erläuterung

- „Wurzel“ einer Ordnerstruktur. Der Ordner kann weitere Unterordner enthalten.
- Jeder Datensatz in der Master-Tabelle wird durch einen Primärschlüssel eindeutig identifiziert.

Auswahl der Detail-Tabelle

```
SELECT
      "Track"."Name", "Track"."Composer",
      "InvoiceLine"."UnitPrice",
      "InvoiceLine"."Quantity"

FROM "Track"
INNER JOIN "InvoiceLine"

ON ("Track"."TrackId" = "InvoiceLine"."TrackId")

ORDER BY "Track"."Composer", "Track"."Name";
```

Erläuterung

- Unterordner, die aber auch selber wieder andere Ordner enthalten können.
- Details zu einem Datensatz in einer anderen Tabelle.
- Jeder Datensatz wird wie in der Master-Tabelle durch ein Primärschlüssel eindeutig identifiziert.
- Mindestens ein Datenfeld in der Detail-Tabelle enthält einen Verweis auf einen Datensatz in der dazugehörigen Master-Tabelle.

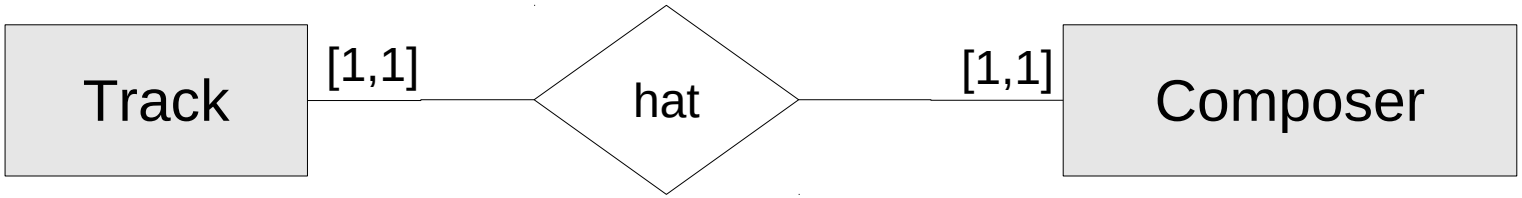
Verknüpfung zwischen den beiden Tabellen

```
SELECT
      "Track"."Name", "Track"."Composer",
      "InvoiceLine"."UnitPrice",
      "InvoiceLine"."Quantity"

FROM "Track"
INNER JOIN "InvoiceLine"

ON ("Track"."TrackId" = "InvoiceLine"."TrackId")

ORDER BY "Track"."Composer", "Track"."Name";
```

Erläuterung

- Dem Befehl `ON` folgt die Verknüpfungsbedingung zwischen der Master- und der Detail-Tabelle.
- Das Gleichheitszeichen symbolisiert eine „ist gleich“-Bedingung.
- Links vom Operator steht häufig der Primärschlüssel aus der Master-Tabelle. Rechts vom Operator wird der dazugehörige Fremdschlüssel definiert.

1 : 1-Relation



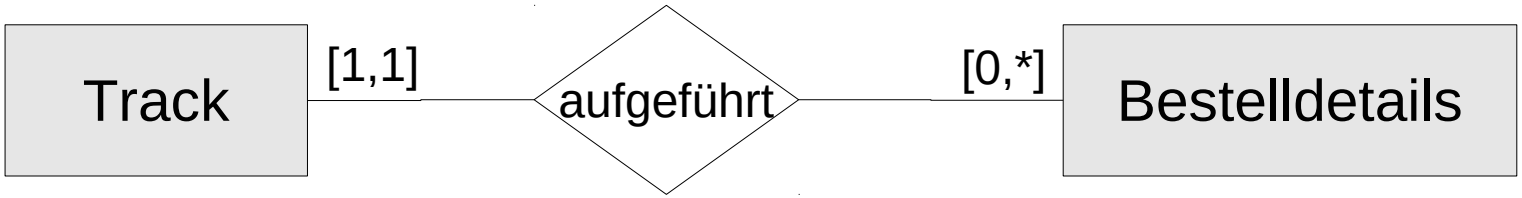
- Beziehungen mit dem Verb „ist“ oder „hat“ bilden eine 1 : 1 – Beziehung ab.
- Jeder Primärschlüssel aus der einen Tabelle wird exakt einmal in der anderen Tabelle verwendet.

… in SQL

```
SELECT
      "Track"."Name", "Track"."Composer"
FROM "Track"
ORDER BY "Track"."Composer", "Track"."Name";
```

- Jede Auswahlabfrage, die als Datenquelle eine Tabelle hat, kann als 1 : 1 – Relation dargestellt werden.

1:n-Relation



- Ein Track wird x Mal zu einem bestimmten Preis bestellt in verschiedenen Bestellungen aufgeführt.
- Jeder Primärschlüssel aus der einen Tabelle kann beliebig oft in der anderen Tabelle verwendet werden.
- Diese Relation ist die am häufigsten vorkommende zwischen zwei Tabellen.

... in SQL

```
SELECT
      "Track"."Name", "Track"."Composer",
      "InvoiceLine"."UnitPrice",
      "InvoiceLine"."Quantity"

FROM "Track"
INNER JOIN "InvoiceLine"
ON ("Track"."TrackId" = "InvoiceLine"."TrackId")

ORDER BY "Track"."Composer", "Track"."Name";
```

Inner Join - Verknüpfung

```
SELECT
      "Track"."Name", "Track"."Composer",
      "InvoiceLine"."UnitPrice",
      "InvoiceLine"."Quantity"

FROM "Track"
INNER JOIN "InvoiceLine"
ON ("Track"."TrackId" = "InvoiceLine"."TrackId")

ORDER BY "Track"."Composer", "Track"."Name";
```

Erläuterung

- Zu einem Fremdschlüssel in der Detail-Tabelle muss ein Primärschlüssel in der Master-Tabelle vorhanden sein.
- Jeder Primärschlüssel aus der Master-Tabelle muss mindestens einmal als Fremdschlüssel in der Detail-Tabelle verwendet werden. Andernfalls wird der Datensatz nicht angezeigt.

Beispiel

Primär-schlüssel	Erdteil
1	Europa
2	Asien
3	Afrika
4	Amerika

Primär-schlüssel	Fremd-schlüssel	Land
1	1	Belgien
2	1	Polen
3	3	Algier
4	2	Indien
5		Spanien

InnerJoin

Erdteil	Land
Europa	Belgien
Europa	Polen
Afrika	Algier
Asien	Indien

LEFT OUTER JOIN - Verknüpfung

```
SELECT
      "Artist"."Name",
      "Album"."Title"

FROM "Artist"
LEFT OUTER JOIN  "Album"
ON ("Artist"."ArtistId" = "Album"."ArtistId")

ORDER BY "Album"."Title", "Artist"."Name";
```

Erläuterung

- Alle Daten aus der Master-Tabelle werden angezeigt. Alle Daten aus der Tabelle links `JOIN` werden angezeigt.
- Falls passende Informationen in der Detail-Tabelle vorhanden sind, werden diese zusätzlich angezeigt. Aus der Tabelle rechts `JOIN` werden nur Datensätze mit einem passenden Fremdschlüssel angezeigt.

Beispiel

Primär-schlüssel	Erdteil
1	Europa
2	Asien
3	Afrika
4	Amerika

Primär-schlüssel	Fremd-schlüssel	Land
1	1	Belgien
2	1	Polen
3	3	Algier
4	2	Indien
5		Spanien

Left Outer Join

Erdteil	Land
Europa	Belgien
Europa	Polen
Afrika	Algier
Asien	Indien
Amerika	

RIGHT OUTER JOIN - Verknüpfung

```
SELECT
      "Artist"."Name",
      "Album"."Title"

FROM "Artist"
RIGHT OUTER JOIN  "Album"
ON ("Artist"."ArtistId" = "Album"."ArtistId")

ORDER BY "Album"."Title", "Artist"."Name";
```

Erläuterung

- Alle Daten aus der Detail-Tabelle werden angezeigt. Alle Daten aus der Tabelle rechts `JOIN` werden angezeigt.
- Falls passende Informationen in der Master-Tabelle vorhanden sind, werden diese zusätzlich angezeigt. Aus der Tabelle links `JOIN` werden nur Datensätze mit einem zu dem Fremdschlüssel passenden Primärschlüssel angezeigt.
- Hinweis: Ein RIGHT OUTER JOIN kann jederzeit in ein LEFT OUTER JOIN durch Tausch der Tabellen umgewandelt werden.

Beispiel

Primär-schlüssel	Erdteil
1	Europa
2	Asien
3	Afrika
4	Amerika

Primär-schlüssel	Fremd-schlüssel	Land
1	1	Belgien
2	1	Polen
3	3	Algier
4	2	Indien
5		Spanien

Right Outer Join

Erdteil	Land
Europa	Belgien
Europa	Polen
Afrika	Algier
Asien	Indien
	Spanien

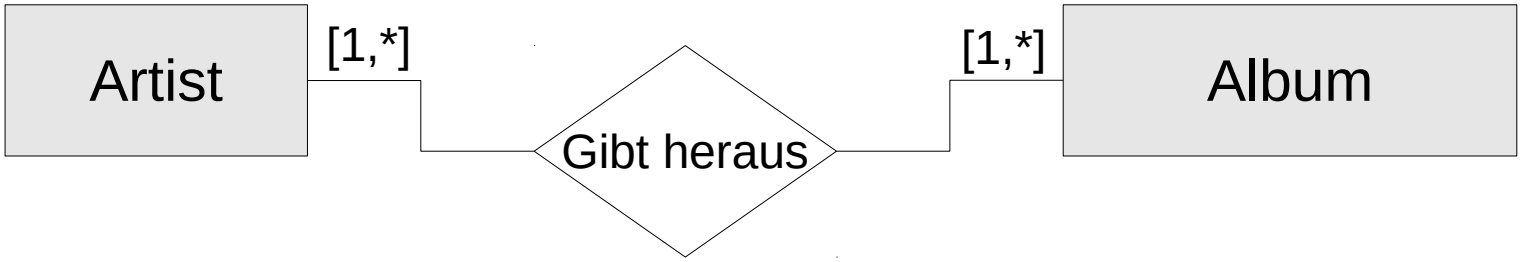
Kombination von JOIN

```
SELECT
      "Track"."Name",
      "Artist"."Name",
      "Album"."Title"

FROM "Track"
INNER JOIN ("Album"
          LEFT OUTER JOIN  "Artist"
          ON ("Album"."ArtistId" =
              "Artist"."ArtistId"))

ON ("Track"."AlbumId" = "Album"."AlbumId");
```

Erläuterung

- Entsprechend der Klammerung werden die Verknüpfungen von innen nach außen aufgelöst.
- Einer OUTER JOIN – Verknüpfung sollte keine INNER JOIN – Verknüpfung folgen.

m : n-Relation



- Beliebig viele Künstler können beliebig viele Alben herausgeben.
- Die Relation wird in einer Pseudo-Tabelle abgebildet, die die Beziehung beschreibt. In dieser Pseudo-Tabelle werden nur Schlüsselwerte aus den beiden Tabellen benutzt plus Attribute, die die Beziehung beschreiben.

… in SQL

```
SELECT "Playlist"."Name", "Track"."Name"

FROM "Playlist"

INNER JOIN ("PlaylistTrack"

      INNER JOIN "Track"
      On ("PlaylistTrack"."TrackId" =
          "Track"."TrackId"))

ON ("PlaylistTrack"."PlaylistId" =
    "Playlist"."PlaylistId")

ORDER BY "Playlist"."Name";
```

Innere Verknüpfung

```
"PlaylistTrack"INNER JOIN "Track"
On ("PlaylistTrack"."TrackId" = "Track"."TrackId"))
```

- Die Quelle `PlaylistTrack` ist die Master-Tabelle. Daten aus dieser Tabelle werden nur angezeigt, wenn der dazugehörige Primärschlüssel mindestens einmal in der Quelle `Track` vorkommt.
- Die, in der Detail-Tabelle `Track` genutzten Fremdschlüssel müssen in der Master-Tabelle definiert sein

Beispiel

Primär-schlüssel	TrackID
t1	Track A
t2	Track B
t3	Track C
t4	Track D

Primär-schlüssel	TrackID	
1	t1	
2	t2	
3	t4	
4	t1	

InnerJoin

Track	
Track A	
Track B	
Track D	

Äußere Verknüpfung

```
FROM "Playlist"
INNER JOIN "PlaylistTrack"

ON ("PlaylistTrack"."PlaylistId" =
    "Playlist"."PlaylistId")
```

- Die Quelle `Playlist` ist die Master-Tabelle. Daten aus dieser Tabelle werden nur angezeigt, wenn der dazugehörige Primärschlüssel mindestens einmal in der Quelle `PlaylistTrack` vorkommt.
- Die, in der Detail-Tabelle `PlaylistTrack` genutzten Fremdschlüssel müssen in der Master-Tabelle definiert sein

Beispiel

Primär-schlüssel	PlaylistID
p1	List A
p2	List B
p3	List C
p4	List D

Primär-schlüssel	TrackID	PlaylistID
1		p1
2		p3
3		p1
4		p2

InnerJoin

Track	Playlist
	List A
	List C
	List B

Vollständige Verknüpfung

```
FROM "Playlist"

INNER JOIN ("PlaylistTrack"

      INNER JOIN "Track"
      On ("PlaylistTrack"."TrackId" =
          "Track"."TrackId"))

ON ("PlaylistTrack"."PlaylistId" =
    "Playlist"."PlaylistId")
```

- Die Verknüpfungen werden von innen nach außen aufgelöst.

Äußere und innere Verknüpfung

Primär-schlüssel	TrackID
t1	Track A
t2	Track B
t3	Track C
t4	Track D

Primär-schlüssel	PlaylistID
p1	List A
p2	List B
p3	List C
p4	List D

Primär-schlüssel	TrackID	PlaylistID
1	t1	p1
2	t2	p3
3	t4	p1
4	t1	p2

InnerJoin

Track	Playlist
Track A	List A
Track B	List C
Track D	List A
Track A	List B