Outer-Joins

Outer-Joins (z.B. ein Left-Outer-Join) zeigen als Ergebnis nur die Datensätze, die keine Referenzdaten in der zweiten Tabelle besitzen.

Beispiel:

Es sollen Daten der Kunden angezeigt werden, die noch keine Reservierung gebucht haben.

```
SELECT K.KdNr, Nachname, Vorname, Ort, AusleihDatum AS AusDat, Leih
    Dauer AS LDauer
FROM Kunden K LEFT JOIN Reservierungen R ON K.KdNr=R.KdNr
WHERE R.KdNr IS NULL;
```

Die Ausgabe lautet nun:

KdNr	Nachname	Vorname	Ort	AusDat	LDauer
1	Palmert	Carlo	Jettingen		
4	Schulze	Anna	Ulm		
6	Müller	Andrea	Biberach		
9	Winkler	Claus	Thalfingen		
10	Hase	Hanna	Günzburg		
11	Hahn	Isabelle	Senden		

Merke:

Inner-Join (auch Equi-Join, Natural Join): Abfragen über mehrere Tabellen, bei denen nur die Schnittmenge der Daten angezeigt wird.

Left-Join / Right Join (auch Left-Inner-Join und Right-Inner-Join): Außer der Schnittmenge wird auch der linke / rechte Bereich der Teilmengen eingeschlossen.

Outer-Join: schließt die Schnittmenge aus und zeigt nur die Datensätze an, die keine Einträge in der verbundenen Tabelle besitzen. Er wird in Access gebildet durch den Zusatz WHERE [Feldname] IS NULL.

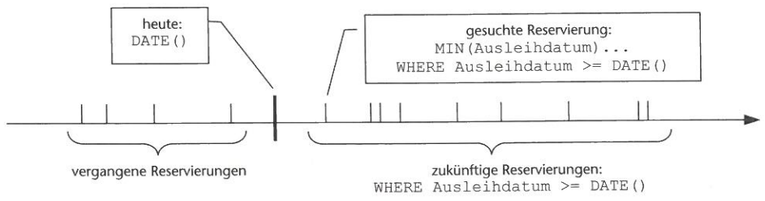
6.3.12 Unterabfragen

Unterabfragen (Subqueries) benötigt man, wenn die Suchbedingung vom Ergebnis einer anderen Abfrage abhängig ist. Sie liefern nur solche Daten, die innerhalb des WHERE-Abschnittes zu einer logisch bewertbaren Bedingung führen, sonst erfolgt eine Fehlermeldung.

Beispiel:

Wir erstellen eine Abfrage, die den Namen des Kunden anzeigt, dessen Reservierung als nächstes fällig ist. Auch die Fahrradnummer und Bezeichnung des Fahrrades werden ausgegeben.

Zweckmäßig ist es, zunächst eine Abfrage zu bilden, die das nächste Reservierungsdatum liefert. Die Funktion MIN(Ausleihdatum) gibt das früheste gespeicherte Reservierungsdatum zurück, die WHERE-Bedingung Ausleihdatum >= DATE() schränkt dann auf das kleinste Datum ab heute ein.



Die Abfrage zur Suche des gewünschten Datumswertes, die anschließend als Unterabfrage dient, lautet somit:


```
SELECT MIN(Ausleihdatum)
FROM Reservierungen
WHERE Ausleihdatum >= Date();
```

Das Ergebnis dieser Abfrage stellt den Wert des kleinsten zukünftigen Ausleihdatums dar, z. B.

Unterabfrage
16.10.202X

Nun muss der entsprechende Datensatz in der Tabelle Reservierungen durch die Bedingung WHERE Ausleihdatum = [Subquery] gesucht werden. Da Daten aus mehreren Tabellen ausgelesen werden, müssen die Tabellen Kunden, Reservierungen und Fahrräder über die notwendigen Equi-Joins miteinander in Beziehung gesetzt werden.

Die gesamte Abfrage lautet nun:

```
SELECT F.Fahrradnummer, F.Bezeichnung, Nachname, Ausleihdatum
FROM Fahrraeder AS F, Reservierungen AS R, Kunden AS K
WHERE K.KdNr = R.KdNr
    AND F.Fahrradnummer = R.FRadNr
    AND Ausleihdatum = (
            SELECT MIN(Ausleihdatı
            FROM Reservierungen
            WHERE Ausleihdatum >= Date()
            );
```

Natural Joins

Subquery

Die Ausgabe lautet dann z. B.:

NaechsteAusleihe			
FNr	**Bezeichnung**	**Name**	**Ausleihdatum**
21	StormRide	Winter	16.10.202X

Hinweis:

Das Ergebnis der Unterabfrage muss vom Datentyp mit dem Vergleichsfeld übereinstimmen. Auch darf eine Unterabfrage nur dann mehrere Ergebnisse liefern, wenn sie z. B. Bestandteil einer Bedingung mit dem IN-Operator ist und somit einer Feldliste entspricht.

Komplexe Abfragen sollten stets modular entwickelt werden. Dadurch werden Fehler rechtzeitig entdeckt und können leichter beseitigt werden.

Beispiel:

Alle Daten der Kunden, die in einem Ort der Fahrradhersteller wohnen, werden aufgelistet. Voraussetzung ist, dass zunächst eine Tabelle „Hersteller" mit den Daten der Fahrradhersteller erzeugt wird.)

```
SELECT *
FROM Kunden
WHERE Kunden.Ort IN
      (
         SELECT Hersteller.Ort
         FROM Hersteller
      );
```