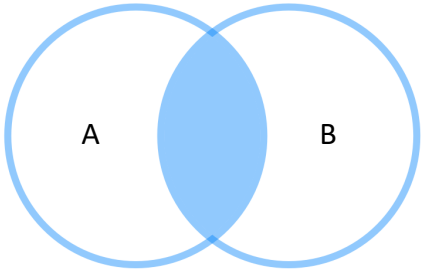
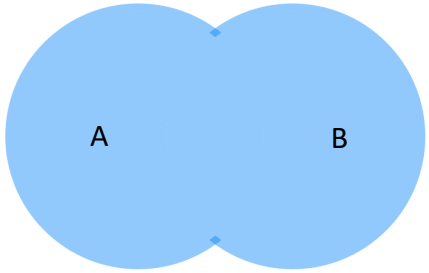
Merge operations in SQL

Remember: The most common ways to merge data

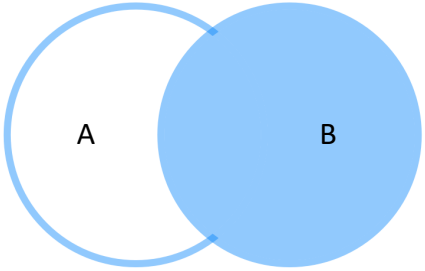
Inner Join



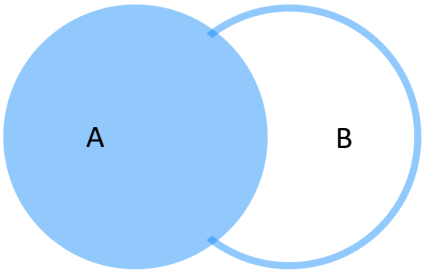
Full Outer Join



Right Outer Join



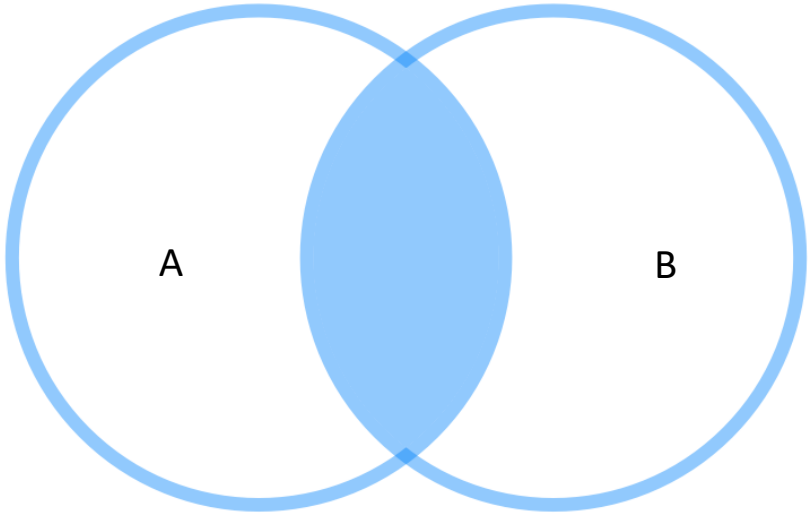
Left Outer Join



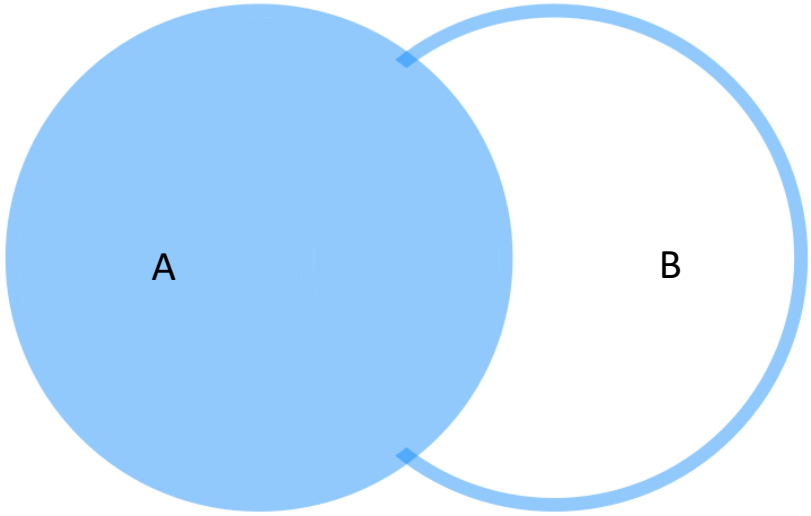
Recap Lecture 20: Merge

SQLite supports…

Inner join



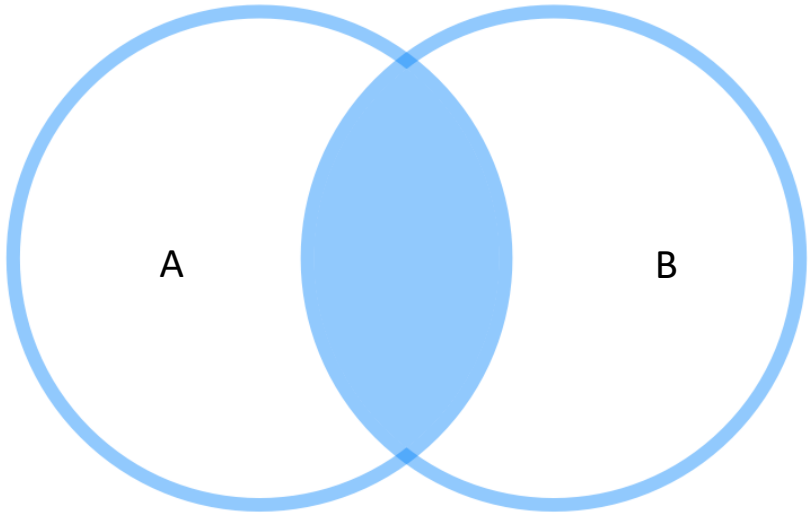
Left outer join



Various merge operations are possible in SQL

1. Inner join
2. Left outer join
3. Inner join by multiple variables

Inner join



R data.table

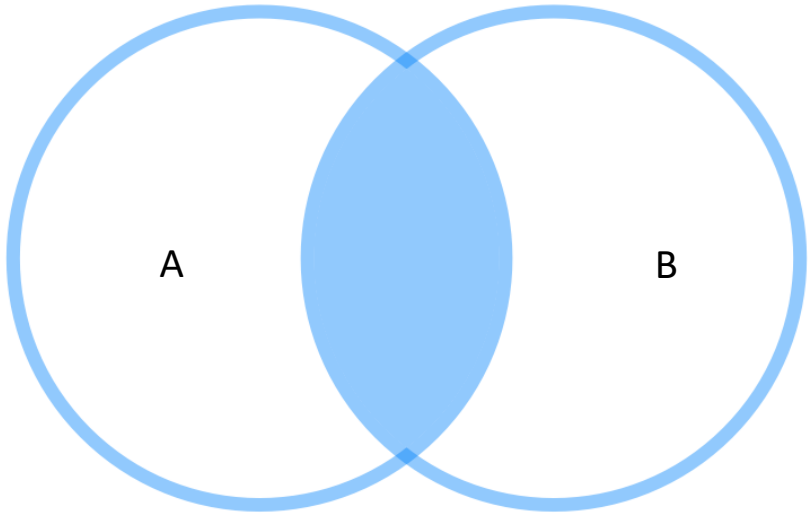
```
merge(A, B, by= "ID", all=FALSE)
```

SQL in R

```
dbGetQuery(con, "SELECT * FROM A
                 INNER JOIN B
                 ON A.ID=B.ID;")
```

1. Common identifier in first table
2. Common identifier in second table

Inner join



R data.table

```
merge(A, B, by= "ID", all=FALSE)
```

SQL in R

```
dbGetQuery(con, "SELECT * FROM A
                 INNER JOIN B
                 ON A.ID=B.ID;")
```

1. Common identifier in first table
2. Common identifier in second table

Inner join merges on common identifiers present in both data.tables

A (`myData`)

Customer	TransDate	Quantity	PurchAmount	Cost
149332	2005-11-15	1	199.95	107.00
172951	2008-08-29	1	199.95	108.00
120621	2007-10-19	1	99.95	49.00
149236	2005-11-14	1	39.95	18.95
149236	2007-12-06	1	79.95	35.00
...	...	...	...	...

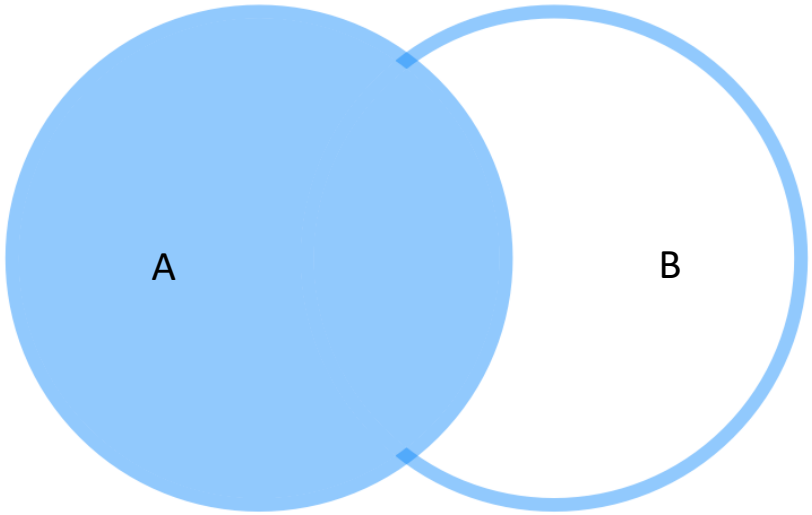
B (`CustData`)

Customer	Gender	Birthdate	ZIP	JoinDate
80365	f	1991-08-26	US-06332	2009-09-15
149332	m	1998-07-07	US-08873	2005-11-05
84374	m	1977-07-10	US-06400	1988-08-10
149236	f	1955-08-15	US-92646	1971-02-16
100001	m	1974-05-08	US-02332	1992-02-21
...	...	...	...	...

Merge rows with the same customer ID if customer ID occurs in both tables

Customer	TransDate	PurchAmount	Cost	Gender	Birthdate	ZIP	JoinDate
149332	2005-11-15	1	199.95	m	1998-07-07	US-08873	05.11.2005
149236	2005-11-14	1	39.95	f	1955-08-15	US-92646	16.02.1971
149236	2001-06-12	1	79.95	f	1955-08-15	US-92646	16.02.1971
...	...	...	...	...	...	...	...

Left outer join



R data.table

```
merge(A, B, by= "ID", all.x=TRUE)
```

SQL in R

```
dbGetQuery(con, "SELECT * FROM A
                 LEFT JOIN B
                 ON A.ID=B.ID;")
```

Left join merges on all of the common identifiers in the left specified table

A (`myData`) ← left table

Customer	TransDate	Quantity	PurchAmount	Cost
149332	2005-11-15	1	199.95	107.00
172951	2008-08-29	1	199.95	108.00
120621	2007-10-19	1	99.95	49.00
149236	2005-11-14	1	39.95	18.95
149236	2007-12-06	1	79.95	35.00
...	...	...	...	...

B (`CustData`) ← right table

Customer	Gender	Birthdate	ZIP	JoinDate
80365	f	1991-08-26	US-06332	2009-09-15
149332	m	1998-07-07	US-08873	2005-11-05
84374	m	1977-07-10	US-06400	1988-08-10
149236	f	1955-08-15	US-92646	1971-02-16
100001	m	1974-05-08	US-02332	1992-02-21
...	...	...	...	...

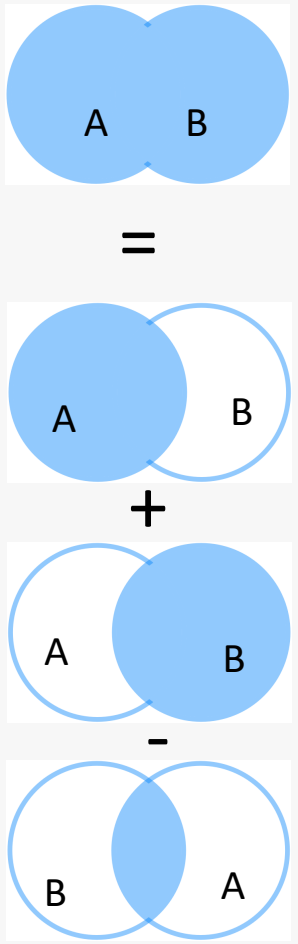
Append column entries of table B if customer ID matches with customer ID of table A.

Customer	TransDate	Quantity	PurchAmount	Cost	Gender	Birthdate	ZIP	JoinDate
149332	2005-11-15	1	199.95	107.00	m	1998-07-07	US-08873	2005-11-05
172951	2008-08-29	1	199.95	108.00	NA	NA	NA	NA
120621	2007-10-19	1	99.95	49.00	NA	NA	NA	NA
149236	2005-11-14	1	39.95	18.95	m	1998-07-07	US-08873	2005-11-05
149236	2001-06-12	1	79.95	35.00	m	1998-07-07	US-08873	2005-11-05
...	...	...	...	...	...	...	...	...

Sidenote: Think outside the box
Enforcing a full outer join with SQLite

```
dbGetQuery(con,    "SELECT *
                    FROM A
                    LEFT JOIN B
                    ON A.ID = B.ID
                   UNION ALL
                    SELECT *
                    FROM B
                    LEFT JOIN A
                    ON B.ID = A.ID
                    WHERE A.ID IS NULL;")
```

Left outer join of X on Y... 1

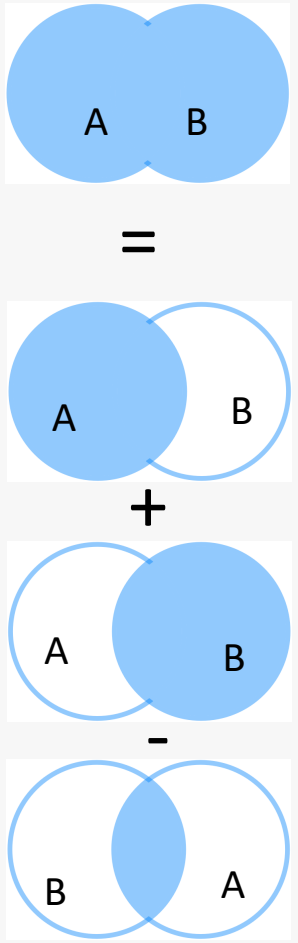


Sidenote: Think outside the box
Enforcing a full outer join with SQLite

```
dbGetQuery(con,    "SELECT *
                    FROM A
                    LEFT JOIN B
                    ON A.ID = B.ID
                   UNION ALL
                    SELECT *
                    FROM B
                    LEFT JOIN A
                    ON B.ID = A.ID
                    WHERE A.ID IS NULL;")
```

1 Left outer join of X on Y...

2 ... combined with...



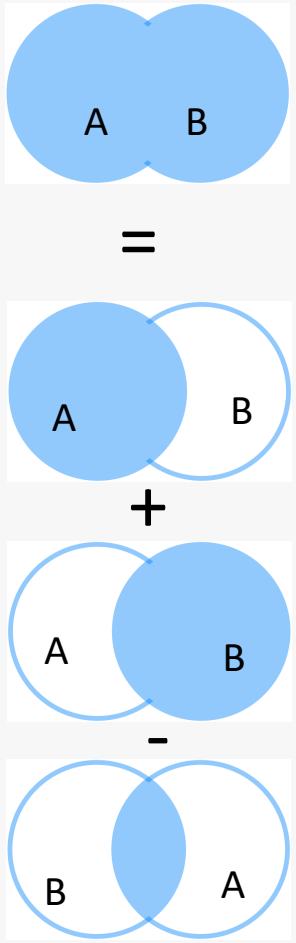
Sidenote: Think outside the box
Enforcing a full outer join with SQLite

```
dbGetQuery(con,    "SELECT *
                    FROM A
                    LEFT JOIN B
                    ON A.ID = B.ID
                   UNION ALL
                    SELECT *
                    FROM B
                    LEFT JOIN A
                    ON B.ID = A.ID
                    WHERE A.ID IS NULL;")
```

1 Left outer join of X on Y...

2 ... combined with...

3 ...left outer join of Y on X minus all elements occurring in both tables.

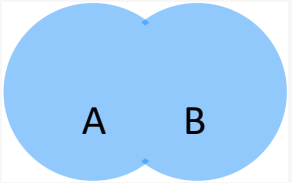


Sidenote: Full and right outer join in SQL

Other database systems as MySQL or SQL Server offer the following options:

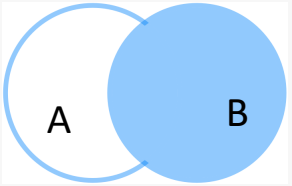
1. **Full outer join**

```
SELECT *
FROM X
FULL OUTER JOIN Y
ON X.ID=Y.ID;
```

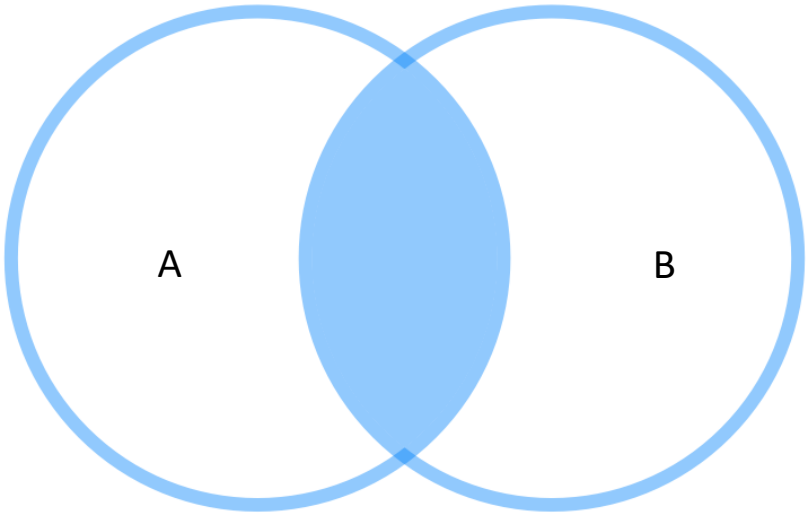


2. **Right outer join**

```
SELECT *
FROM X
RIGHT JOIN Y
ON X.ID=Y.ID;
```



Inner join by multiple variables



R data.table

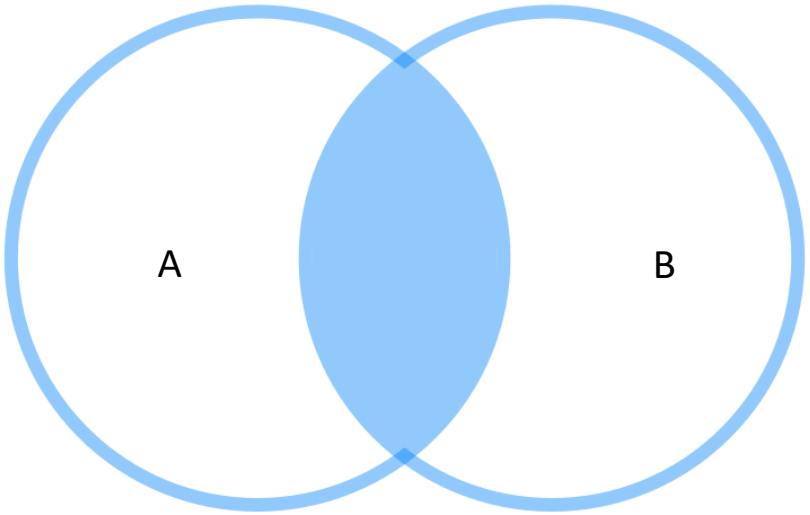
```
merge(A, B, by=c("ID", "TransDate"), all=FALSE)
```

SQL in R

```
dbGetQuery(con, "SELECT * FROM A
                 INNER JOIN B
                 ON A.ID=B.ID
                 AND A.TransDate=B.Transdate;")
```

Connect multiple identifiers with `AND`

Inner join by multiple variables



R data.table

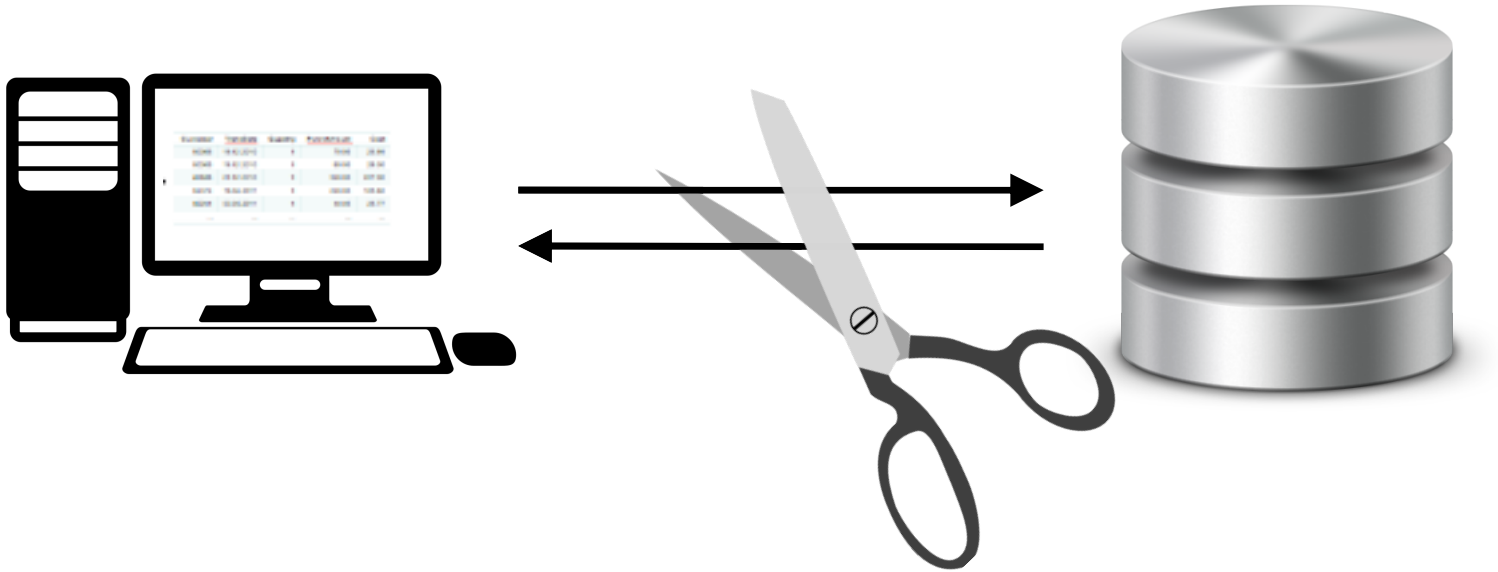
```
merge(A, B, by=c("ID", "TransDate"), all=FALSE)
```

SQL in R

```
dbGetQuery(con, "SELECT * FROM A
                 INNER JOIN B
                 ON A.ID=B.ID
                 AND A.TransDate=B.Transdate;")
```

Connect multiple identifiers with `AND`

DB step 3:
Close connection to free resources



```
dbDisconnect(conn=con)
```

Recap Lecture 06:
Load Data