

```
1  -- Demonstrates aggregation by groups with GROUP BY
2  -- Uses longlist.db
3
4  -- Finds average rating for each book
5  SELECT "book_id", ROUND(AVG("rating"), 2) AS "average rating" FROM "ratings"
6  GROUP BY "book_id";
7
8  -- Joins titles
9  SELECT "title", ROUND(AVG("rating"), 2) AS "average rating" FROM "ratings"
10 JOIN "books" ON "books"."id" = "ratings"."book_id"
11 GROUP BY "book_id";
12
13 -- Chooses books with a rating of 4.0 or higher
14 SELECT "title", ROUND(AVG("rating"), 2) AS "average rating" FROM "ratings"
15 JOIN "books" ON "books"."id" = "ratings"."book_id"
16 GROUP BY "book_id"
17 HAVING "average rating" > 4.0;
```

```
1 -- Demonstrates joining tables with JOIN
2 -- Uses sea_lions.db
3
4 -- Shows all sea lions for which we have data
5 SELECT * FROM "sea_lions"
6 JOIN "migrations" ON "migrations"."id" = "sea_lions"."id";
7
8 -- Shows all sea lions, whether or not we have data
9 SELECT * FROM "sea_lions"
10 LEFT JOIN "migrations" ON "migrations"."id" = "sea_lions"."id";
11
12 -- Shows all data, whether or not there are matching sea lions
13 SELECT * FROM "sea_lions"
14 RIGHT JOIN "migrations" ON "migrations"."id" = "sea_lions"."id";
15
16 -- Shows all data and all sea lions
17 SELECT * FROM "sea_lions"
18 FULL JOIN "migrations" ON "migrations"."id" = "sea_lions"."id";
19
20 -- JOINS sea lions and migrations without specifying matching column
21 SELECT * FROM "sea_lions"
22 NATURAL JOIN "migrations";
23
24 -- Uses WHERE after joining a table
25 SELECT * FROM "sea_lions"
26 JOIN "migrations" ON "migrations"."id" = "sea_lions"."id"
27 WHERE "migrations"."distance" > 1500;
```

```
1 -- Demonstrates subqueries
2 -- Uses longlist.db
3
4 -- Finds all books published by MacLehose Press, with hard-coded id
5 SELECT "id" FROM "publishers" WHERE "publisher" = 'MacLehose Press';
6
7 SELECT "title" FROM "books" WHERE "publisher_id" = 12;
8
9 -- Finds all books published by MacLehose Press, with a nested query
10 SELECT "title" FROM "books" WHERE "publisher_id" = (
11     SELECT "id" FROM "publishers" WHERE "publisher" = 'MacLehose Press'
12 );
13
14 -- Finds all ratings for "In Memory of Memory"
15 SELECT "rating" FROM "ratings" WHERE "book_id" = (
16     SELECT "id" FROM "books" WHERE "title" = 'In Memory of Memory'
17 );
18
19 -- Finds average rating for "In Memory of Memory"
20 SELECT AVG("rating") FROM "ratings" WHERE "book_id" = (
21     SELECT "id" FROM "books" WHERE "title" = 'In Memory of Memory'
22 );
23
24 -- Finds author who wrote "The Birthday Party"
25 SELECT "id" FROM "books" WHERE "title" = 'The Birthday Party';
26
27 SELECT "author_id" FROM "authored" WHERE "book_id" = (
28     SELECT "id" FROM "books" WHERE "title" = 'The Birthday Party'
29 );
30
31 SELECT "name" FROM "authors" WHERE "id" = (
32     SELECT "author_id" FROM "authored" WHERE "book_id" = (
33         SELECT "id" FROM "books" WHERE "title" = 'The Birthday Party'
34     )
35 );
36
37 -- Finds all books by Fernanda Melchor, using IN
38 SELECT "id" FROM "authors" WHERE "name" = 'Fernanda Melchor';
39
40 SELECT "book_id" FROM "authored" WHERE "author_id" = (
41     SELECT "id" FROM "authors" WHERE "name" = 'Fernanda Melchor'
42 );
```

```
43
44 SELECT "title" FROM "books" WHERE "id" IN (
45     SELECT "book_id" FROM "authored" WHERE "author_id" = (
46         SELECT "id" FROM "authors" WHERE "name" = 'Fernanda Melchor'
47     )
48 );
49
50 -- Uses IN to search for multiple authors
51 SELECT "title" FROM "books" WHERE "id" IN (
52     SELECT "book_id" FROM "authored" WHERE "author_id" IN (
53         SELECT "id" FROM "authors" WHERE "name" IN ('Fernanda Melchor', 'Annie Ernaux')
54     )
55 );
```

```
1 -- Demonstrates set operations
2 -- Uses longlist.db
3
4 -- UNION
5 -- Selects all authors, labeling as authors
6 SELECT 'author' AS "profession", "name" FROM "authors";
7
8 -- Selects all translators, labeling as translators
9 SELECT 'translator' AS "profession", "name" FROM "translators";
10
11 -- Combines authors and translators into one result set
12 SELECT 'author' AS "profession", "name" FROM "authors";
13 UNION
14 SELECT 'translator' AS "profession", "name" FROM "translators";
15
16 -- INTERSECT (Assume names are unique)
17 -- Finds authors and translators
18 SELECT "name" FROM "authors"
19 INTERSECT
20 SELECT "name" FROM "translators";
21
22 -- Finds books translated by Sophie Hughes
23 SELECT "book_id" FROM "translated" WHERE "translator_id" = (
24     SELECT "id" FROM "translators" WHERE name = 'Sophie Hughes'
25 );
26
27 -- Finds books translated by Margaret Jull Costa
28 SELECT "book_id" FROM "translated" WHERE "translator_id" = (
29     SELECT "id" FROM "translators" WHERE name = 'Margaret Jull Costa'
30 );
31
32 -- Finds intersection of books
33 SELECT "book_id" FROM "translated" WHERE "translator_id" = (
34     SELECT "id" FROM "translators" WHERE name = 'Sophie Hughes'
35 )
36 INTERSECT
37 SELECT "book_id" FROM "translated" WHERE "translator_id" = (
38     SELECT "id" FROM "translators" WHERE name = 'Margaret Jull Costa'
39 );
40
41 -- Finds intersection of books
42 SELECT "title" FROM "books" WHERE "id" = (
```

```
43     SELECT "book_id" FROM "translated" WHERE "translator_id" = (  
44     SELECT "id" FROM "translators" WHERE name = 'Sophie Hughes'  
45     )  
46     INTERSECT  
47     SELECT "book_id" FROM "translated" WHERE "translator_id" = (  
48         SELECT "id" FROM "translators" WHERE name = 'Margaret Jull Costa'  
49     )  
50 );  
51  
52 -- EXCEPT (Assume names are unique)  
53 -- Finds translators who are not authors  
54 SELECT "name" FROM "translators"  
55 EXCEPT  
56 SELECT "name" FROM "authors";
```