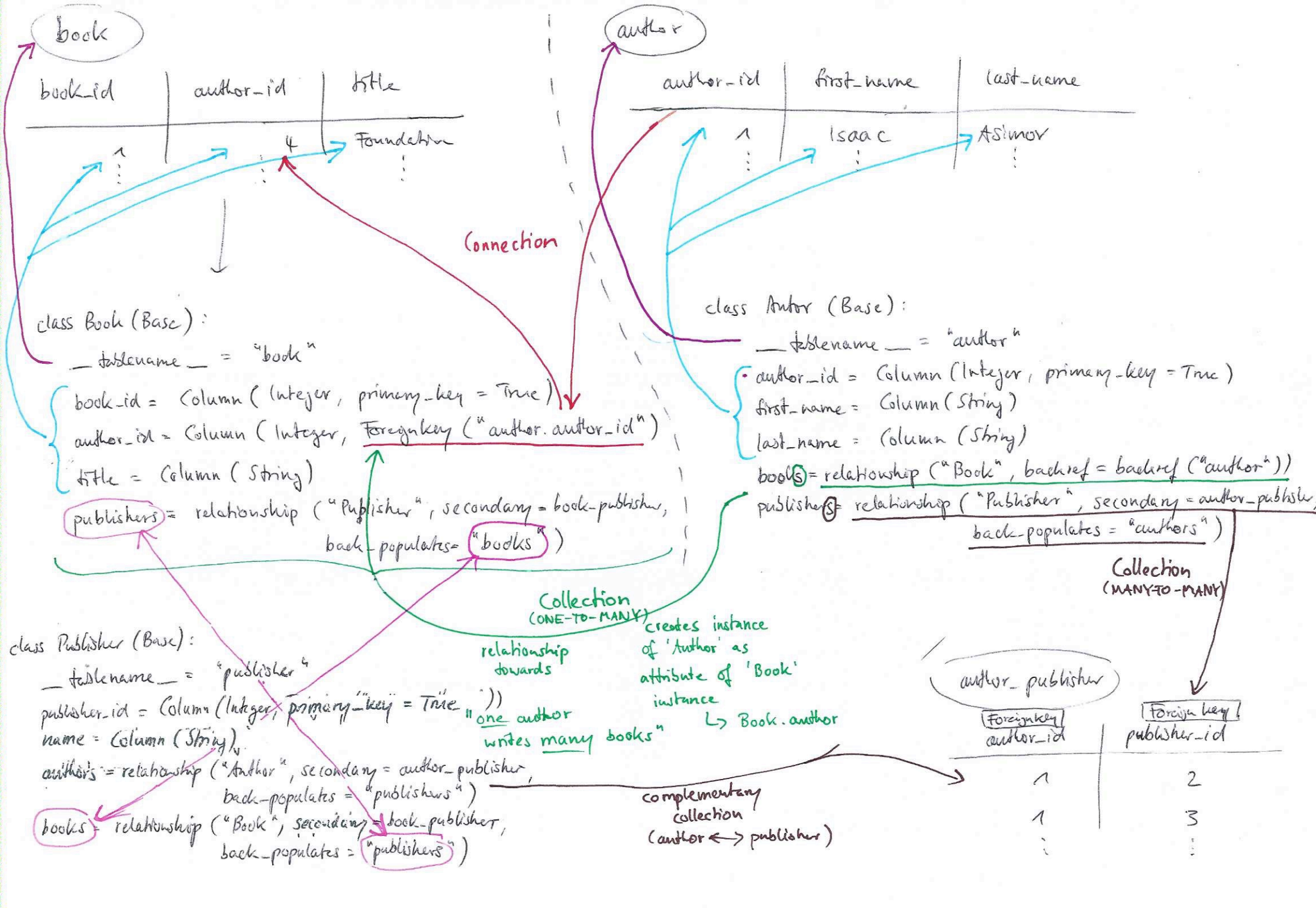
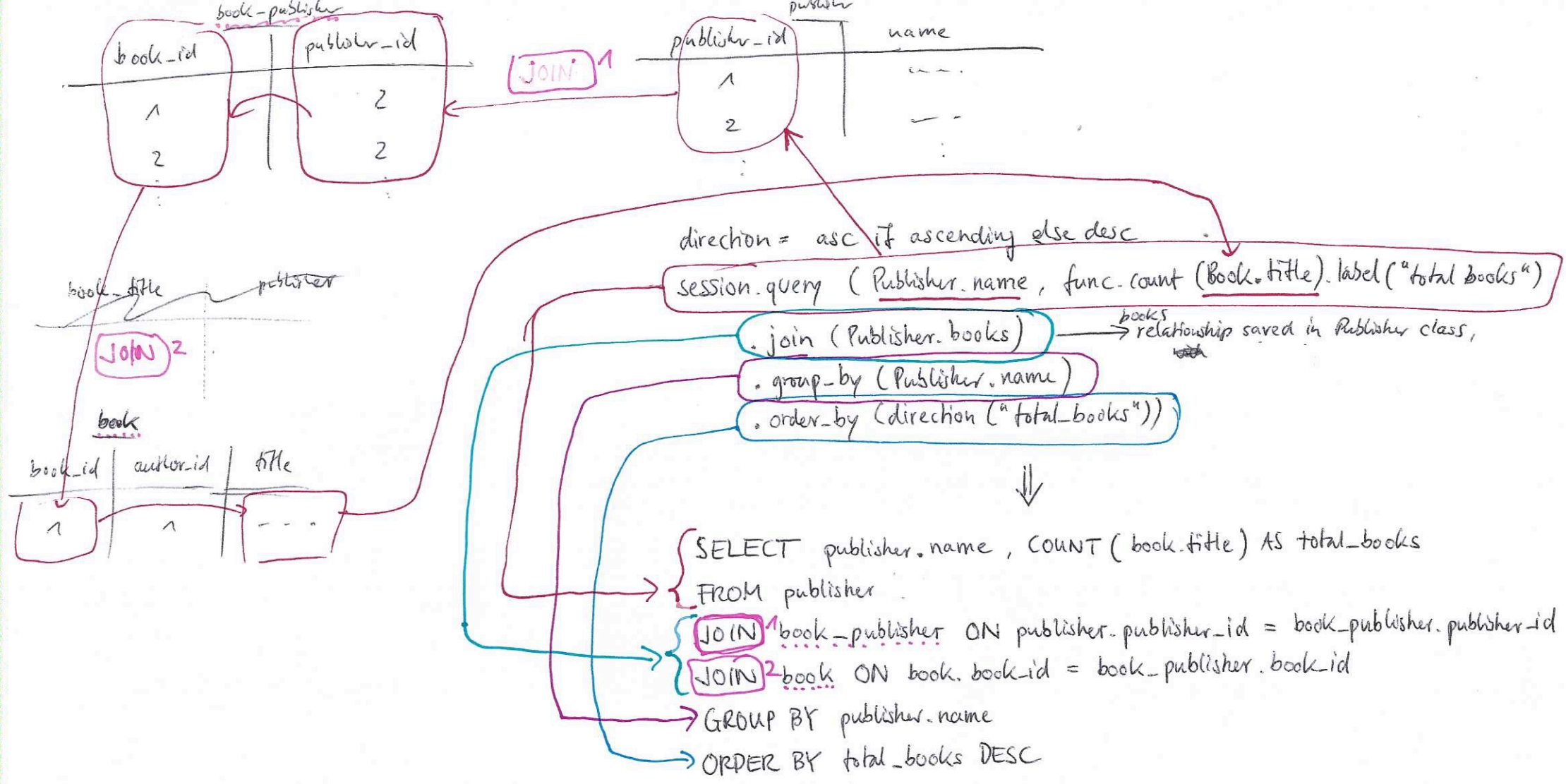

book
book_id
author_id
title
4
Foundation
author
author_id
first_name
last_name
1
Isaac
Asimov
Connection
class Book (Base):
__tablename__ = "book"
book_id = Column (Integer, primary_key = True)
author_id = Column (Integer, ForeignKey ("author.author_id"))
title = Column (String)
publishers = relationship ("Publisher", secondary = book_publisher,
back_populates = "books")
class Author (Base):
__tablename__ = "author"
author_id = Column (Integer, primary_key = True)
first_name = Column (String)
last_name = Column (String)
books = relationship ("Book", backref = backref ("author"))
publishers = relationship ("Publisher", secondary = author_publisher,
back_populates = "authors")
Collection
(MANY-TO-MANY)
Collection
(ONE-TO-MANY)
relationship
towards
creates instance
of 'Author' as
attribute of 'Book'
instance
↳ Book.author
"one author
writes many books"
class Publisher (Base):
__tablename__ = "publisher"
publisher_id = Column (Integer, primary_key = True))
name = Column (String)
authors = relationship ("Author", secondary = author_publisher,
back_populates = "publishers")
books = relationship ("Book", secondary = book_publisher,
back_populates = "publishers")
complementary
collection
(author ↔ publisher)
author_publisher
Foreignkey
author_id
Foreign key
publisher_id
1
2
1
3


book_publisher
book_id
publisher_id
1
2
2
2
JOIN 1
publisher
publisher_id
name
1
2
book_title
publisher
JOIN 2
book
book_id
author_id
title
1
1
direction = asc if ascending else desc
session.query (Publisher.name , func.count (Book.title).label("total books")
.join (Publisher.books)
books relationship saved in Publisher class,
.group_by (Publisher.name)
.order_by (direction ("total_books"))
SELECT publisher.name , COUNT (book.title) AS total_books
FROM publisher
JOIN 1 book_publisher ON publisher.publisher_id = book_publisher.publisher_id
JOIN 2 book ON book.book_id = book_publisher.book_id
GROUP BY publisher.name
ORDER BY total_books DESC