BOOK STORE MANAGEMENT SYSTEM

ENTITIES

1. Book: id, name, description, publisher_id, ISBN, quantity , price

2. Category: id, name, description

3. Author: id, name, birth_date, nationality

4. Publisher: id, name, address, contact_information

5. Customer: id, name, address, phone_number, email

6. Order: id, customer_id, order_date, order_status, total_price.

7. Payment: id order_id, payment_date, payment_method, payment_amount.

8. Employee : id, name, job_title, department, hire_date, contact_information.

RELATIONSHIPS

- One book can belong to one or more categories, but each category can have many books(many-to-many)

- Many books can be written by one or more authors, and one author can write one or more books.(many-to-many)

- One book can be published by only one publisher, and one publisher can publish many books. (one-to-many)

- One customer can place many orders, but each order is only placed by one customer. (one-to-many)

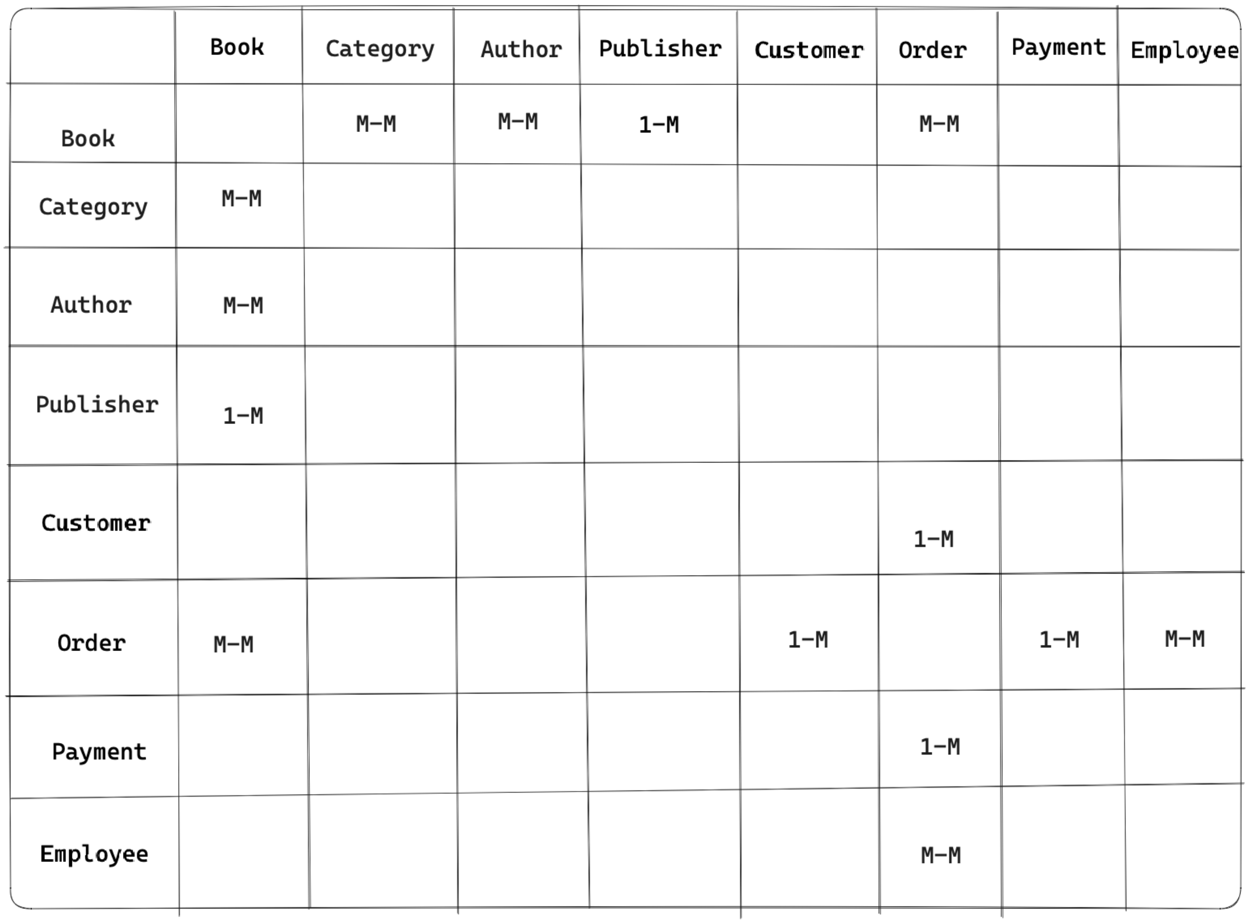
- One order can contain many books, and one book can be in many orders (many-to-many)

- One order can be associated with one or many payments, but each payment is only associated with one order. (one-to-many)

- An employee can process multiple orders, and an order can be processed by one or more employees.(many-to-many)

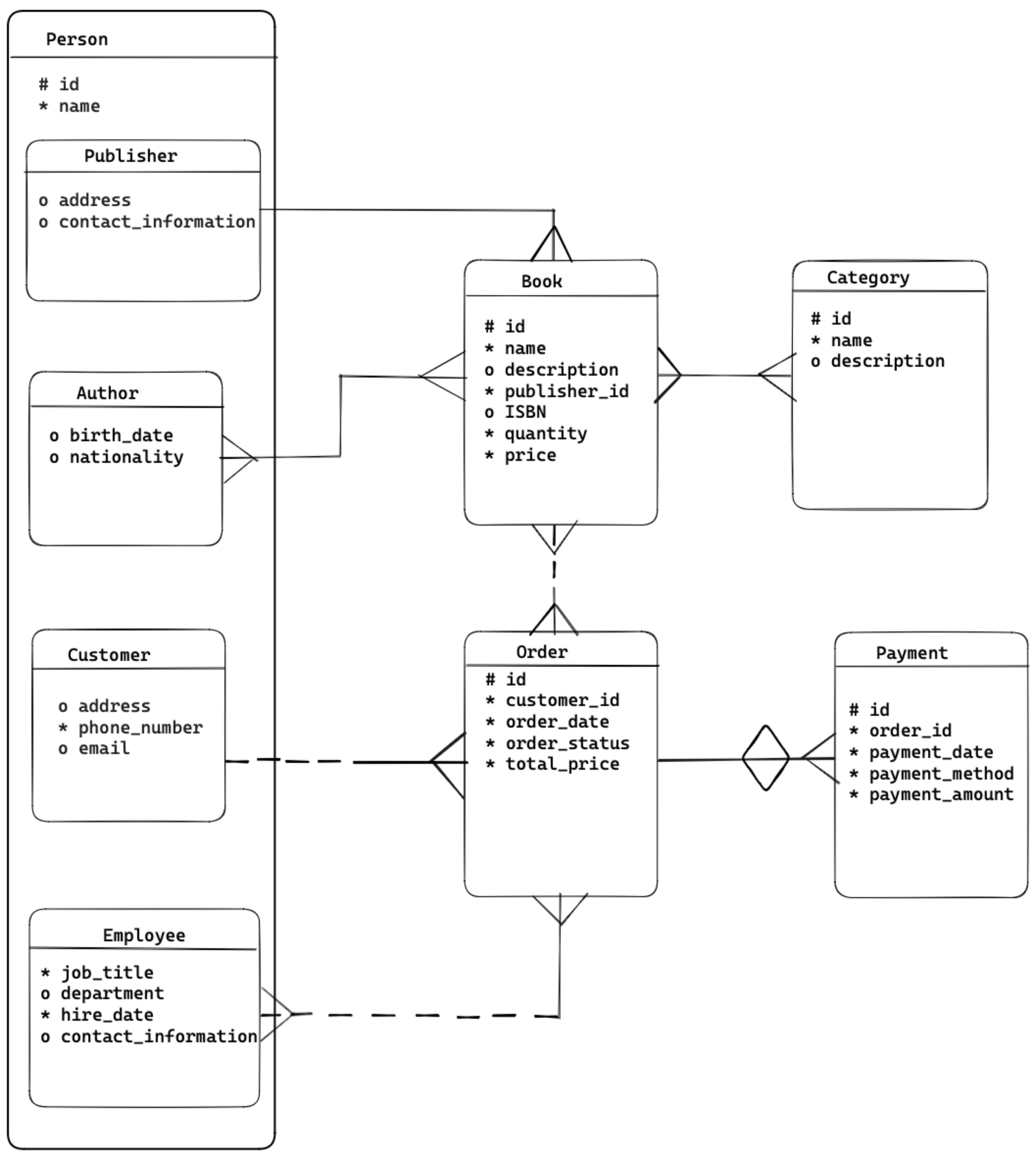
Matrix Diagram

Matrix Diagram



	Book	Category	Author	Publisher	Customer	Order	Payment	Employee
Book		M–M	M–M	1–M		M–M		
Category	M–M							
Author	M–M							
Publisher	1–M							
Customer						1–M		
Order	M–M				1–M		1–M	M–M
Payment						1–M		
Employee						M–M		

ERD


Person
# id
* name
Publisher
o address
o contact_information
Author
o birth_date
o nationality
Customer
o address
* phone_number
o email
Employee
* job_title
o department
* hire_date
o contact_information
Book
# id
* name
o description
* publisher_id
o ISBN
* quantity
* price
Category
# id
* name
o description
Order
# id
* customer_id
* order_date
* order_status
* total_price
Payment
# id
* order_id
* payment_date
* payment_method
* payment_amount

Writing SQL DDL statements for implementing ERD (create table, constraints, defining keys: pks and fks)

```
CREATE TABLE Book (
  id NUMBER PRIMARY KEY,
  name VARCHAR2(255) NOT NULL,
  description VARCHAR2(255),
  publisher_id NUMBER,
  ISBN VARCHAR2(255),
  quantity NUMBER,
  price NUMBER(10, 2),
  CONSTRAINT fk_book_publisher FOREIGN KEY (publisher_id) REFERENCES Publisher(id)
);

CREATE TABLE Category (
  id NUMBER PRIMARY KEY,
  name VARCHAR2(255) NOT NULL,
  description VARCHAR2(255)
);

CREATE TABLE Author (
  id NUMBER PRIMARY KEY,
  name VARCHAR2(255) NOT NULL,
  birth_date DATE,
  nationality VARCHAR2(255)
);

CREATE TABLE Publisher (
  id NUMBER PRIMARY KEY,

  name VARCHAR2(255) NOT NULL,

  address VARCHAR2(255),

  contact_information VARCHAR2(255)

);


CREATE TABLE Customer (

  id NUMBER PRIMARY KEY,

  name VARCHAR2(255) NOT NULL,

  address VARCHAR2(255),

  phone_number VARCHAR2(20),

  email VARCHAR2(255)

);


CREATE TABLE “Order” (

  id NUMBER PRIMARY KEY,

  customer_id NUMBER,

  order_date DATE,

  order_status VARCHAR2(255),

  total_price NUMBER(10, 2),

  CONSTRAINT fk_order_customer FOREIGN KEY (customer_id) REFERENCES Customer(id)

);


CREATE TABLE Payment (

  id NUMBER PRIMARY KEY,

  order_id NUMBER,

  payment_date DATE,

  payment_method VARCHAR2(255),

  payment_amount NUMBER(10, 2),
  CONSTRAINT fk_payment_order FOREIGN KEY (order_id) REFERENCES "Order"(id)
);

CREATE TABLE Employee (
  id NUMBER PRIMARY KEY,
  name VARCHAR2(255) NOT NULL,
  job_title VARCHAR2(255),
  department VARCHAR2(255),
  hire_date DATE,
  contact_information VARCHAR2(255)
);

CREATE TABLE Book_Category (
  book_id NUMBER,
  category_id NUMBER,
  CONSTRAINT fk_book_category_book FOREIGN KEY (book_id) REFERENCES Book(id),
  CONSTRAINT fk_book_category_category FOREIGN KEY (category_id) REFERENCES Category(id)
);

CREATE TABLE Book_Author (
  book_id NUMBER,
  author_id NUMBER,
  CONSTRAINT fk_book_author_book FOREIGN KEY (book_id) REFERENCES Book(id),
  CONSTRAINT fk_book_author_author FOREIGN KEY (author_id) REFERENCES Author(id)
);

CREATE TABLE Order_Book (
  order_id NUMBER,
  book_id NUMBER,
  CONSTRAINT fk_order_book_order FOREIGN KEY (order_id) REFERENCES "Order"(id),
  CONSTRAINT fk_order_book_book FOREIGN KEY (book_id) REFERENCES Book(id)
);
 CREATE TABLE Order_Employee (
  order_id NUMBER,
  employee_id NUMBER,
  CONSTRAINT fk_order_employee_order FOREIGN KEY (order_id) REFERENCES "Order"(id),
  CONSTRAINT fk_order_employee_employee FOREIGN KEY (employee_id) REFERENCES Employee(id)
);
```

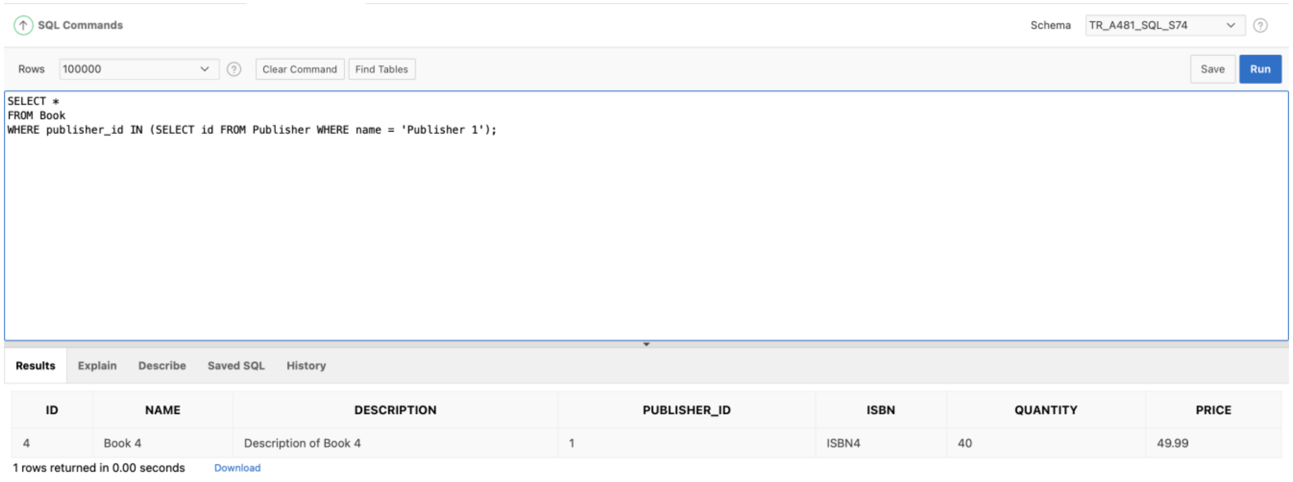
Entering data to the Database

```
INSERT INTO Publisher (id, name, address, contact_information)
VALUES
  (3, 'Publisher 3', 'Address 3', 'Contact Info 3');

INSERT INTO Book (id, name, description, publisher_id, ISBN, quantity, price)
VALUES
(1, 'Book 1', 'Description 1', 1, '1234', 10, 15.99);

INSERT INTO Category (id, name, description)
 VALUES
(1, 'Fiction', 'Books of fictional genre');

INSERT INTO Author (id, name, birth_date, nationality)
VALUES
(1, 'F. Scott Fitzgerald', TO_DATE('2003/05/03', 'yyyy/mm/dd '), 'American');
```

```
INSERT INTO Customer (id, name, address, phone_number, email)
 VALUES
 (1, 'John Doe', '456 Elm Street, Anytown', '555-1234', 'johndoe@example.com');

INSERT INTO “Order” (id, customer_id, order_date, order_status, total_price)
VALUES
(1, 1, TO_DATE('2003/05/03', 'yyyy/mm/dd ') , 'Shipped', 159.90);

INSERT INTO Payment (id, order_id, payment_date, payment_method, payment_amount)
VALUES
(1, 1, TO_DATE('2003/05/03', 'yyyy/mm/dd '), 'Credit Card', 159.90);

INSERT INTO Employee (id, name, job_title, department, hire_date, contact_information)
VALUES
(3, 'Name 3', 'Job title 3', 'Department 3', TO_DATE('2003/05/03', 'yyyy/mm/dd'),
'name3@example.com');

INSERT INTO Book_Category (book_id, category_id)
VALUES
 (1, 1);
INSERT INTO Book_Author (book_id, author_id)
VALUES
(1, 1);
INSERT INTO Order_Book (order_id, book_id)
 VALUES
 (1, 1);
INSERT INTO Order_Employee (order_id, employee_id)
VALUES
(1, 1);
```

Writing SQL DML Statements to reach information

One statement including subquery

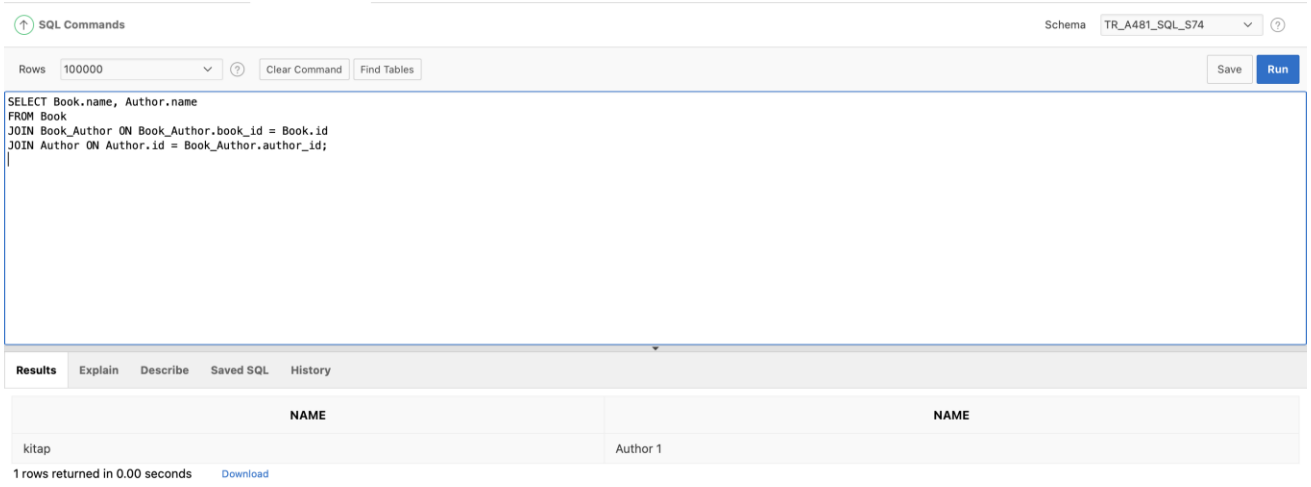


```
SELECT *
FROM Book
WHERE publisher_id IN (SELECT id FROM Publisher WHERE name = 'Publisher 1');
```

ID	NAME	DESCRIPTION	PUBLISHER_ID	ISBN	QUANTITY	PRICE
4	Book 4	Description of Book 4	1	ISBN4	40	49.99

1 rows returned in 0.00 seconds

One statement including join

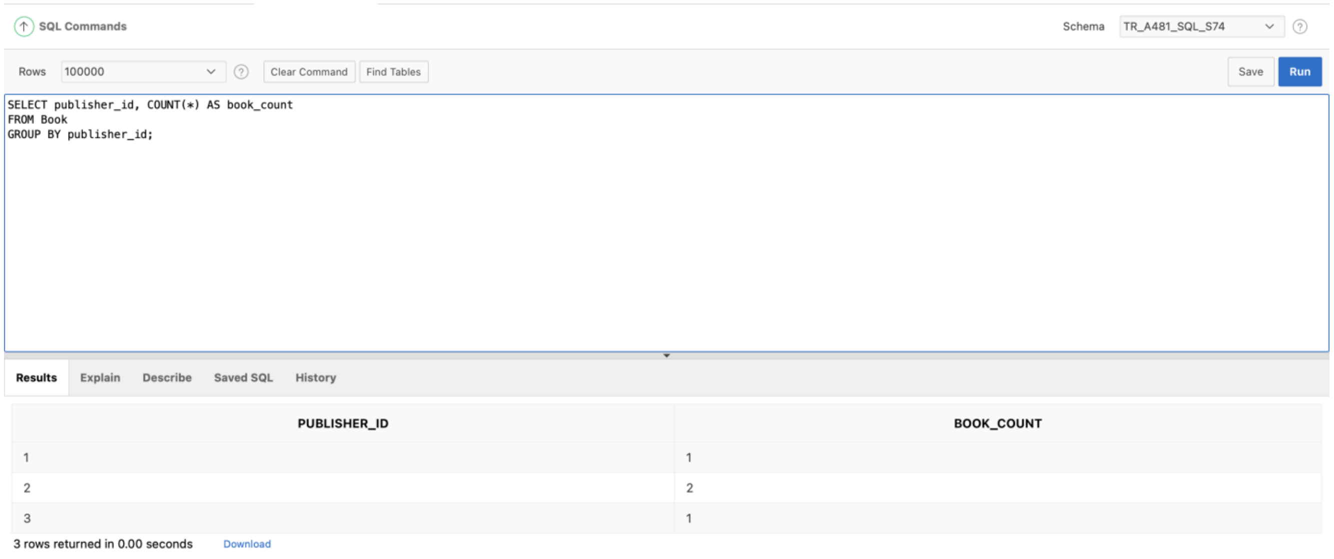


```
SELECT Book.name, Author.name
FROM Book
JOIN Book_Author ON Book_Author.book_id = Book.id
JOIN Author ON Author.id = Book_Author.author_id;
```

NAME	NAME
kitap	Author 1

1 rows returned in 0.00 seconds

One statement including group by

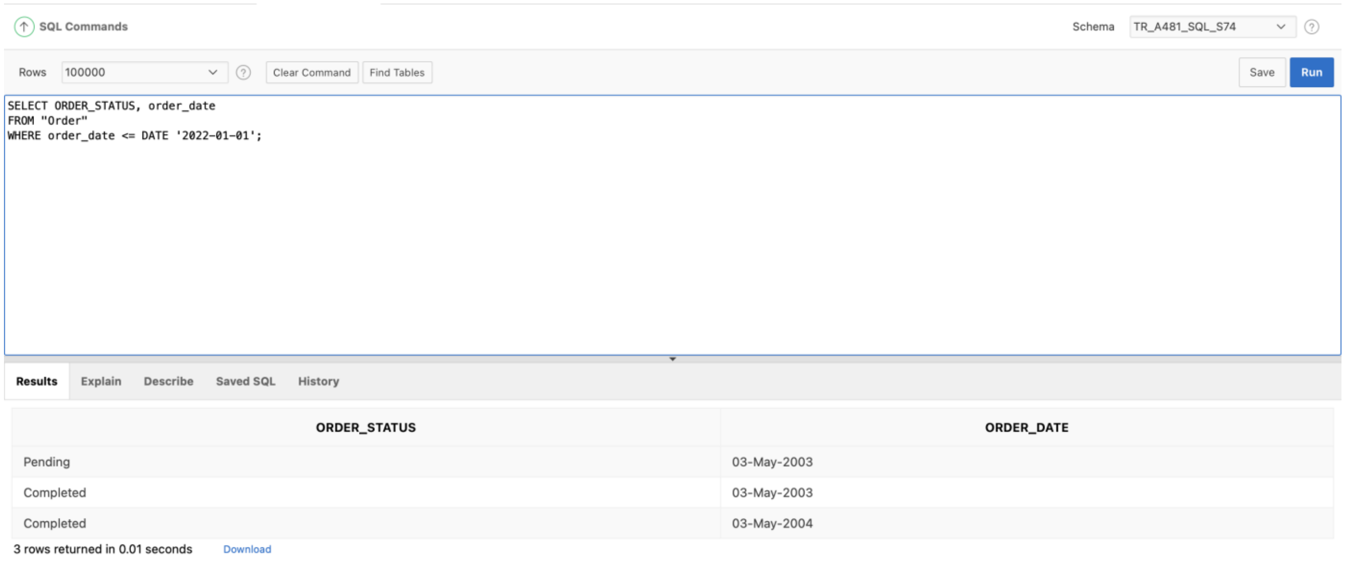


```
SELECT publisher_id, COUNT(*) AS book_count
FROM Book
GROUP BY publisher_id;
```

PUBLISHER_ID	BOOK_COUNT
1	1
2	2
3	1

3 rows returned in 0.00 seconds

One statement including date function

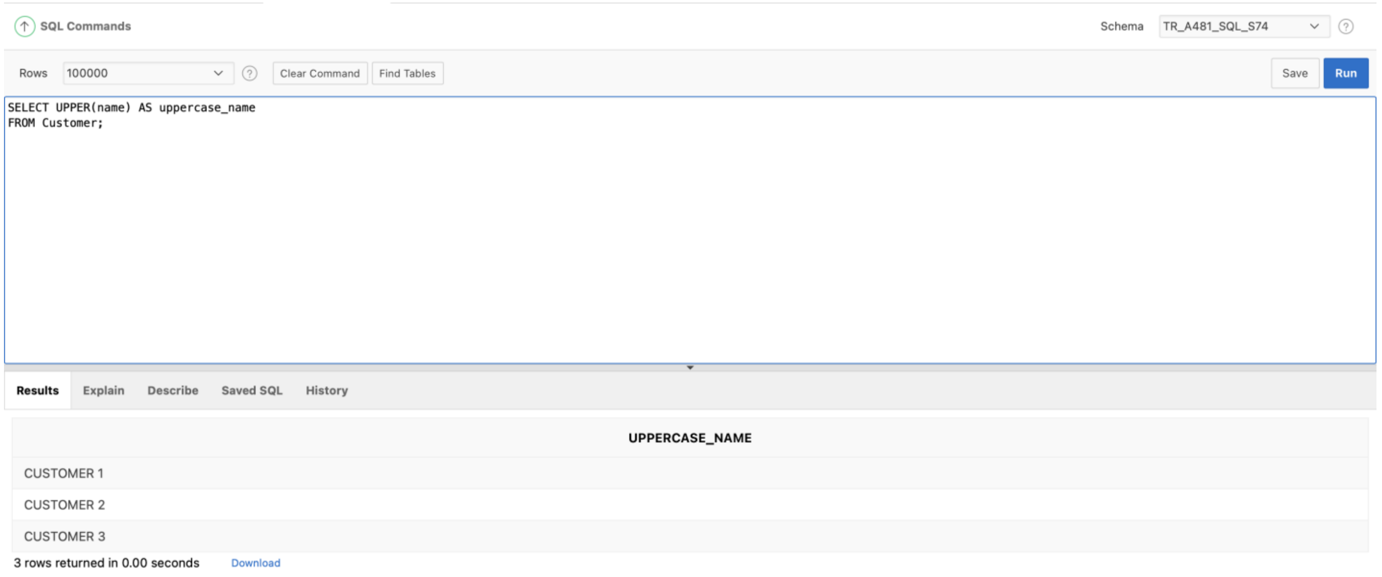


```
SELECT ORDER_STATUS, order_date
FROM "Order"
WHERE order_date <= DATE '2022-01-01';
```

ORDER_STATUS	ORDER_DATE
Pending	03-May-2003
Completed	03-May-2003
Completed	03-May-2004

3 rows returned in 0.01 seconds

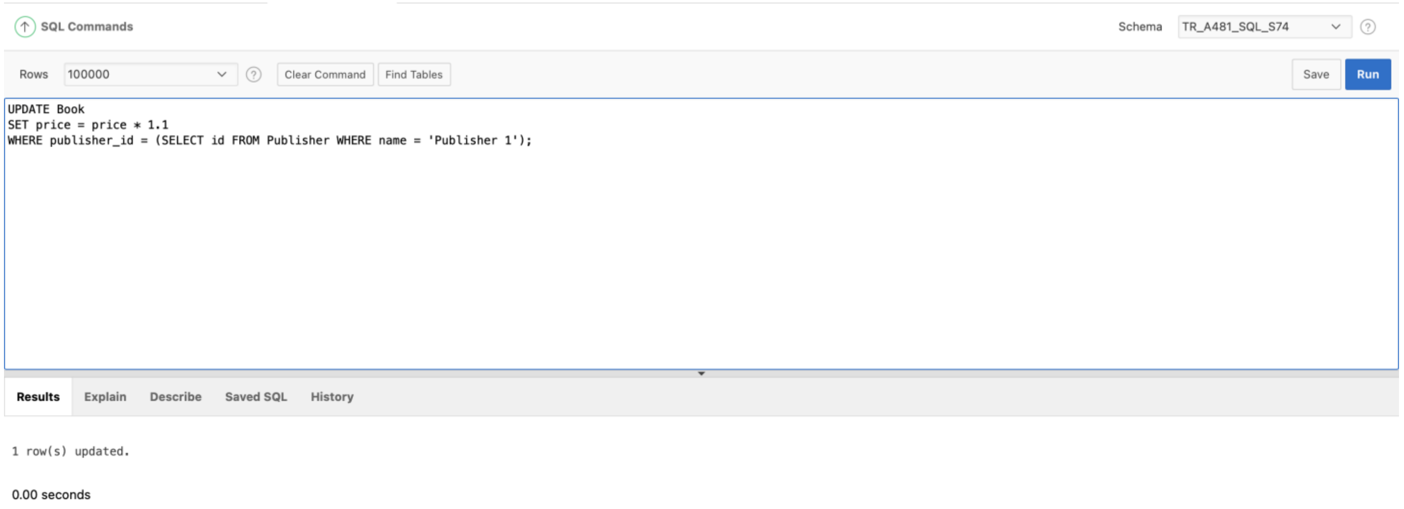
One statement including character function



```
SELECT UPPER(name) AS uppercase_name
FROM Customer;
```

```
UPPERCASE_NAME
CUSTOMER 1
CUSTOMER 2
CUSTOMER 3
3 rows returned in 0.00 seconds
```

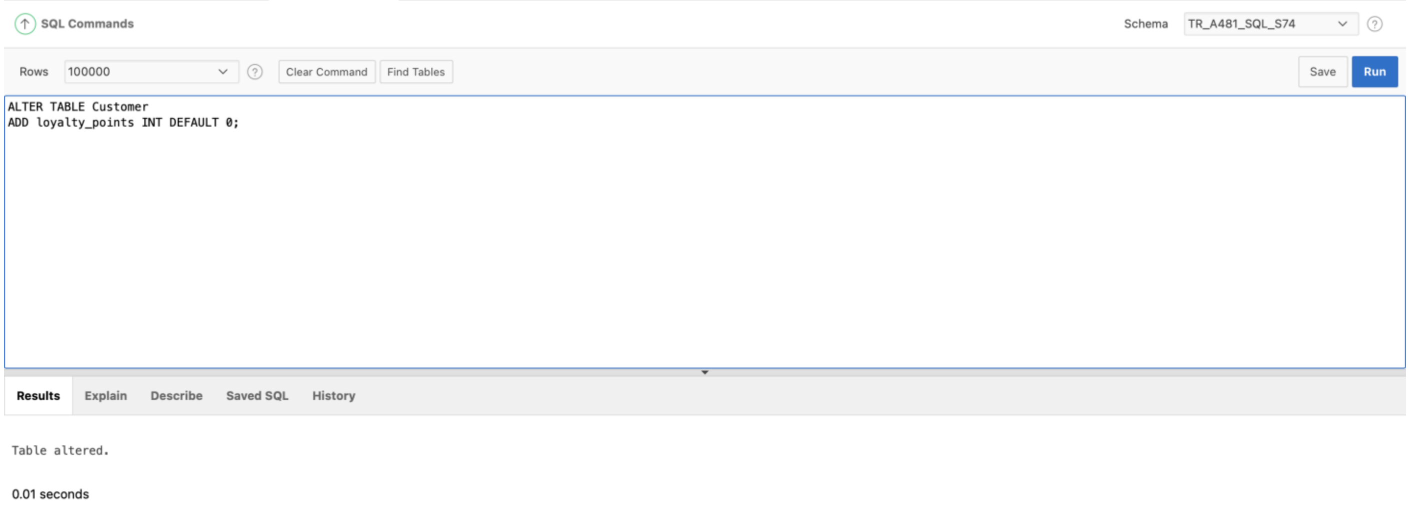
One statement including update



```
UPDATE Book
SET price = price * 1.1
WHERE publisher_id = (SELECT id FROM Publisher WHERE name = 'Publisher 1');
```

```
1 row(s) updated.

0.00 seconds
```

One statement including alter table



```
ALTER TABLE Customer
ADD loyalty_points INT DEFAULT 0;
```

```
Table altered.
```

0.01 seconds