**# programme: Database Testing [SENG8075]**

**# Section:3**

**# Submitted to: Professor Andy Chow**

**# project partners:**

**1. Akida Laliwala [8975812]**

**2. Prachi Thakral [8973781]**

**3. Divtej Pal Singh [8962715]**

**# Duties of team member**

--> Creation of tables and Insertion till books table is done by \*\***Divtej Pal Singh**\*\*.

--> Insertion into remaining table and CRUD Operation / DDL & DML is done by \*\***Prachi Thakral**\*\*.

--> Solution of asked requirements and typescript is done by \*\***Akida Laliwala**\*\*.

**#Creation of Table**

#Author table:

```create table authors (author\_id int primary key,

first\_name varchar (20),

last\_name varchar(20),

email\_address varchar(20));```

| # Attributes | # Type | # Description |

|---------------|--------------|------------------------|

| author\_id | int | Primary key |

| first\_name | varchar (20) | Author’s first name |

| last\_name | varchar (20) | Author’s last name |

| email\_address | varchar (20) | Author’s email address |

#publishers table

```create table publishers (publisher\_id int primary key,

publisher\_name varchar (50));```

| Attributes | Type | Description |

|---------------|--------------|---------------------------|

| publisher\_id | int | Primary key |

| publisher\_name| varchar (50) | Publisher’s name |

#books table

```create table books (book\_id int primary key,

book\_title varchar (30),

book\_genre varchar(20),

book\_type varchar(20),

publication\_date date,

price decimal(10,2),

author\_id int, foreign key (author\_id) references authors(author\_id),

publisher\_id int, foreign key (publisher\_id) references publishers(publisher\_id),

isbn varchar(15));```

| # Attributes | # Type | # Description |

|------------------|----------------|--------------------------|

| book\_id | int | Primary key |

| book\_title | varchar (30) | Title of the book |

| book\_genre | varchar (20) | Book’s genre |

| book\_type | varchar (20) | ‘physical-book’or’e-book’|

Or ‘Audio-book’ |

| Publication\_date | Date | Book’s publication date |

| price | Decimal (10,2) | Book’s price |

| author\_id | int | Foreign key |

| publisher\_id | int | Foreign key |

| isbn | varchar (15) | Book’s isbn code |

#customer table

```create table customer (customer\_id int primary key,

first\_name varchar(20),

last\_name varchar(20),

contact\_number varchar(20),

address varchar(100),

total\_spent\_amount decimal(10,2),

registration\_date date );```

| # Attributes | # Type | # Description |

|--------------------|----------------|------------------------------|

| customer\_id | int | Primary key |

| first\_name | varchar (20) | Customer’s first name |

| last\_name | varchar (20) | Customer’s last name |

| contact\_number | varchar (20) | Customer’s contact number |

| address | Varchar (100) | Customer’s address |

| total\_spent\_amount | Decimal (10,2) | Total Spent amountby Customer|

| registration\_date | Date | Customer’s registration date |

#orders table

```create table orders (order\_id int primary key,

customer\_id int, foreign key (customer\_id) references customer(customer\_id),

order\_date date,

bill\_amount decimal (10,2));```

| # Attributes| # Type | # Description |

|-------------|----------------|---------------|

| order\_id | int | Primary key |

| customer\_id | int | Foreign key |

| order\_date | date | Order’s date |

| bill\_amount | Decimal (10,2) | Bill’s amount |

#order\_item table

```create table order\_item (order\_item\_id int primary key,

order\_id int, foreign key (order\_id) references orders(order\_id),

book\_id int, foreign key (book\_id) references books(book\_id),

quantity int,

price decimal(10,2));```

| # Attributes | # Type | # Description |

|---------------|----------------|------------------|

| order\_item\_id | int | Primary key |

| order\_id | int | Foreign key |

| book\_id | int | Foreign key |

| quantity | int | Quantity of books|

| Price | Decimal (10,2) | Book’s price |

#review table

```create table review (review\_id int primary key,

customer\_id int, foreign key (customer\_id) references customer(customer\_id),

book\_id int, foreign key (book\_id) references books(book\_id),

review\_date date,

review\_comment text,

ratings int);```

| # Attributes | # Type | Description |

|----------------|--------|---------------------------|

| review\_id | int | Primary key |

| customer\_id | int | Foreign key |

| book\_id | int | Foreign key |

| review\_date | date | Date of feedback |

| review\_comment | Text | Feedback of customer |

| ratings | int | Ratings given by customer |

**# Insertion into the tables**

#Author table:

```insert into authors (author\_id,first\_name,last\_name,email\_address)

values (1,'Emily','johnson','emily@author.com'),

(2,'Rebecca','Yarros','rebros@author.com'),

(3,'Katherine','paterson','katherin@author.com'),

(4,'Kristin','Hannah','weintra@stmartin.com'),

(5,'Ashley','Elston','rstevenson@prh.com'),

(6,'sarah J.','Mass','sarah@stmartin.com'), (7,'Freida','McFadden','freida@author.com'),

(8,'Victoria','Aveyard','vicyard@author.com'),

(9,'Walter',' Isaacson','water@acson.com'),

(10,'Jojo','Moyes','jojomoyes@author.com'),

(11,'Gregg','Olsen','gregg@author.com'), (12,'Leigh','Bardugo','leigh@author.com'),

(13,'Greg','Iles','giles34@gmail.com'), (14,'Armistead','Maupin','john@armistead.com'), (15,'David','Levithan','david@levithan.com'), (16,'April','Henry','April@mysteries.com'), (17,'Cory','Doctorow','doc@craphound.com'), (18,'Chanel','Cleeton','chanelc@cleeton.com'),

(19,'J.R.R','Tolkien','jrrtol@gmail.com'),

(20,'Nalini','Nalini','nalini@singh.com'), (21,'Alyson','Schafer','hello@schafer.com'), (22,'Dennis','Lee','dennis.lee@gmail.com'), (23,'Ryan','Andrews','ryan1994@gmail.com'), (24,'Karen','Inglis','kpinglis@press.com');```

| author\_id | first\_name | last\_name, | email\_address |

|-----------|------------|------------|----------------------|

| 1 | Emily | Johnson | emily@author.com |

| 2 | Rebecca | Yarros | rebros@author.com |

| 3 | Katherine | Paterson | katherin@author.com |

| 4 | Kristin | Hannah | weintra@stmartin.com |

| 5 | Ashley | Elston | rstevenson@prh.com |

| 6 | Sarah J. | Mass | sarah@stmartin.com |

| 7 | Freida | McFadden | freida@author.com |

| 8 | Victoria | Aveyard | vicyard@author.com |

| 9 | Walter | Isaacson | water@acson.com |

| 10 | Jojo | Moyes | jojomoyes@author.com |

| 11 | Gregg | Olsen | gregg@author.com |

| 12 | Leigh | Bardugo | leigh@author.com |

| 13 | Greg | Iles | giles34@gmail.com |

| 14 | Armistead | Maupin | john@armistead.com |

| 15 | David | Levithan | david@levithan.com |

| 16 | April | Henry | April@mysteries.com |

| 17 | Cory | Doctorow | doc@craphound.com |

| 18 | Chanel | Cleeton | chanelc@cleeton.com |

| 19 | J.R.R | Tolkien | jrrtol@gmail.com |

| 20 | Nalini | Nalini | nalini@singh.com |

| 21 | Alyson | Schafer | hello@schafer.com |

| 22 | Dennis | Lee | dennis.lee@gmail.com |

| 23 | Ryan | Andrews | ryan1994@gmail.com |

| 24 | Karen | Inglis | kpinglis@press.com |

#publishers table

```insert into publishers (publisher\_id,publisher\_name)

values (1,'Penguin Publishing Group'),

(2, 'Red Tower Books'),

(3,'HarperCollins'),

(4,'St. Martins Press'),

(5,'Pamela Dorman Books'),

(6,'Bloomsburry Publishing'),

(7,'Poisoned Pen Press'),

(8,'HarperTeen'),

(9,'Simon & Schuster'),

(10,'A Reeses Book Club Pick'),

(11,'Thomas & Mercer'),

(12,'Imprint'),

(13,'William Morrow Paperbacks'),

(14,'Harper Perennial'),

(15,'Ember'),

(16,'Square Fish'),

(17,'First Second'),

(18,'Berkley'),

(19,'HarperCollinsChildren’sBooks'),

(20,'Berkley'),

(21,'Collins'),

(22,'HarperCollins'),

(23,'First Second'),

(24,'Well Said Press');```

|publisher\_id | publisher\_name |

|--------------|--------------------------------|

| 1 | Penguin Publishing Group |

| 2 | Red Tower Books |

| 3 | HarperCollins |

| 4 | St. Martins Press |

| 5 | Pamela Dorman Books |

| 6 | Bloomsbury Publishing |

| 7 | Poisoned Pen Press |

| 8 | HarperTeen |

| 9 | Simon & Schuster |

| 10 | A Reese's Book Club Pick |

| 11 | Thomas & Mercer |

| 12 | Imprint |

| 13 | William Morrow Paperbacks |

| 14 | Harper Perennial |

| 15 | Ember |

| 16 | Square Fish |

| 17 | First Second |

| 18 | Berkley |

| 19 | HarperCollins Children’s Books |

| 20 | Berkley (duplicated) |

| 21 | Collins |

| 22 | HarperCollins (duplicated) |

| 23 | First Second (duplicated) |

| 24 | Well Said Press |

#books table

```insert into books (book\_id,book\_title,book\_genre,book\_type,publication\_date,price,author\_id,publisher\_id,isbn)

values (1,'Funny Story','Romance','physical','2022-01-01',4.31,1,1,'978-0593441282'),

(2,'Fourth wing','Fantacy','physical','2023-05-02',24,2,2,'9781649374042'),

(3,'Bridge to Terabithia','Fiction','audio book','2017-05-02',10.0,3,3,'9780064401845'),

(4,'The Women','Historical Fiction','e-book','2024-02-06',28,4,4,'1250178630'),

(5,'First Lie Wins','Thriller','e-book','2024-01-02',16,5,5,'0593492919'),

(6,'House of Flame and Shadow','Fantacy','Audio book','2024-01-30',20,6,6,'1635574102'),

(7,'The Teacher','Thriller','Physical book','2024-02-06',38,7,7,'1728296218'),

(8,'Red Queen','Fantasy','physical book','2015-02-10',20,8,8,'0062310631'),

(9,'Elon Musk','Realist Literature','Audio book','2023-09-23',35,9,9,'1982181281'),

(10,'The Giver of Stars','mystery','e-book','2019-10-19',28,10,10,'0593152263'),

(11,'If You Tell','Realist Literature','physical book','2019-12-01',28.40,11,11,'1542005221'),

(12,'Rule of Wolves','Fantasy','e-book','2021-03-30',26.13,12,12,'125014230X'),

(13,'The Bone Tree','Thriller','Physical book','2016-09-13',21.99,13,13,'978-0062311122'),

(14,'Goodbye Barbary Lane','Humor','e-book','2016-12-06',24.99,14,14,'978-0062563774'),

(15,'David Levithans Every Day','Romance','Audio book','2013-09-10',18.99,15,15,'978-0307931894'),

(16,'Girl, Stolen','Mystery','Physical book','2012-03-13',15.99,16,16,'978-0312674755'),

(17,'In Real Life','Realistic Fiction','e-book','2014-10-14',17.14,17,17,'978-1596436589'),

(18,'Next Year in Havana','Historical Fiction','Audio book','2018-02-16',19.74,18,18,'978-0399586682'),

(19,'The Hobbit','Fantasy','Physical book','2012-06-26',12.99,19,19,'978-0007458424'),

(20,'Archangels Blade','Romance','e-book','2011-09-06',9.99,20,20,'978-0425243916'),

(21,'Aint Misbehavin','Non-fiction','Audio book','2014-03-25',21.0,21,21,'978-1443427609'),

(22,'Garbage Delight','Poetry','Physical book','2012-05-22',10.79,22,22,'978-1443411554'),

(23,'This Was Our Pact','Graphic','e-book','2019-06-11',21.99,23,23,'978-1626720534'),

(24,'The Secret Lake','Adventure Fiction','Audio book','2011-08-04',10.55,24,24,'978-0956932303');```

| book\_id | book\_title | book\_genre | book\_type | publication\_date | price | author\_id | Publisher\_id | isbn |

|---------|---------------------------|--------------------|---------------|------------------|-------|-----------|--------------|----------------|

| 1 | Funny Story | Romance | physical | publication\_date | 4.31 | 1 | 1 | 978-0593441282 |

| 2 | Fourth Wing | Fantasy | physical | 2022-01-01 | 24.00 | 2 | 2 | 9781649374042 |

| 3 | Bridge to Terabithia | Fiction | audio book | 2023-05-02 | 10.00 | 3 | 3 | 9780064401845 |

| 4 | The Women | Historical Fiction | e-book | 2017-05-02 | 28.00 | 4 | 4 | 1250178630 |

| 5 | First Lie Wins | Thriller | e-book | 2024-02-06 | 16.00 | 5 | 5 | 0593492919 |

| 6 | House of Flame and Shadow | Fantasy | Audio book | 2024-01-02 | 20.00 | 6 | 6 | 1635574102 |

| 7 | The Teacher | Thriller | Physical book | 2024-01-30 | 38.00 | 7 | 7 | 1728296218 |

| 8 | Red Queen | Fantasy | physical book | 2024-02-06 | 20.00 | 8 | 8 | 0062310631 |

| 9 | Elon Musk | Realist Literature | Audio book | 2015-02-10 | 35.00 | 9 | 9 | 1982181281 |

| 10 | The Giver of Stars | Mystery | e-book | 2023-09-23 | 28.00 | 10 | 10 | 0593152263 |

| 11 | If You Tell | Realist Literature | physical book | 2019-10-19 | 28.40 | 11 | 11 | 1542005221 |

| 12 | Rule of Wolves | Fantasy | e-book | 2019-12-01 | 26.13 | 12 | 12 | 125014230X |

| 13 | The Bone Tree | Thriller | Physical book | 2021-03-30 | 21.99 | 13 | 13 | 978-0062311122 |

| 14 | Goodbye Barbary Lane | Humor | e-book | 2016-09-13 | 24.99 | 14 | 14 | 978-0062563774 |

| 15 | David Levithans Every Day | Romance | Audio book | 2016-12-06 | 18.99 | 15 | 15 | 978-0307931894 |

| 16 | Girl, Stolen | Mystery | Physical book | 2013-09-10 | 15.99 | 16 | 16 | 978-0312674755 |

| 17 | In Real Life | Realistic Fiction | e-book | 2012-03-13 | 17.14 | 17 | 17 | 978-1596436589 |

| 18 | Next Year in Havana | Historical Fiction | Audio book | 2014-10-14 | 19.74 | 18 | 18 | 978-0399586682 |

| 19 | The Hobbit | Fantasy | Physical book | 2018-02-16 | 12.99 | 19 | 19 | 978-0007458424 |

| 20 | Archangels Blade | Romance | e-book | 2012-06-26 | 9.99 | 20 | 20 | 978-0425243916 |

| 21 | Aint Misbehavin | Non-fiction | Audio book | 2011-09-06 | 21.00 | 21 | 21 | 978-1443427609 |

| 22 | Garbage Delight | Poetry | Physical book | 2014-03-25 | 10.79 | 22 | 22 | 978-1443411554 |

| 23 | This Was Our Pact | Graphic | e-book | 2012-05-22 | 21.99 | 23 | 23 | 978-1626720534 |

| 24 | The Secret Lake | Adventure Fiction | Audio book | 2019-06-11 | 10.55 | 24 | 24 | 978-0956932303 |

#customer table

```insert into customer (customer\_id,first\_name,last\_name,contact\_number,address,total\_spent\_amount,registration\_date) values (101,'Akida','Aazam',437-9632145,'41,king street,waterloo,ON',700.70,'2010-01-25'), (102,'Prachi','Thakral',437-1236985,'48,erb street,waterloo,ON',500.50,'2011-02-20'),

(103,'Divtej','Singh',752-1239687,'56,philip street,waterloo,ON',300.30,'2012-03-27'),

(104,'Swati','Sansarwal',720-1254789,'125,queen street,waterloo,ON',450.30,'2013-05-19'),

(105,'Mayank','Sharma',778-9875463,'78,lester street,waterloo,ON',235.40,'2014-06-15'),

(106,'Kartik','Singh',798-4587962,'88,albert street,montreal,QC',550.30,'2015-07-17'),

(107,'Zaroon','Zunaid',451-4158796,'12,hickory street,montreal,QC',720.20,'2016-08-08'),

(108,'Kashaf','Murtuza',652-7895412,'10,mackay street,waterloo,ON',260.30,'2020-09-05'),

(109,'Michel','Wilson',478-1245789,'45,nocker street,montreal,QC',215.20,'2022-10-29'),

(110,'Rubby','Smith',458-5278964,'2,marshal street,montreal,QC',350.30,'2023-11-10'),

(111,'Kethrin','Johnson',965-8574961,'50,hithrow street,waterloo,ON',500.30,'2022-06-27'),

(112,'David','Wilson',958-4578961,'90,regina street,toronto,ON',400.30,'2023-03-30'),

(113,'Kiara','Davis',452-7854963,'98,Columbia street,Hamilton,ON',600.60,'2022-12-27'),

(114,'Nayra','Patinson',658-5285749,'86,Linclon street,Brandford,ON',250.70,'2023-05-03'),

(115,'Kunj','Singh',124-8574961,'89,water street,Cambridge,ON',374.17,'2022-11-11');```

| customer\_id | first\_name | last\_name | contact\_number | address | total\_spent\_amount | registration\_date |

|:-----------:|:----------:|:---------:|:--------------:|:------------------------------:|:------------------:|:-----------------:|

| 101 | Akida | Aazam | 437-9632145 | 41,king street,waterloo,ON | 700.70 2010-01-25 |

| 102 | Prachi | Thakral | 437-1236985 | 48,erb street,waterloo,ON | 500.50 | 2011-02-20 |

| 103 | Divtej | Singh | 752-1239687 | 56,philip street,waterloo,ON | 300.30 | 2012-03-27 |

| 104 | Swati | Sansarwal | 720-1254789 | 125,queen street,waterloo,ON | 450.30 | 2013-05-19 |

| 105 | Mayank | Sharma | 778-9875463 | 78,lester street,waterloo,ON | 235.40 | 2014-06-15 |

| 106 | Kartik | Singh | 798-4587962 | 88,albert street,montreal,QC | 550.30 | 2015-07-17 |

| 107 | Zaroon | Zunaid | 451-4158796 | 12,hickory street,montreal,QC | 720.20 | 2016-08-08 |

| 108 | Kashaf | Murtuza | 652-7895412 | 10,mackay street,waterloo,ON | 260.30 | 2020-09-05 |

| 109 | Michel | Wilson | 478-1245789 | 45,nocker street,montreal,QC | 215.20 | 2022-10-29 |

| 110 | Rubby | Smith | 458-5278964 | 2,marshal street,montreal,QC | 350.30 | 2023-11-10 |

| 111 | Kethrin | Johnson | 965-8574961 | 50,hithrow street,waterloo,ON | 500.30 | 2022-06-27 |

| 112 | David | Wilson | 958-4578961 | 90,regina street,toronto,ON | 400.30 | 2023-03-30 |

| 113 | Kiara | Davis | 452-7854963 | 98,Columbia street,Hamilton,ON | 600.60 | 2022-12-27 |

| 114 | Nayra | Patinson | 658-5285749 | 86,Linclon street,Brandford,ON | 250.70 | 2023-05-03 |

| 115 | Kunj | Singh | 124-8574961 | 89,water street,Cambridge,ON | 374.17 | 2022-11-11 |

#orders table

```insert into orders (order\_id, customer\_id, order\_date, bill\_amount)

values (001,101,'2023-01-25',100.11),

(002,102,'2022-02-02',50.11),

(003,103,'2021-03-14',120.10),

(004,104,'2020-04-18',45.11),

(005,105,'2019-05-12',70.17),

(006,106,'2018-06-07',110.11),

(007,107,'2023-07-11',60.16),

(008,108,'2022-08-03',55.15),

(009,109,'2021-09-01',105.05),

(010,110,'2020-10-04',75.75),

(011,111,'2019-11-09',30.11),

(012,112,'2018-12-12',40.40),

(013,113,'2023-01-13',111.11),

(014,114,'2022-02-22',99.99),

(015,115,'2021-02-19',130.13);```

| order\_id | customer\_id | order\_date | bill\_amount |

|:--------:|:-----------:|:----------:|:-----------:|

| 001 | 101 | 2023-01-25 | 100.11 |

| 002 | 102 | 2022-02-02 | 50.11 |

| 003 | 103 | 2021-03-14 | 120.10 |

| 004 | 104 | 2020-04-18 | 45.11 |

| 005 | 105 | 2019-05-12 | 70.17 |

| 006 | 106 | 2018-06-07 | 110.11 |

| 007 | 107 | 2023-07-11 | 60.16 |

| 008 | 108 | 2022-08-03 | 55.15 |

| 009 | 109 | 2021-09-01 | 105.05 |

| 010 | 110 | 2020-10-04 | 75.75 |

| 011 | 111 | 2019-11-09 | 30.11 |

| 012 | 112 | 2018-12-12 | 40.40 |

| 013 | 113 | 2023-01-13 | 111.11 |

| 014 | 114 | 2022-02-22 | 99.99 |

| 015 | 115 | 2021-02-19 | 130.13 |

#order\_item table

```insert into order\_item (order\_item\_id, order\_id, book\_id, quantity, price)

values (01, 001, 2, 2, 48),

(02, 002, 5, 1, 16),

(03, 003, 3, 2, 20),

(04, 004, 7, 1, 38),

(05, 005, 11, 1, 28.40),

(06, 006, 17, 1, 17.14),

(07, 007, 20, 2, 20),

(08, 008, 19, 3, 42),

(09, 009, 12, 1, 26.13),

(10, 010, 14, 2, 50),

(11, 011, 24, 5, 53),

(12, 012, 21, 2, 42),

(13, 013, 16, 3, 51),

(14, 014, 9, 4, 140),

(15, 015, 8, 2, 40);```

| order\_item\_id | order\_id | book\_id | quantity | price |

|:-------------:|:--------:|:-------:|:--------:|:------:|

| 01 | 001 | 2 | 2 | 48.00 |

| 02 | 002 | 5 | 1 | 16.00 |

| 03 | 003 | 3 | 2 | 20.00 |

| 04 | 004 | 7 | 1 | 38.00 |

| 05 | 005 | 11 | 1 | 28.40 |

| 06 | 006 | 17 | 1 | 17.14 |

| 07 | 007 | 20 | 2 | 20.00 |

| 08 | 008 | 19 | 3 | 42.00 |

| 09 | 009 | 12 | 1 | 26.13 |

| 10 | 010 | 14 | 2 | 50.00 |

| 11 | 011 | 24 | 5 | 53.00 |

| 12 | 012 | 21 | 2 | 42.00 |

| 13 | 013 | 16 | 3 | 51.00 |

| 14 | 014 | 9 | 4 | 140.00 |

| 15 | 015 | 8 | 2 | 40.00 |

#review table

```insert into review (review\_id, customer\_id, book\_id, review\_date, review\_comment, ratings)

values (1,101,2,'2023-09-25','Excellent Book!',4),

(2,102,5,'2023-01-02','Nice Book!',3),

(3,103,3,'2022-05-15','Quite slow!',2),

(4,104,7,'2023-05-25','Nice book to Recommend',5),

(5,105,11,'2022-09-22','Fentastic story!',4),

(6,106,17,'2021-08-10','Quite boring!',1),

(7,107,2,'2023-12-12','Good Book to read!',3),

(8,108,19,'2023-03-16','Nice Book!',3),

(9,109,12,'2022-11-21','Fentastic story!',4),

(10,110,14,'2021-12-26','Quite boring!',1),

(11,111,24,'2020-02-10','Good Book to read!',3),

(12,112,21,'2019-12-12','Quite slow!',2),

(13,113,16,'2023-07-15','Nice book to Recommend',5),

(14,114,9,'2023-01-27','Excellent Book!',4),

(15,115,8,'2022-08-20','Nice Book!',3);```

| review\_id | customer\_id | book\_id | review\_date | review\_comment | ratings |

| 1 | 101 | 2 | 2023-09-25 | Excellent Book! | 4 |

| 2 | 102 | 5 | 2023-01-02 | Nice Book! | 3 |

| 3 | 103 | 3 | 2022-05-15 | Quite slow! | 2 |

| 4 | 104 | 7 | 2023-05-25 | Nice book to Recommend | 5 |

| 5 | 105 | 11 | 2022-09-22 | Fantastic story! | 4 |

| 6 | 106 | 17 | 2021-08-10 | Quite boring! | 1 |

| 7 | 107 | 2 | 2023-12-12 | Good Book to read! | 3 |

| 8 | 108 | 19 | 2023-03-16 | Nice Book! | 3 |

| 9 | 109 | 12 | 2022-11-21 | Fantastic story! | 4 |

| 10 | 110 | 14 | 2021-12-26 | Quite boring! | 1 |

| 11 | 111 | 24 | 2020-02-10 | Good Book to read! | 3 |

| 12 | 112 | 21 | 2019-12-12 | Quite slow! | 2 |

| 13 | 113 | 16 | 2023-07-15 | Nice book to Recommend | 5 |

| 14 | 114 | 9 | 2023-01-27 | Excellent Book! | 4 |

| 15 | 115 | 8 | 2022-08-20 | Nice Book! | 3 |

**# DDL Operations**

1. Creation of database:

```create database online\_book\_store;```

1. Alter table in database:

```drop table public.books;```

1. Drop database:

```drop database online\_book\_store;```

**# DML Operations/ CRUD Operation**

1. Insert into customer table:

```insert into customer (customer\_id,first\_name,last\_name,contact\_number,address,total\_spent\_amount,registration\_date) values (116,'Edward','Coolin',437-8574259,'41,Lenster street,waterloo,ON',50.70,'2024-05-25'); ```

1. Read from customer table:

```select \* from customer where customer\_id = 116;```

1. Update into customer table:

``` update customer

set total\_spent\_amount = 100

where customer\_id = 116;```

1. Delete from customer table:

``` delete from customer where customer\_id = 116;```

* Updating into orders table:

```update orders

set order\_date = case

when order\_id = 001 then '2023-12-31'

when order\_id = 002 then '2024-01-01'

when order\_id = 008 then '2023-11-05'

when order\_id = 013 then '2023-11-17'

when order\_id = 014 then '2023-09-11'

ELSE order\_date

END

WHERE order\_id IN (001,002,008,013,014);```

**# Solution of asked Requirements**

1. Details of authers of same genre book published in the last 10 years.

```select a.author\_id,a.first\_name,a.last\_name, count(book\_id) as BookCount

from books b

join authors a on b.author\_id = a.author\_id

where b.book\_genre = 'Fantasy' and publication\_date >= current\_date - interval '10 years'

group by a.author\_id,a.first\_name,a.last\_name

having count(b.book\_id) > 0;```

| ID | author\_id | first\_name | last\_name | bookcount |

|:---:|:---------:|:----------:|:---------:|:--------:|

| 1 | 2 | Rebecca | Yarros | 1 |

| 2 | 6 | sarah J. | Mass | 1 |

| 3 | 8 | Victoria | Aveyard | 1 |

| 4 | 12 | Leigh | Bardugo | 1 |

1. Loyal customer who has spent more than $51 in the last year

```select c.customer\_id, c.first\_name, c.last\_name, sum(o.bill\_amount) as TotalSpent

from customer c

join orders o on c.customer\_id = o.customer\_id

where o.order\_date >= current\_date - interval '1 year'

group by c.customer\_id

having sum(o.bill\_amount) > 51;```

| ID | customer\_id | first\_name | last\_name | totalspent |

|:--:|:----------:|:---------:|:--------:|:---------:|

| 1 | 101 | Akida | Aazam | 100.11 |

| 2 | 114 | Nayra | Patinson | 99.99 |

| 3 | 108 | Kashaf | Murtuza | 55.15 |

| 4 | 113 | Kiara | Davis | 111.11 |

| 5 | 107 | Zaroon | Zunaid | 60.16 |

1. Books which have better user ratings than average

```select b.book\_id, b.book\_title, avg(r.ratings) as AvgRating

from Books b

join review r on b.book\_id = r.book\_id

group by b.book\_id

having avg(r.ratings) > (select avg(ratings) from review);```

| ID | book\_id | book\_title | avgrating |

|:---:|:-------:|:--------------:|:-------------------:|

| 1 | 2 | Fourth wing | 3.500000000000000000 |

| 2 | 7 | The Teacher | 5.000000000000000000 |

| 3 | 11 | If You Tell | 4.000000000000000000 |

| 4 | 9 | Elon Musk | 4.000000000000000000 |

| 5 | 16 | Girl, Stolen | 5.000000000000000000 |

| 6 | 12 | Rule of Wolves | 4.000000000000000000 |

1. Most popular genre by sales

```select book\_genre, sum(order\_item.quantity) as total\_sales

from books

join order\_item on books.book\_id = order\_item.book\_id

group by book\_genre

order by total\_sales desc

limit 5;```

| ID | Book\_genre | total\_sales|

|----|----------------------|------------|

| 1 |Fantasy | 8 |

| 2 |Realist Literature | 5 |

| 3 |Adventure Fiction | 5 |

| 4 |Mystery | 3 |

| 5 |Thriller | 2 |

1. 10 most recent review posted by customers

```select r.review\_id, r.book\_id, b.book\_title, r.customer\_id, c.first\_name, r.ratings, r.review\_comment, r.review\_date

from review r

join books b on r.book\_id = b.book\_id

join customer c on r.customer\_id = c.customer\_id

order by r.review\_date desc

limit 10;```

|ID |review\_id | book\_id| book\_title |customer\_id|first\_name|ratings |review\_comments|review\_date

|1| 7 | 2 |Fourth wing | 107 | Zaroon | 3 | Good Book to read! | 2023-12-12 |

|2| 1 | 2 | Fourth wing | 101 | Akida | 4 | Excellent Book! | 2023-09-25 |

|3| 13 | 16 | Girl, Stolen | 113 | Kiara | 5 | Nice book to Recommend | 2023-07-15 |

|4| 4 | 7 | The Teacher | 104 | Swati | 5 | Nice book to Recommend | 2023-05-25 |

|5| 8 | 19 | The Hobbit | 108 | Kashaf | 3 | Nice Book! | 2023-03-16 |

|6| 14 | 9 | Elon Musk | 114 | Nayra | 4 | Excellent Book! | 2023-01-27 |

|7| 2 | 5 | First Lie Wins | 102 | Prachi | 3 | Nice Book! | 2023-01-02 |

|8| 9 | 12 | Rule of Wolves | 109 | Michel | 4 | Fantastic story!

| 2022-11-21 |

|9| 5 | 11 | If You Tell | 105 | Mayank | 4 | Fantastic story! | 2022-09-22 |

|10| 15 | 8 | Red Queen | 115 | Kunj | 3 | Nice Book! | 2022-08-20 |

**# Type script interface**

* **connection.ts**

```import { Pool } from 'pg';

const pool = new Pool({

user: 'postgres',

host: 'localhost',

database: 'postgres',

password: 'password',

port: 5432

});

pool.on('connect', (client) => {

client.query('SET search\_path TO public');

});

export default pool;```

* **customer.ts**

```import { Pool, QueryResult } from 'pg';

import pool from './connection';

const TABLE\_NAME = 'public.customer';

export interface customer\_detail {

customer\_id: number;

first\_name: string;

last\_name: string;

contact\_number: string;

address: string;

total\_spent\_amount: number;

registration\_date: Date;

}

export interface customers {

create(customer: customer\_detail): Promise<customer\_detail>;

read(customer\_id: number): Promise<customer\_detail | null>;

update(customer\_id: number, customer: Partial<customer\_detail>): Promise<customer\_detail>;

delete(customer\_id: number): Promise<boolean>;

}

export class customer\_store implements customers {

private pool: Pool;

constructor() {

this.pool = pool;

}

async create(customer: customer\_detail): Promise<customer\_detail> {

const query = `

INSERT INTO ${TABLE\_NAME} (customer\_id, first\_name, last\_name, contact\_number, address, total\_spent\_amount, registration\_date)

VALUES ($1, $2, $3, $4, $5, $6, $7)

RETURNING \*`;

const values = [customer.customer\_id, customer.first\_name, customer.last\_name, customer.contact\_number, customer.address, customer.total\_spent\_amount, customer.registration\_date];

const result: QueryResult = await this.pool.query(query, values);

return result.rows[0];

}

async read(customer\_id: number): Promise<customer\_detail | null> {

const query = 'SELECT \* FROM ${TABLE\_NAME} WHERE customer\_id = $1';

const result: QueryResult = await this.pool.query(query, [customer\_id]);

return result.rows[0] || null;

}

async update(customer\_id: number, customer: Partial<customer\_detail>): Promise<customer\_detail> {

const fields = Object.keys(customer).map((key, index) => `${key} = $${index + 2}`).join(', '); const values = Object.values(customer);

const query = `UPDATE ${TABLE\_NAME} SET ${fields} WHERE customer\_id = $1 RETURNING \*`;

const result: QueryResult = await this.pool.query(query, [customer\_id, ...values]);

return result.rows[0] || null;

}

async delete(customer\_id: number): Promise<boolean> {

const query = 'DELETE FROM ${TABLE\_NAME} WHERE customer\_id = $1';

const result: QueryResult = await this.pool.query(query, [customer\_id]);

return result.rowCount !== null && result.rowCount > 0;

}

}```

* **customer\_crud\_operation.ts**

```import pool from './connection';

import { customer\_store, customer\_detail } from './customer';

async function checkDatabaseAndTable() {

const client = await pool.connect();

try {

const result = await client.query('SELECT current\_user, current\_database()'); console.log('Current user:', result.rows[0].current\_user);

console.log('Current database:', result.rows[0].current\_database);

// Check if table exists

const tableCheck = await client.query(`

SELECT EXISTS (

SELECT FROM information\_schema.tables

WHERE table\_schema = 'public'

AND table\_name = 'customer'

);

`);

console.log('Customer table exists:', tableCheck.rows[0].exists);

// If table exists, check its structure

if (tableCheck.rows[0].exists) {

const tableStructure = await client.query(`

SELECT column\_name, data\_type

FROM information\_schema.columns

WHERE table\_name = 'customer';

`);

console.log('Table structure:', tableStructure.rows);

}

} catch (error) {

console.error('Error checking database and table:', error);

}finally {

client.release();

}

}

async function main() {

try {

await checkDatabaseAndTable();

const customer\_details = new customer\_store();

// Create a new customer

const new\_customer: customer\_detail = {

customer\_id: 116,

first\_name: 'John',

last\_name: 'Doe',

contact\_number: '123-4567890',

address: '123 Main St, Anytown, ON',

total\_spent\_amount: 0,

registration\_date: new Date('2024-01-01')

};

const created\_customer = await customer\_details.create(new\_customer); console.log('Created a new Customer:', created\_customer);

// Read a customer

const read\_customer = await customer\_details.read(created\_customer.customer\_id); console.log('Reading a Customer:', read\_customer);

// Update a customer

const updated\_customer = await customer\_details.update(created\_customer.customer\_id, { total\_spent\_amount: 100.50 });

console.log('Updated a Customer:', updated\_customer);

// Delete a customer

const deleted\_customer = await customer\_details.delete(created\_customer.customer\_id);

console.log('Deleted a Customer:', deleted\_customer);

}catch(error){

console.error('An error occurred:', error);

if (error instanceof Error) {

console.error('Error message:', error.message);

console.error('Error stack:', error.stack);}

}finally {

await pool.end();

}

}

main().catch(console.error);```

**# References**

<https://markdown.land/markdown-code-block>

<https://www.typescriptlang.org/docs/handbook/interfaces.html>