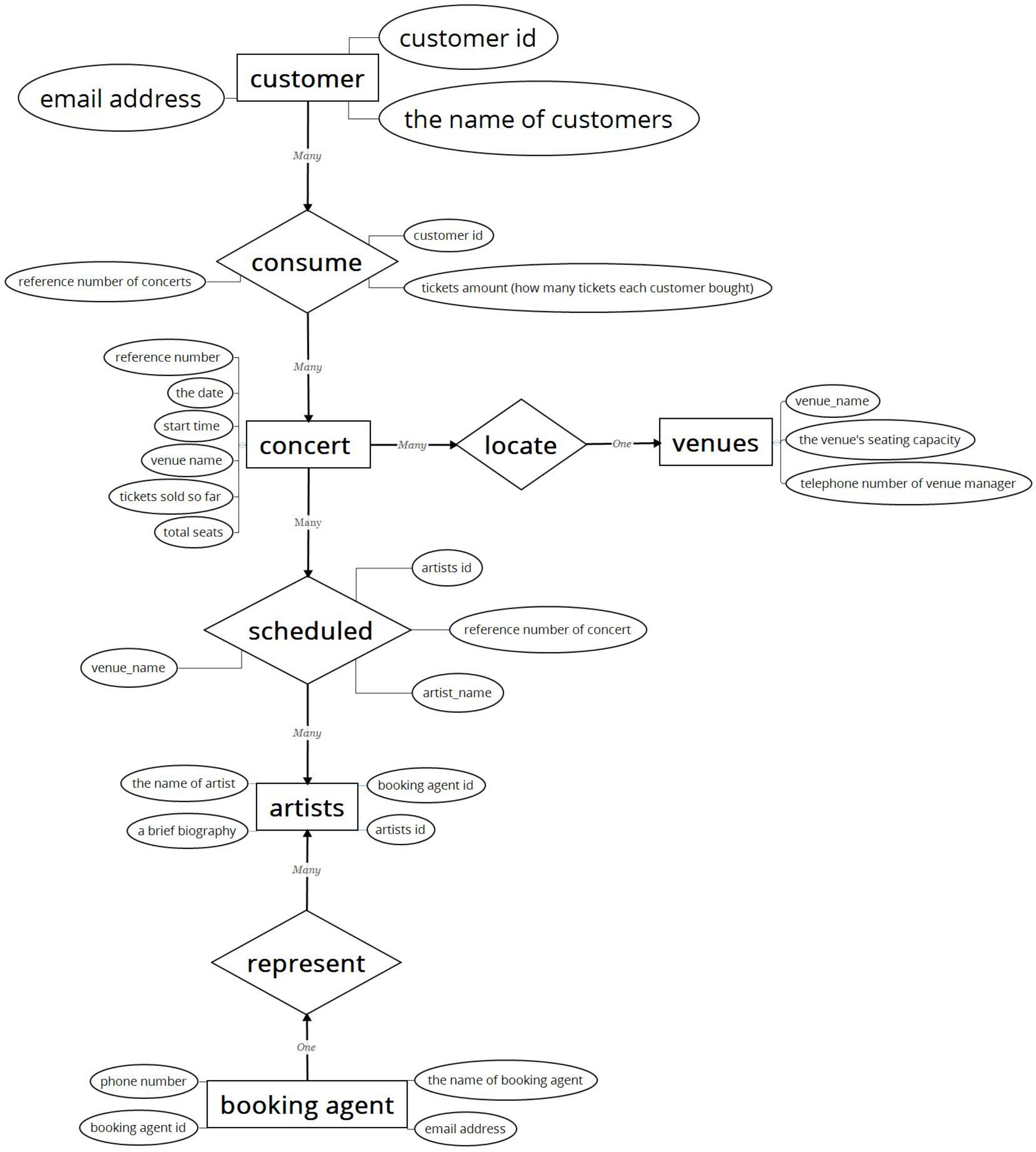
Answer Sheet of Database Course Work

Task 1): entity-relationship

diagram



Assumptions:

Table customer :

The contact detail of customers is an assumption. the attribute of a customer includes:

- a unique customer id;
- the email address of each customer;
- the name of each customer;

Table consume :

The whole table consume is an assumption which stores details of customer purchase details. Includes:

- customer ids reference to customer id in table customer;
- reference number of concerts reference to reference number in table concert;
- a column namely ticket_bought is to count how many tickets each customer bought for each concert;

Table concert:

Based on the problem statement:

- A column namely venue name is added to identify the venues where this concert is located;
- A column namely total seats is added to calculate the tickets left unsold.

Table venues:

Venue name is added to create a many-to-one relationship with concerts.

Table booking agent

The contact detail of booking agents is an assumption. Including:

- a unique booking agent id also reference to booking agent id in Table artist;
- the phone number of a booking agent;
- the email address of a booking agent;
- the name of a booking agent;

Table scheduled:

The whole table scheduled is an assumption to help arrange booking agent into concerts. Includes:

- booking agent id which references to the booking agent id in Table booking agent;
- artist name, which references to the the Table artists, is added for answering query;
- reference number is the reference number of each concert which references to reference number in Table concert;
- unique venue name references to the Table concert is added;

Table artist:

Based on the problem statement, an attribute namely agent id is added into this table to clear who the booking agent is to represent the artist.

Task 2)

Table and Database Create SQL

Note: I use SQLyog as the graphical user interface. So the SQL below contains create database and use database statement.

```
SELECT DATABASE();
CREATE DATABASE b9042902;
use b9042902;

CREATE TABLE booking_agent(
agent_id VARCHARACTER(128) PRIMARY KEY,
agent_name VARCHARACTER(128),
email_address VARCHARACTER(128),
phone_number INT(32)
);

CREATE TABLE artists(
artist_id VARCHARACTER(128) PRIMARY KEY,
artist_name VARCHARACTER(128) NOT NULL,
introduction VARCHARACTER(512) NOT NULL,
agent_id VARCHARACTER(128) NOT NULL,
FOREIGN KEY (agent_id) REFERENCES booking_agent(agent_id)
ON UPDATE CASCADE -- link this table with table Booking_Agent
);

CREATE TABLE venues(
venue_name VARCHARACTER(128) UNIQUE NOT NULL,
capacity INT NOT NULL,
venue_manager_phonenumber VARCHARACTER(32) NOT NULL
);

CREATE TABLE concert(
reference_number INT AUTO_INCREMENT,
concert_date DATE NOT NULL,
start_time TIME NOT NULL,
tickets_sold INT,
venue_name VARCHARACTER(128) NOT NULL,
total_seats INT NOT NULL,
PRIMARY KEY(reference_number,venue_name)
);

CREATE TABLE customers(
cid VARCHARACTER(128) PRIMARY KEY,
cname VARCHARACTER(128) NOT NULL,
email_address VARCHARACTER(128) NOT NULL
);

CREATE TABLE scheduled(
artist_id VARCHARACTER(128) NOT NULL,
artist_name VARCHARACTER(128) NOT NULL,
reference_number INT NOT NULL,
venue_name VARCHARACTER(128) NOT NULL,
FOREIGN KEY (artist_id) REFERENCES artists(artist_id) ON UPDATE CASCADE,
FOREIGN KEY (reference_number) REFERENCES concert(reference_number)
ON UPDATE CASCADE
);

CREATE TABLE consume(
cid VARCHARACTER(128) NOT NULL,
reference_number INT NOT NULL,
ticket_bought INT NOT NULL,
FOREIGN KEY(cid) REFERENCES customers(cid) ON UPDATE CASCADE,
FOREIGN KEY(reference_number) REFERENCES concert(reference_number)
ON UPDATE CASCADE
);
```

Populate Data SQL

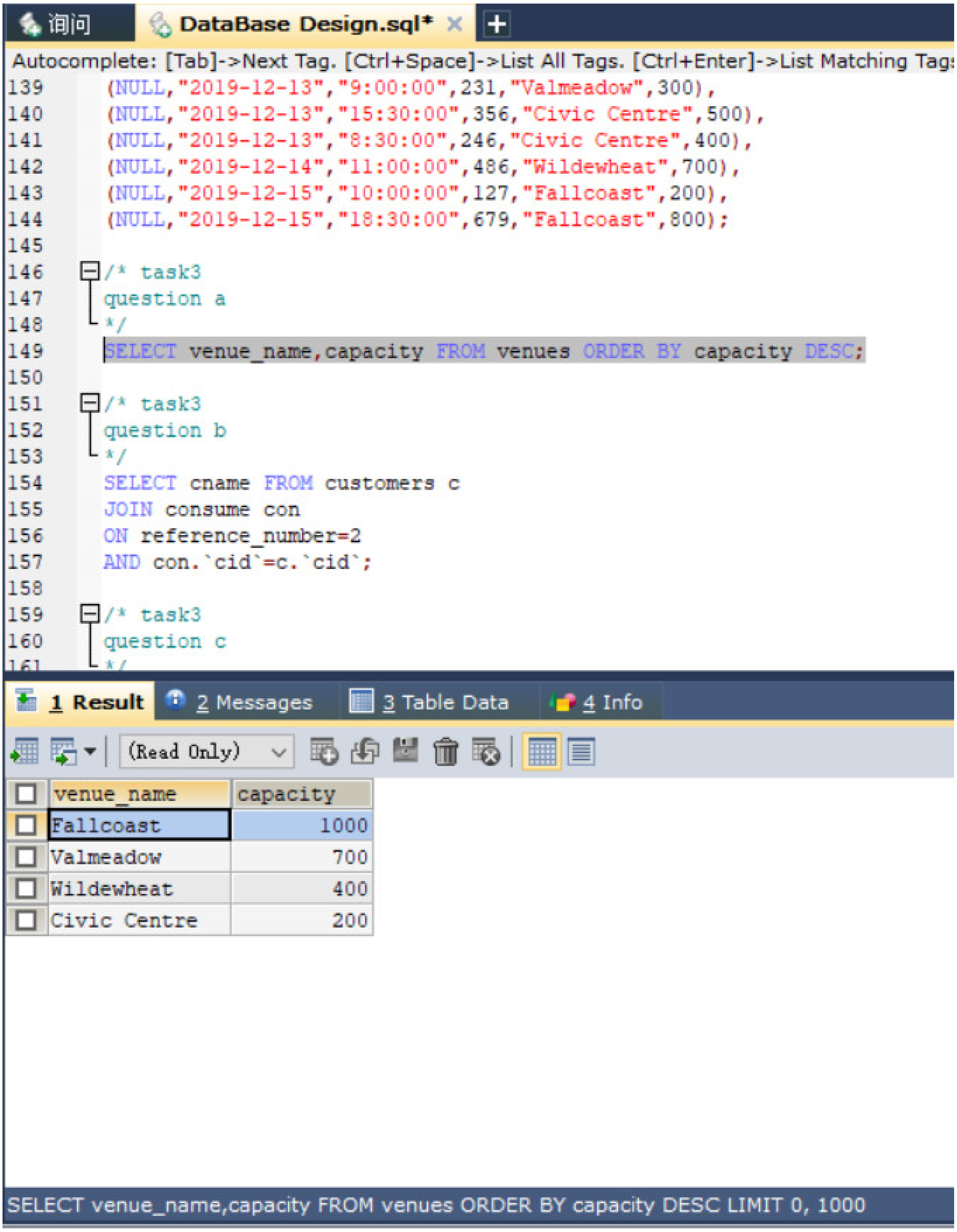
```
-- INSERT SQL
INSERT INTO booking_agent (agent_id,agent_name,email_address,phone_number)
VALUES
('B1','Gwen Skelton','B1.GS@concert.uk',01632960537),
('B2','Daisy Fox','B2.DF@concert.uk',01632960019),
('B3','Rowan Webster','B3.RW@concert.uk',01632960820);

INSERT INTO artists (artist_id,artist_name,introduction,agent_id)
VALUES
('A1','Tom Bates','an artists','B1'),
('A2','Alfred Bailey','an artists','B1'),
('A3','Dan Brent','an artists','B2'),
('A4','Elliott Stapleton','an artists','B3'),
('A5','Katy Walcott','an artists','B3'),
('A6','Arabella Baker','an artists','B3'),
('A7','Charlotte Holmes','an artists','B3');

INSERT INTO customers (cid,cname,email_address)
VALUES
('C1','Orv Kotilainen','C1.OK@customer.uk'),
('C2','Wilmette Matsumura','C2.WM@customer.uk'),
('C3','Simmonds Hayden-smith','C3.SH@customer.uk'),
('C4','Clare Keats','C4.CK@customer.uk'),
('C5','Cole Linehan','C5.CL@customer.uk'),
('C6','Stevana Shechter','C6.SS@customer.uk'),
('C7','Allyn Nagaran','C7.AN@customer.uk');
```

```
INSERT INTO venues (venue_name,capacity,venue_manager_phonenumber)
VALUES
("Civic Centre",200,"441632960418"),
("Wildewheat",400,"441632960000"),
("Valmeadow",700,"441632960248"),
("Fallcoast",1000,"441632960281");

INSERT INTO scheduled (artist_id,artist_name,reference_number,venue_name)
VALUES
("A1","Tom Bates",1,"Valmeadow"),
("A1","Tom Bates",2,"Civic Centre"),
("A1","Tom Bates",3,"Valmeadow"),
("A1","Tom Bates",5,"Civic Centre"),
("A2","Alfred Bailey",2,"Civic Centre"),
("A2","Alfred Bailey",3,"Valmeadow"),
("A2","Alfred Bailey",8,"Fallcoast"),
("A3","Dan Brent",3,"Valmeadow"),
("A3","Dan Brent",7,"Fallcoast"),
("A4","Elliott Stapleton",4,"Civic Centre"),
("A4","Elliott Stapleton",5,"Civic Centre"),
("A4","Elliott Stapleton",7,"Fallcoast"),
("A4","Elliott Stapleton",8,"Fallcoast"),
("A5","Katy Walcott",4,"Civic Centre"),
("A6","Arabella Baker",1,"Valmeadow"),
("A6","Arabella Baker",5,"Civic Centre"),
("A7","Charlotte Holmes",1,"Valmeadow");

INSERT INTO consume (cid,reference_number,ticket_bought)
VALUES
("C1",1,2),
("C2",1,3),
("C3",2,5),
("C4",2,1),
("C5",3,6),
("C6",4,1),
("C7",5,7),
("C1",2,2),
("C2",4,3),
("C3",3,5),
("C4",1,1),
("C5",1,6),
("C6",2,1),
("C7",3,7);

INSERT INTO concert
(reference_number,concert_date,start_time,tickets_sold,venue_name,total_seats)
VALUES
(NULL,"2019-12-12","9:00:00",123,"Valmeadow",200),
(NULL,"2019-12-12","15:30:00",21,"Civic Centre",100),
(NULL,"2019-12-13","9:00:00",231,"Valmeadow",300),
(NULL,"2019-12-13","15:30:00",356,"Civic Centre",500),
(NULL,"2019-12-13","8:30:00",246,"Civic Centre",400),
(NULL,"2019-12-14","11:00:00",486,"Wildewheat",700),
(NULL,"2019-12-15","10:00:00",127,"Fallcoast",200),
(NULL,"2019-12-15","18:30:00",679,"Fallcoast",800);
```

Task 3)

SQL for question a

```
/* task3    question a
*/
SELECT venue_name,capacity FROM venues ORDER BY capacity DESC;
```



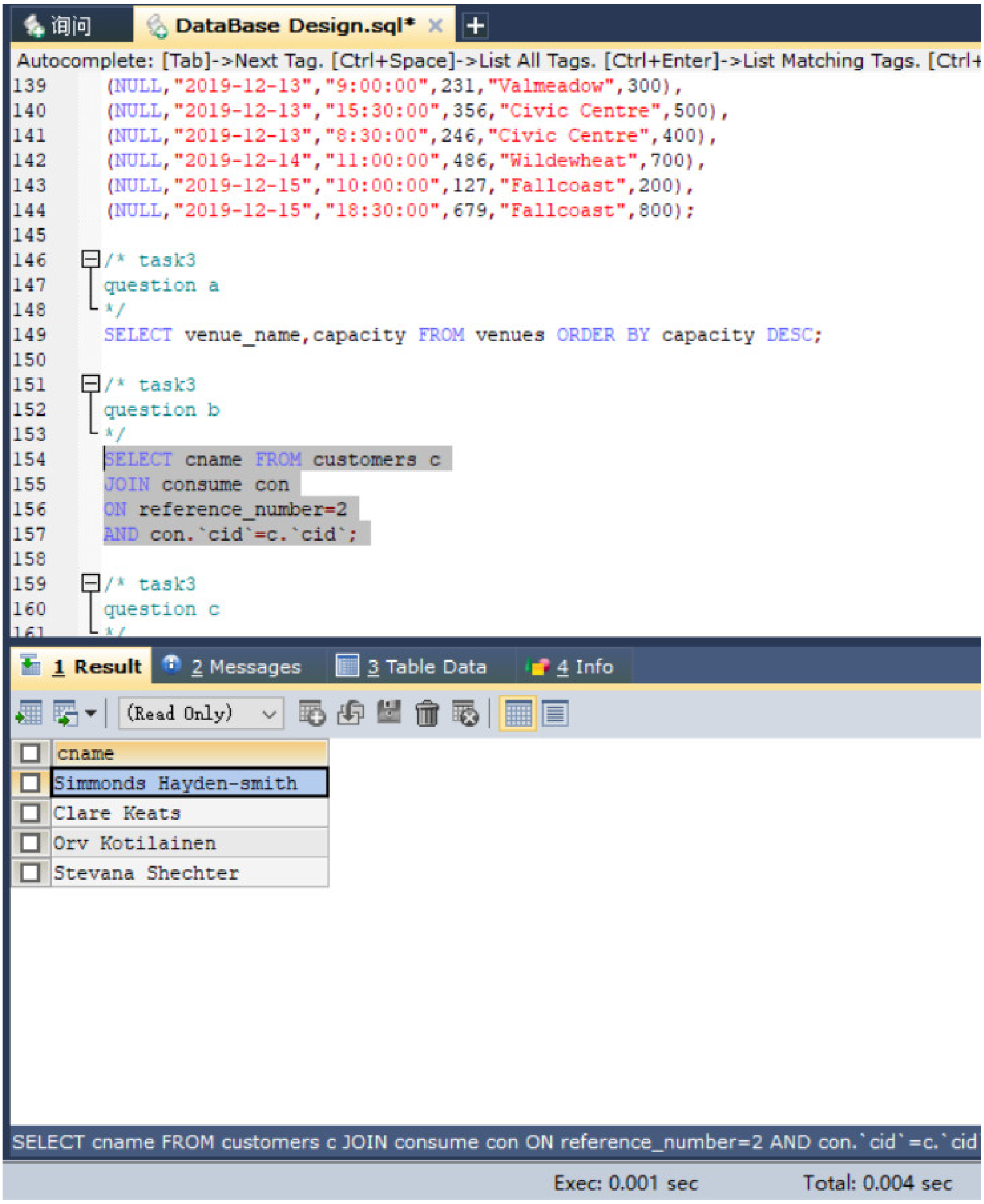
```
Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List All Tags. [Ctrl+Enter]->List Matching Tags
    (NULL,"2019-12-13","9:00:00",231,"Valmeadow",300),
    (NULL,"2019-12-13","15:30:00",356,"Civic Centre",500),
    (NULL,"2019-12-13","8:30:00",246,"Civic Centre",400),
    (NULL,"2019-12-14","11:00:00",486,"Wildewheat",700),
    (NULL,"2019-12-15","10:00:00",127,"Fallcoast",200),
    (NULL,"2019-12-15","18:30:00",679,"Fallcoast",800);

/* task3
question a
*/
SELECT venue_name,capacity FROM venues ORDER BY capacity DESC;

/* task3
question b
*/
SELECT cname FROM customers c
JOIN consume con
ON reference_number=2
AND con.`cid`=c.`cid`;

/* task3
question c
*/
```

venue_name	capacity
Fallcoast	1000
Valmeadow	700
Wildewheat	400
Civic Centre	200

```
SELECT venue_name,capacity FROM venues ORDER BY capacity DESC LIMIT 0, 1000
```

SQL for question b

```
/* task3
question b
*/
SELECT cname FROM customers c
JOIN consume con
ON reference_number=2
AND con.`cid`=c.`cid`;
```



询问
DataBase Design.sql*
Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List All Tags. [Ctrl+Enter]->List Matching Tags. [Ctrl+
1 Result
2 Messages
3 Table Data
4 Info
(Read Only)

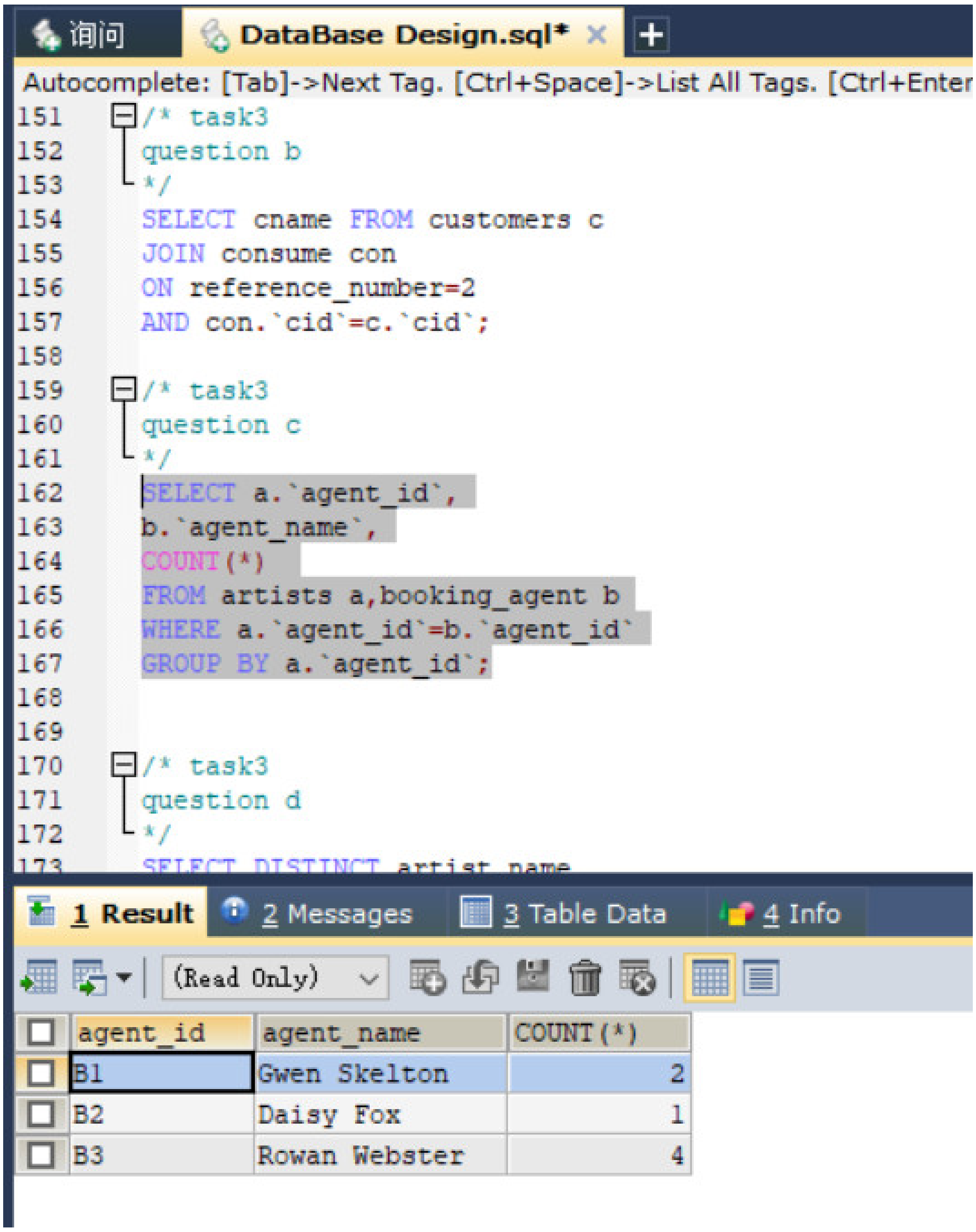
```
(NULL,"2019-12-13","9:00:00",231,"Valmeadow",300),
(NULL,"2019-12-13","15:30:00",356,"Civic Centre",500),
(NULL,"2019-12-13","8:30:00",246,"Civic Centre",400),
(NULL,"2019-12-14","11:00:00",486,"Wildewheat",700),
(NULL,"2019-12-15","10:00:00",127,"Fallcoast",200),
(NULL,"2019-12-15","18:30:00",679,"Fallcoast",800);

/* task3
question a
*/
SELECT venue_name,capacity FROM venues ORDER BY capacity DESC;

/* task3
question b
*/
SELECT cname FROM customers c
JOIN consume con
ON reference_number=2
AND con.`cid`=c.`cid`;

/* task3
question c
*/
```

cname
Simmonds Hayden-smith
Clare Keats
Orv Kotilainen
Stevana Shechter

```
SELECT cname FROM customers c JOIN consume con ON reference_number=2 AND con.`cid`=c.`cid
Exec: 0.001 sec          Total: 0.004 sec
```

SQL for question c

```
/* task3
question c
*/
SELECT a.`agent_id`,
b.`agent_name`,
COUNT(*)
FROM artists a,booking_agent b
WHERE a.`agent_id`=b.`agent_id`
GROUP BY a.`agent_id`;
```



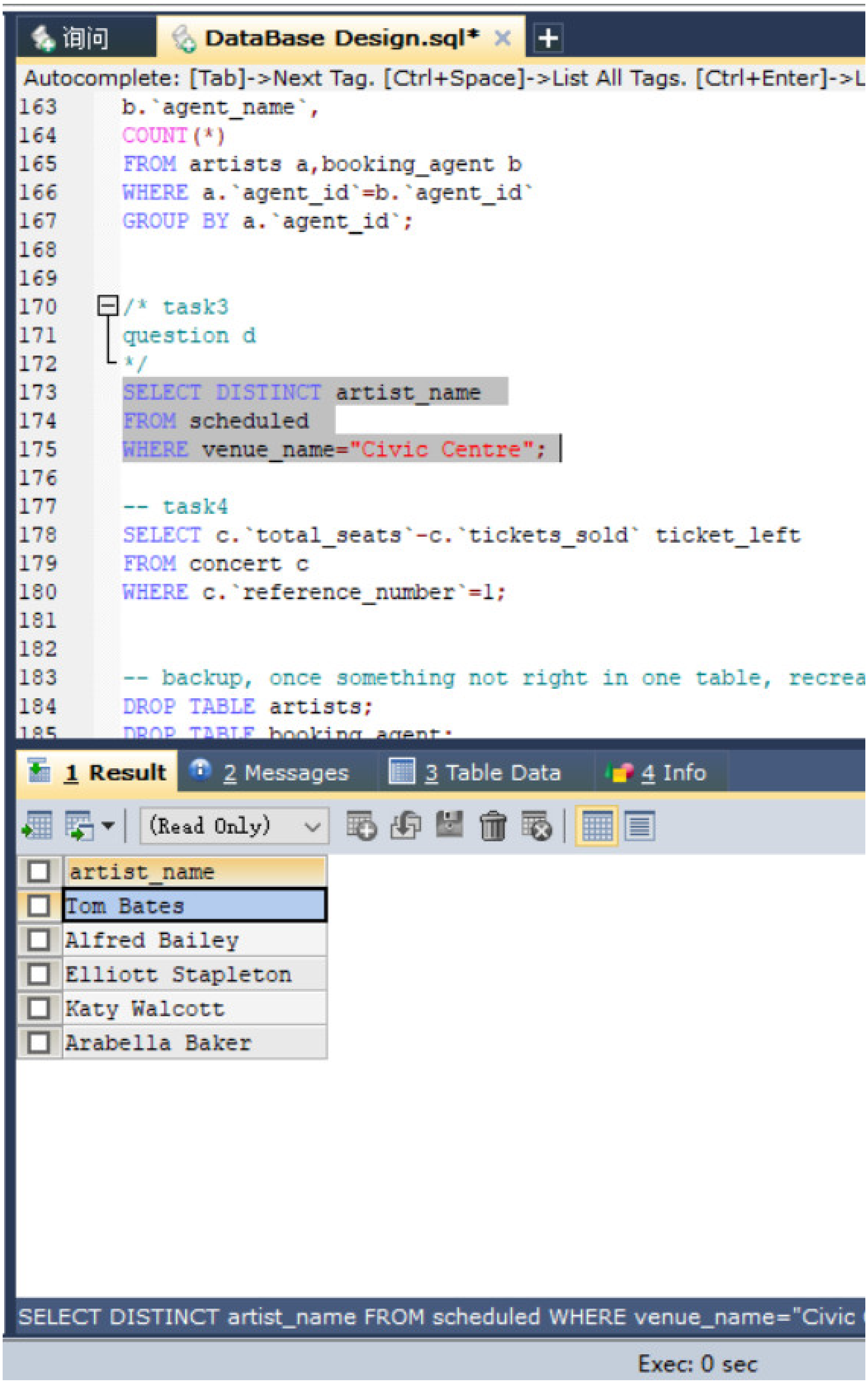
询问
DataBase Design.sql*
Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List All Tags. [Ctrl+Enter
1 Result
2 Messages
3 Table Data
4 Info
(Read Only)

```
/* task3
question b
*/
SELECT cname FROM customers c
JOIN consume con
ON reference_number=2
AND con.`cid`=c.`cid`;

/* task3
question c
*/
SELECT a.`agent_id`,
b.`agent_name`,
COUNT(*)
FROM artists a,booking_agent b
WHERE a.`agent_id`=b.`agent_id`
GROUP BY a.`agent_id`;


/* task3
question d
*/
SELECT DISTINCT artist_name
```

agent_id	agent_name	COUNT(*)
B1	Gwen Skelton	2
B2	Daisy Fox	1
B3	Rowan Webster	4

SQL for question d

```
/* task3
question d
*/
SELECT DISTINCT artist_name
FROM scheduled
WHERE venue_name="Civic Centre";
```


询问
DataBase Design.sql*
Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List All Tags. [Ctrl+Enter]->L

```
b.`agent_name`,
COUNT(*)
FROM artists a,booking_agent b
WHERE a.`agent_id`=b.`agent_id`
GROUP BY a.`agent_id`;


/* task3
question d
*/
SELECT DISTINCT artist_name
FROM scheduled
WHERE venue_name="Civic Centre";

-- task4
SELECT c.`total_seats`-c.`tickets_sold` ticket_left
FROM concert c
WHERE c.`reference_number`=1;


-- backup, once something not right in one table, recrea
DROP TABLE artists;
DROP TABLE booking_agent;
```

```
artist_name
Tom Bates
Alfred Bailey
Elliott Stapleton
Katy Walcott
Arabella Baker
```

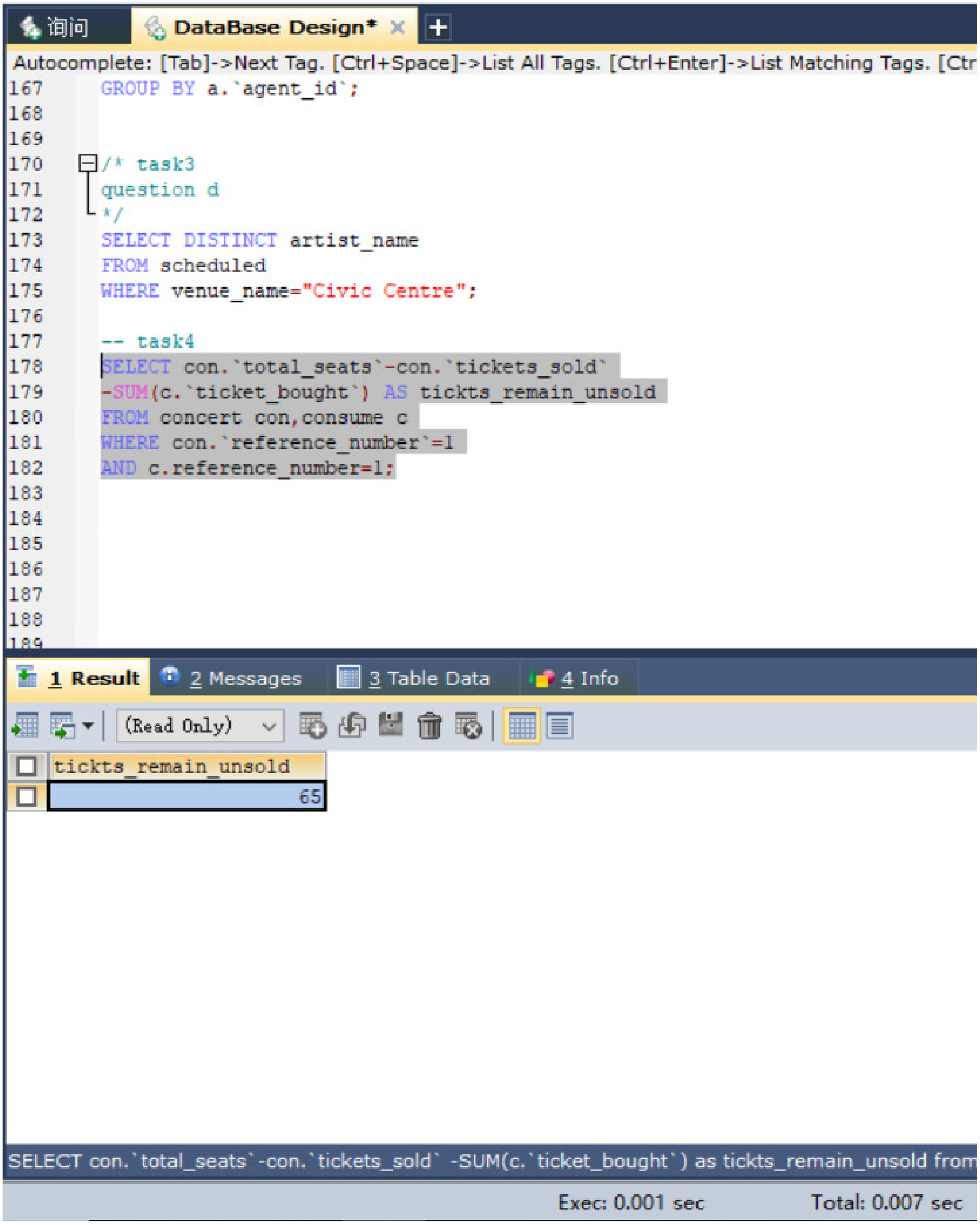
```
SELECT DISTINCT artist_name FROM scheduled WHERE venue_name="Civic
```

Task 4)

Step1. Calculate a sum number of tickets that customers purchased for concert 1;

Step2. the attribute "total tickets" subtracts the sum number and the attribute "tickets_sold", the result is tickets for concert 1 remains unsold.

```
-- task4 SQL
SELECT con.`total_seats`-SUM(c.`ticket_bought`)
FROM concert con,consume c
WHERE con.`reference_number`=1
AND c.reference_number=1;
```



询问
DataBase Design*
Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List All Tags. [Ctrl+Enter]->List Matching Tags. [Ctr

```
GROUP BY a.`agent_id`;


/* task3
question d
*/
SELECT DISTINCT artist_name
FROM scheduled
WHERE venue_name="Civic Centre";

-- task4
SELECT con.`total_seats`-con.`tickets_sold`
-SUM(c.`ticket_bought`) AS tickts_remain_unsold
FROM concert con,consume c
WHERE con.`reference_number`=1
AND c.reference_number=1;
```

1 Result
2 Messages
3 Table Data
4 Info
(Read Only)

```
tickts_remain_unsold
                  65
```

```
SELECT con.`total_seats`-con.`tickets_sold` -SUM(c.`ticket_bought`) as tickts_remain_unsold from
```

```
Exec: 0.001 sec          Total: 0.007 sec
```