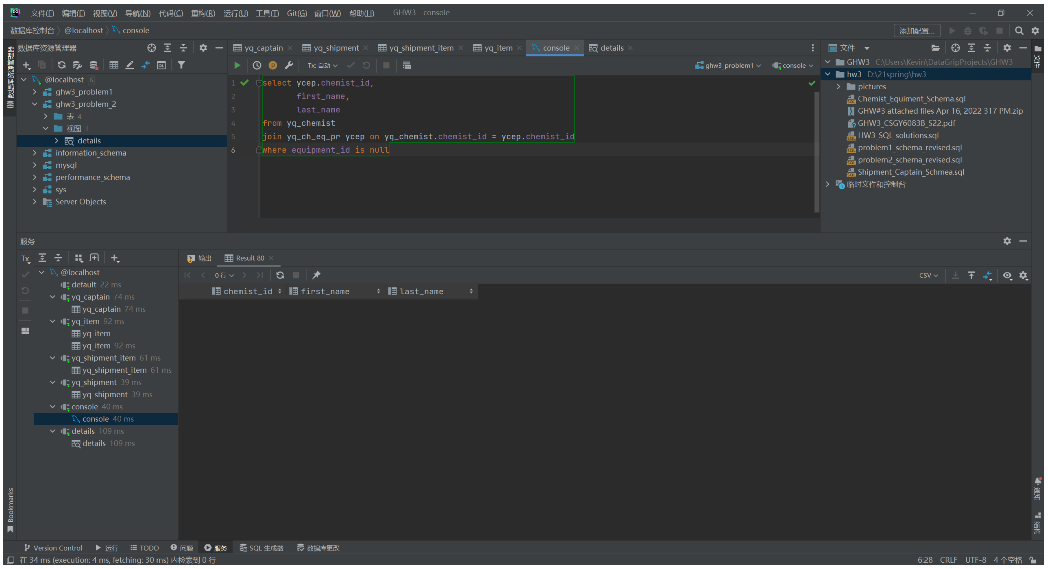
Yinhong Qin
yq2021@nyu.edu

Homework 3

2022 Spring - CS 6083
April 17, 2022

Problem 1

1. Here is the screenshot of the solution SQL and query result for problem 1-a.



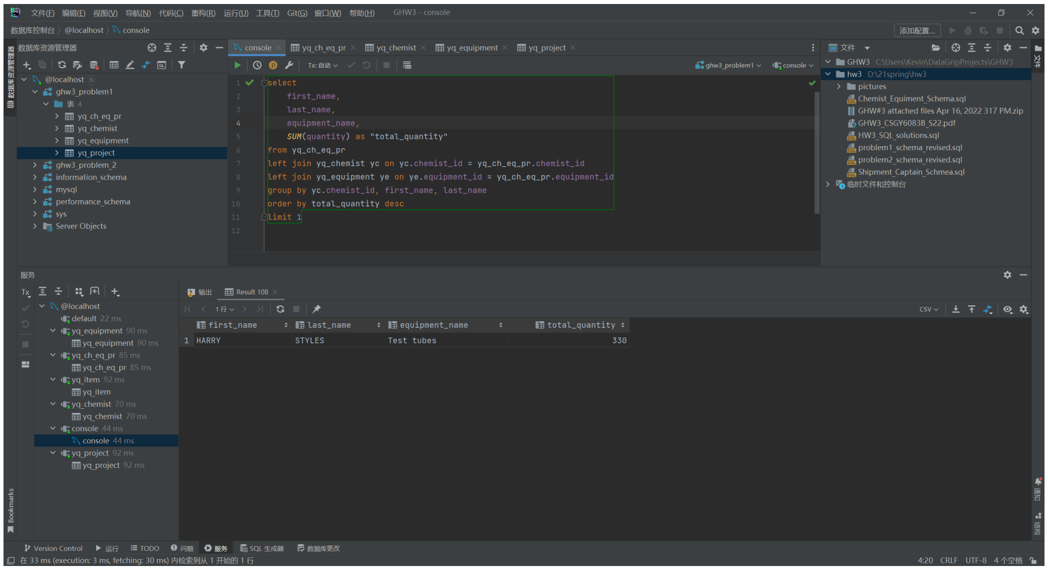
```sql
select ycep.chemist_id,
       first_name,
       last_name
from yq_chemist
join yq_ch_eq_pr ycep on yq_chemist.chemist_id = ycep.chemist_id
where equipment_id is null
```

chemist_id	first_name	last_name

```
在 34 ms (execution: 4 ms, fetching: 30 ms) 内检索到 0 行
```

Figure 1. Screenshot for problem 1-a.

2. Here is the screenshot of the solution SQL and query result for problem 1-b.



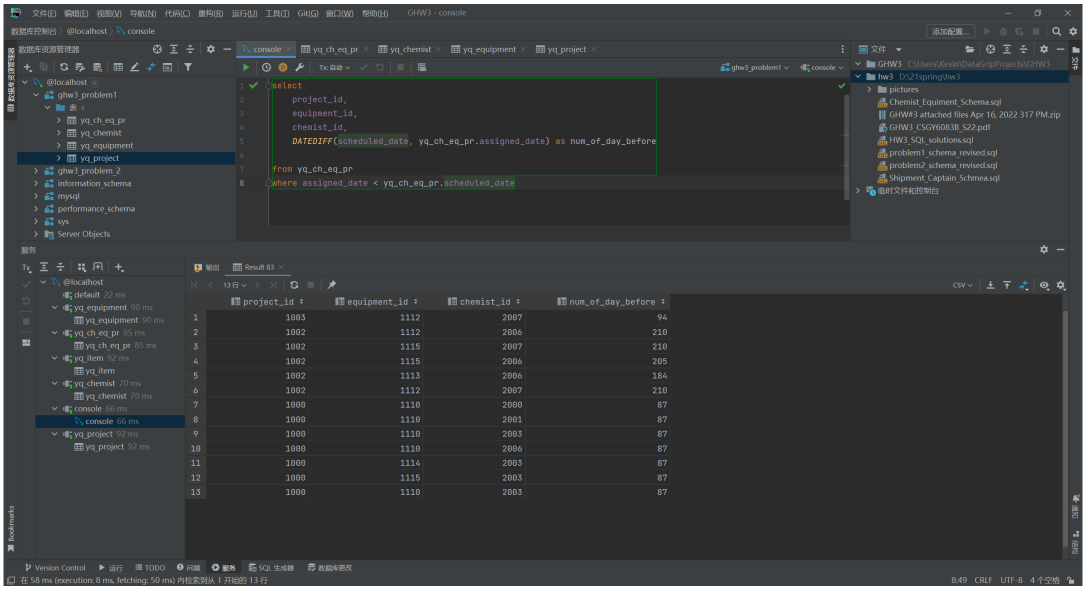
```sql
select
    first_name,
    last_name,
    equipment_name,
    SUM(quantity) as "total_quantity"
from yq_ch_eq_pr
left join yq_chemist yc on yc.chemist_id = yq_ch_eq_pr.chemist_id
left join yq_equipment ye on ye.equipment_id = yq_ch_eq_pr.equipment_id
group by yc.chemist_id, first_name, last_name
order by total_quantity desc
limit 1
```

	first_name	last_name	equipment_name	total_quantity
1	HARRY	STYLES	Test tubes	330

```
在 33 ms (execution: 3 ms, fetching: 30 ms) 内检索到从 1 开始的 1 行
```

Figure 2. Screenshot for problem 1-b.

3. Here is the screenshot of the solution SQL and query result for problem 1-c.



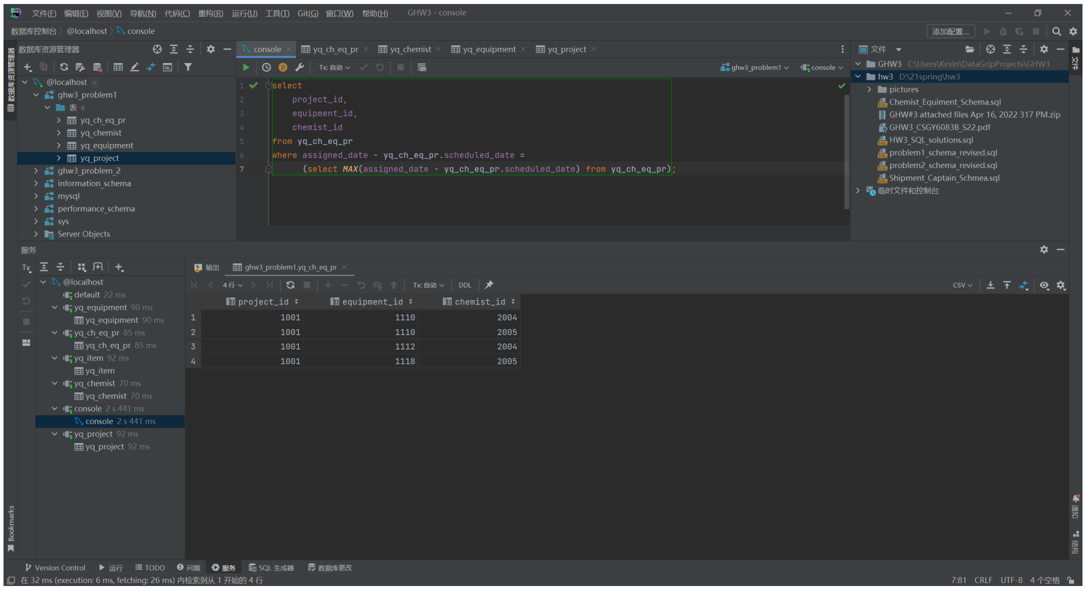
```sql
select
    project_id,
    equipment_id,
    chemist_id,
    DATEDIFF(scheduled_date, yq_ch_eq_pr.assigned_date) as num_of_day_before

from yq_ch_eq_pr
where assigned_date < yq_ch_eq_pr.scheduled_date
```

	project_id	equipment_id	chemist_id	num_of_day_before
1	1003	1112	2007	94
2	1002	1112	2006	210
3	1002	1115	2007	210
4	1002	1115	2006	205
5	1002	1113	2006	184
6	1002	1112	2007	210
7	1000	1110	2000	87
8	1000	1110	2001	87
9	1000	1110	2003	87
10	1000	1110	2006	87
11	1000	1114	2003	87
12	1000	1115	2003	87
13	1000	1110	2003	87

在 58 ms (execution: 8 ms, fetching: 50 ms) 内检索到从 1 开始的 13 行

Figure 3. Screenshot for problem 1-c.

4. Here is the screenshot of the solution SQL and query result for problem 1-d.



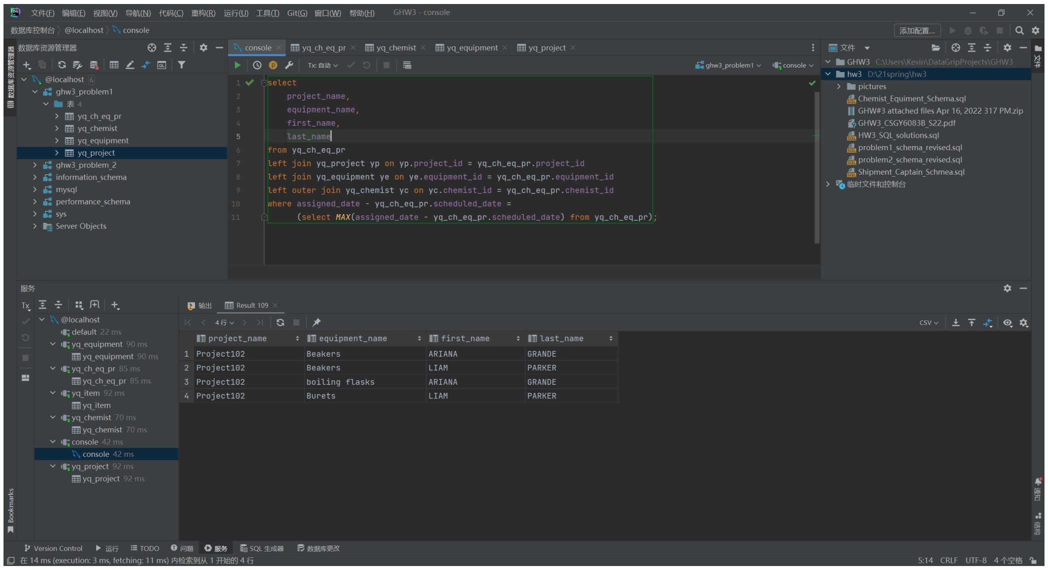
```sql
select
    project_id,
    equipment_id,
    chemist_id
from yq_ch_eq_pr
where assigned_date - yq_ch_eq_pr.scheduled_date =
      (select MAX(assigned_date - yq_ch_eq_pr.scheduled_date) from yq_ch_eq_pr);
```

	project_id	equipment_id	chemist_id
1	1001	1110	2004
2	1001	1110	2005
3	1001	1112	2004
4	1001	1118	2005

在 32 ms (execution: 6 ms, fetching: 26 ms) 内检索到从 1 开始的 4 行

Figure 4. Screenshot for problem 1-d.

5. Here is the screenshot of the solution SQL and query result for problem 1-e.



```sql
select
    project_name,
    equipment_name,
    first_name,
    last_name
from yq_ch_eq_pr
left join yq_project yp on yp.project_id = yq_ch_eq_pr.project_id
left join yq_equipment ye on ye.equipment_id = yq_ch_eq_pr.equipment_id
left outer join yq_chemist yc on yc.chemist_id = yq_ch_eq_pr.chemist_id
where assigned_date - yq_ch_eq_pr.scheduled_date =
      (select MAX(assigned_date - yq_ch_eq_pr.scheduled_date) from yq_ch_eq_pr);
```

	project_name	equipment_name	first_name	last_name
1	Project102	Beakers	ARIANA	GRANDE
2	Project102	Beakers	LIAM	PARKER
3	Project102	boiling flasks	ARIANA	GRANDE
4	Project102	Burets	LIAM	PARKER

```
在 14 ms (execution: 3 ms, fetching: 11 ms) 内检索到从 1 开始的 4 行
```

Figure 5. Screenshot for problem 1-e.

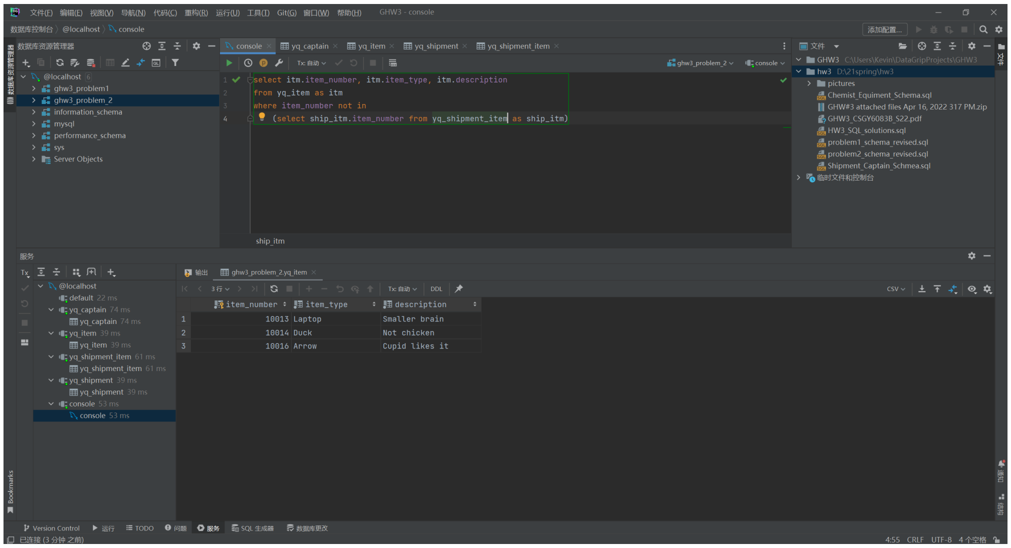
Yinhong Qin
yq2021@nyu.edu

Homework 3

2022 Spring - CS 6083
April 17, 2022

Problem 2

1. Here is the screenshot of the solution SQL and query result for problem 2-a.

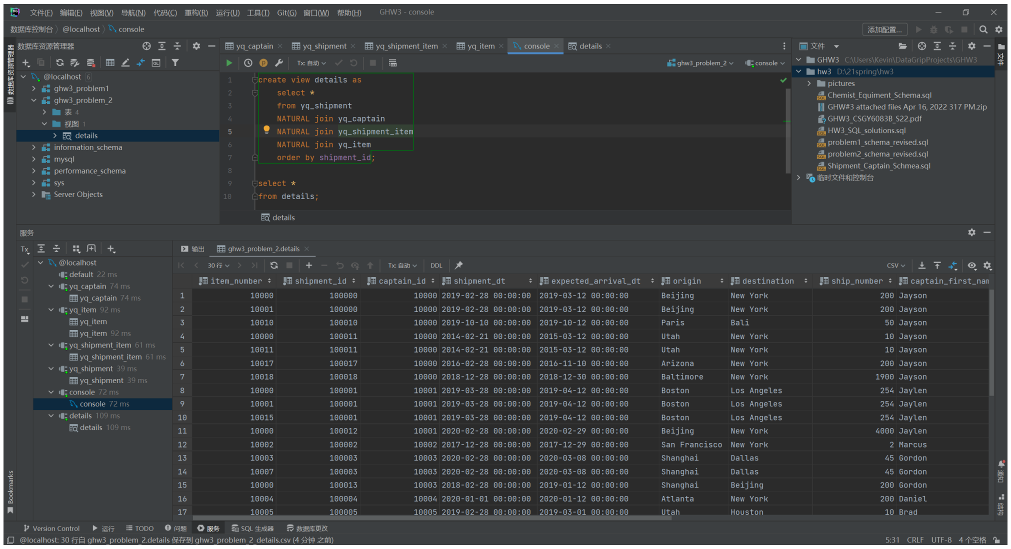


```sql
select itm.item_number, itm.item_type, itm.description
from yq_item as itm
where item_number not in
    (select ship_itm.item_number from yq_shipment_item as ship_itm)
```

	item_number	item_type	description
1	10013	Laptop	Smaller brain
2	10014	Duck	Not chicken
3	10016	Arrow	Cupid likes it

Figure 6. Screenshot for problem 2-a.

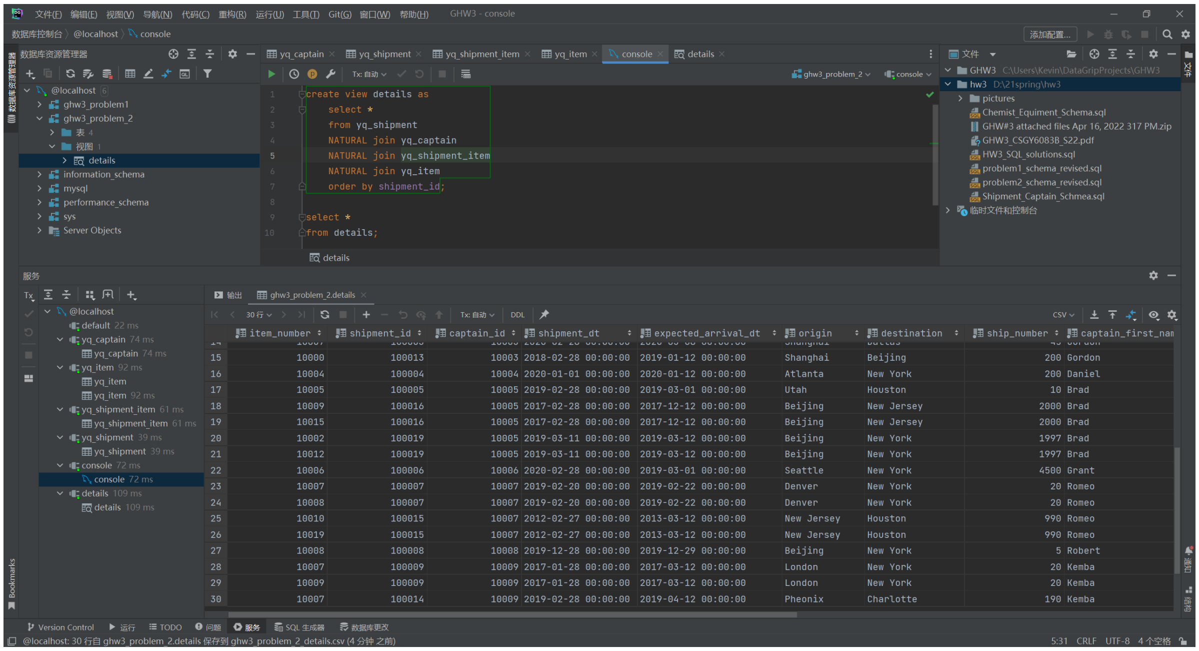
2. Here is the screenshots of the solution SQL and query result for problem 2-b. Please check the full result charts in multiple pictures.



```sql
create view details as
    select *
    from yq_shipment
    NATURAL join yq_captain
    NATURAL join yq_shipment_item
    NATURAL join yq_item
    order by shipment_id;

select *
from details;
```

	item_number	shipment_id	captain_id	shipment_dt	expected_arrival_dt	origin	destination	ship_number	captain_first_nam
1	10000	100000	10000	2019-02-28 00:00:00	2019-03-12 00:00:00	Beijing	New York	200	Jayson
2	10001	100000	10000	2019-02-28 00:00:00	2019-03-12 00:00:00	Beijing	New York	200	Jayson
3	10010	100010	10000	2019-10-10 00:00:00	2019-10-12 00:00:00	Paris	Bali	50	Jayson
4	10000	100011	10000	2014-02-21 00:00:00	2015-03-12 00:00:00	Utah	New York	10	Jayson
5	10011	100011	10000	2014-02-21 00:00:00	2015-03-12 00:00:00	Utah	New York	10	Jayson
6	10017	100017	10000	2016-02-28 00:00:00	2016-11-10 00:00:00	Arizona	New York	200	Jayson
7	10018	100018	10000	2018-12-28 00:00:00	2018-12-30 00:00:00	Baltimore	New York	1900	Jayson
8	10000	100001	10001	2019-03-28 00:00:00	2019-04-12 00:00:00	Boston	Los Angeles	254	Jaylen
9	10001	100001	10001	2019-03-28 00:00:00	2019-04-12 00:00:00	Boston	Los Angeles	254	Jaylen
10	10015	100001	10001	2019-03-28 00:00:00	2019-04-12 00:00:00	Boston	Los Angeles	254	Jaylen
11	10000	100012	10001	2020-02-28 00:00:00	2020-02-29 00:00:00	Beijing	New York	4000	Jaylen
12	10002	100002	10002	2017-12-28 00:00:00	2017-12-29 00:00:00	San Francisco	New York	2	Marcus
13	10003	100003	10003	2020-02-28 00:00:00	2020-03-08 00:00:00	Shanghai	Dallas	45	Gordon
14	10007	100003	10003	2020-02-28 00:00:00	2020-03-08 00:00:00	Shanghai	Dallas	45	Gordon
15	10000	100013	10003	2018-02-28 00:00:00	2019-01-12 00:00:00	Shanghai	Beijing	200	Gordon
16	10004	100004	10004	2020-01-01 00:00:00	2020-01-12 00:00:00	Atlanta	New York	200	Daniel
17	10005	100005	10005	2019-02-28 00:00:00	2019-03-01 00:00:00	Utah	Houston	10	Brad

Figure 7. Screenshot for problem 2-b.

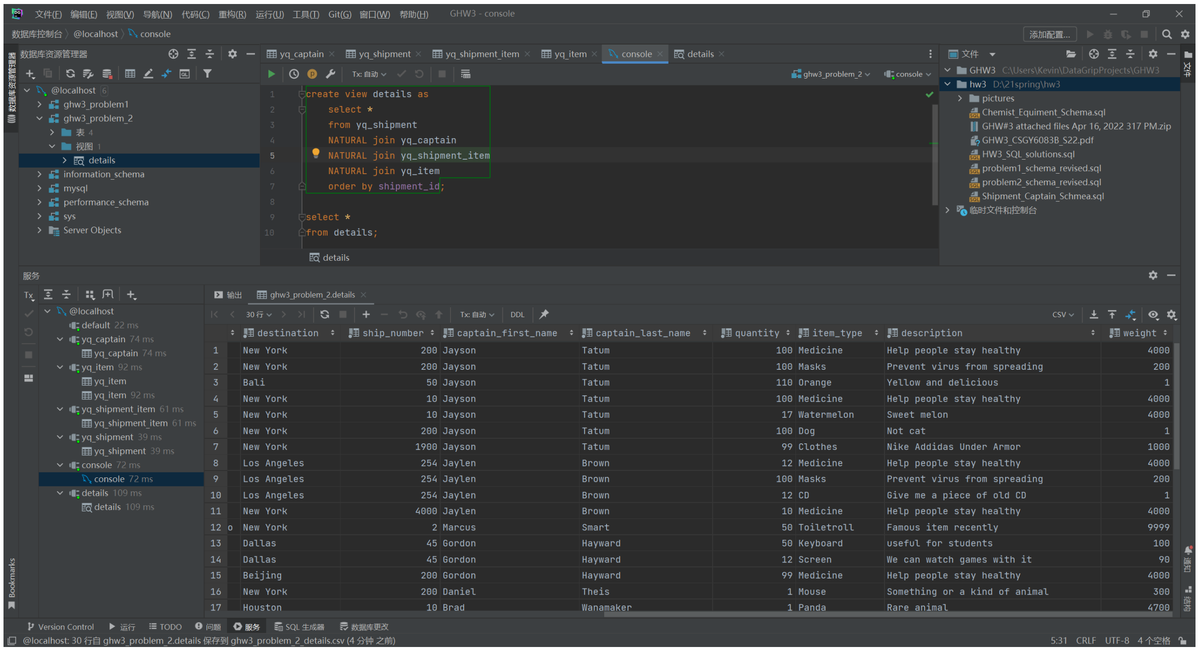


```
create view details as
    select *
    from yq_shipment
    NATURAL join yq_captain
    NATURAL join yq_shipment_item
    NATURAL join yq_item
    order by shipment_id;

select *
from details;
```

	item_number	shipment_id	captain_id	shipment_dt	expected_arrival_dt	origin	destination	ship_number	captain_first_nam
15	10000	100013	10003	2018-02-28 00:00:00	2019-01-12 00:00:00	Shanghai	Beijing	200	Gordon
16	10004	100004	10004	2020-01-01 00:00:00	2020-01-12 00:00:00	Atlanta	New York	200	Daniel
17	10005	100005	10005	2019-02-28 00:00:00	2019-03-01 00:00:00	Utah	Houston	10	Brad
18	10009	100016	10005	2017-02-28 00:00:00	2017-12-12 00:00:00	Beijing	New Jersey	2000	Brad
19	10015	100016	10005	2017-02-28 00:00:00	2017-12-12 00:00:00	Beijing	New Jersey	2000	Brad
20	10002	100019	10005	2019-03-11 00:00:00	2019-03-12 00:00:00	Beijing	New York	1997	Brad
21	10012	100019	10005	2019-03-11 00:00:00	2019-03-12 00:00:00	Beijing	New York	1997	Brad
22	10006	100006	10006	2020-02-28 00:00:00	2019-03-01 00:00:00	Seattle	New York	4500	Grant
23	10007	100007	10007	2019-02-20 00:00:00	2019-02-22 00:00:00	Denver	New York	20	Romeo
24	10008	100007	10007	2019-02-20 00:00:00	2019-02-22 00:00:00	Denver	New York	20	Romeo
25	10010	100015	10007	2012-02-27 00:00:00	2013-03-12 00:00:00	New Jersey	Houston	990	Romeo
26	10019	100015	10007	2012-02-27 00:00:00	2013-03-12 00:00:00	New Jersey	Houston	990	Romeo
27	10008	100008	10008	2019-12-28 00:00:00	2019-12-29 00:00:00	Beijing	New York	5	Robert
28	10007	100009	10009	2017-01-28 00:00:00	2017-03-12 00:00:00	London	New York	20	Kemba
29	10009	100009	10009	2017-01-28 00:00:00	2017-03-12 00:00:00	London	New York	20	Kemba
30	10007	100014	10009	2019-02-28 00:00:00	2019-04-12 00:00:00	Pheonix	Charlotte	190	Kemba

@localhost: 30 行自 ghw3_problem_2.details 保存到 ghw3_problem_2_details.csv (4 分钟 之前)

Figure 8. Screenshot for problem 2-b.

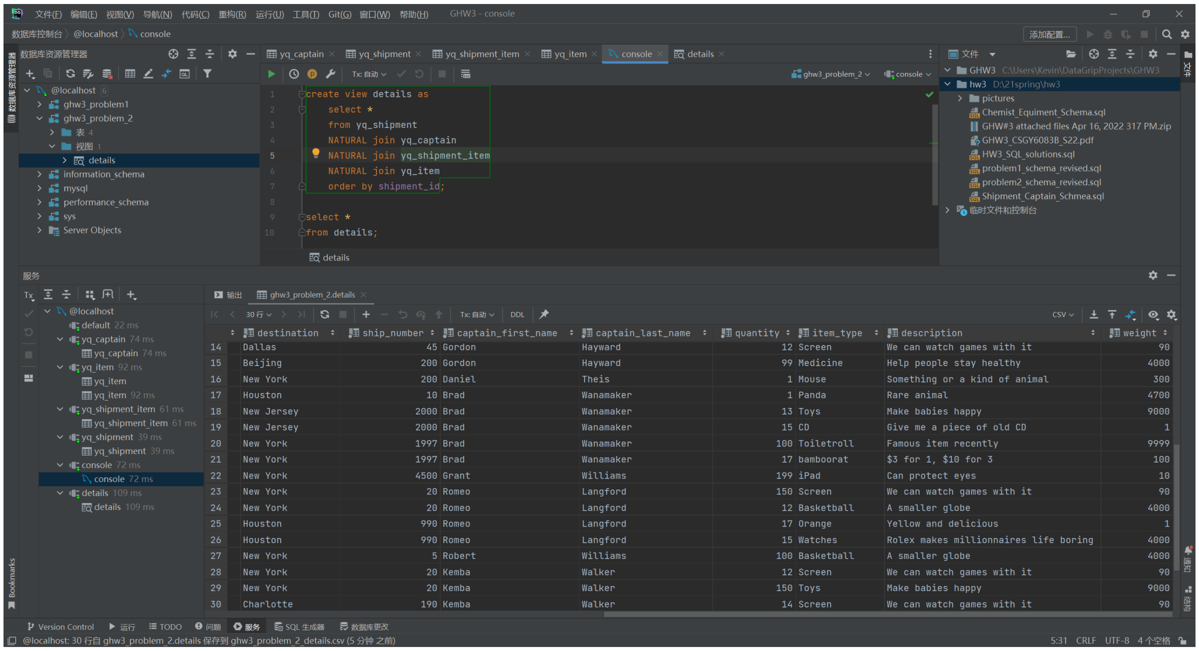


```
create view details as
    select *
    from yq_shipment
    NATURAL join yq_captain
    NATURAL join yq_shipment_item
    NATURAL join yq_item
    order by shipment_id;

select *
from details;
```

		destination	ship_number	captain_first_name	captain_last_name	quantity	item_type	description	weight
1		New York	200	Jayson	Tatum	100	Medicine	Help people stay healthy	4000
2		New York	200	Jayson	Tatum	100	Masks	Prevent virus from spreading	200
3		Bali	50	Jayson	Tatum	110	Orange	Yellow and delicious	1
4		New York	10	Jayson	Tatum	100	Medicine	Help people stay healthy	4000
5		New York	10	Jayson	Tatum	17	Watermelon	Sweet melon	4000
6		New York	200	Jayson	Tatum	100	Dog	Not cat	1
7		New York	1900	Jayson	Tatum	99	Clothes	Nike Addidas Under Armor	1000
8		Los Angeles	254	Jaylen	Brown	12	Medicine	Help people stay healthy	4000
9		Los Angeles	254	Jaylen	Brown	100	Masks	Prevent virus from spreading	200
10		Los Angeles	254	Jaylen	Brown	12	CD	Give me a piece of old CD	1
11		New York	4000	Jaylen	Brown	10	Medicine	Help people stay healthy	4000
12	o	New York	2	Marcus	Smart	50	Toiletroll	Famous item recently	9999
13		Dallas	45	Gordon	Hayward	50	Keyboard	useful for students	100
14		Dallas	45	Gordon	Hayward	12	Screen	We can watch games with it	90
15		Beijing	200	Gordon	Hayward	99	Medicine	Help people stay healthy	4000
16		New York	200	Daniel	Theis	1	Mouse	Something or a kind of animal	300
17		Houston	10	Brad	Wanamaker	1	Panda	Rare animal	4700

@localhost: 30 行自 ghw3_problem_2.details 保存到 ghw3_problem_2_details.csv (4 分钟 之前)

Figure 9. Screenshot for problem 2-b.



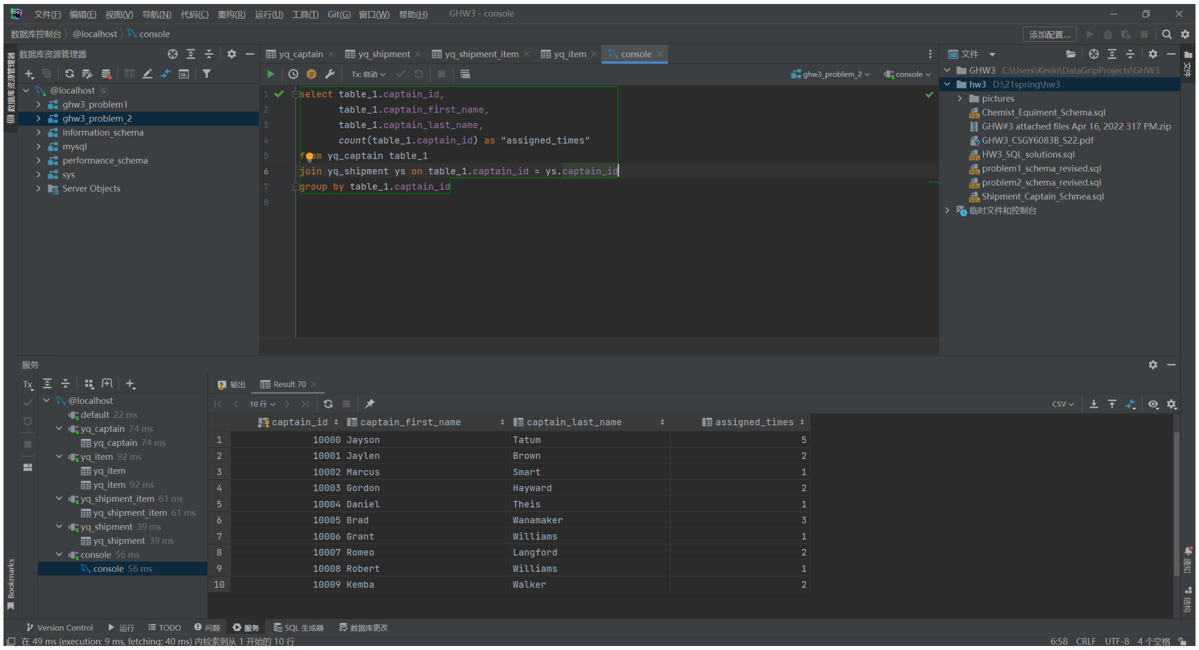
```sql
create view details as
    select *
    from yq_shipment
    NATURAL join yq_captain
    NATURAL join yq_shipment_item
    NATURAL join yq_item
    order by shipment_id;

select *
from details;
```

	destination	ship_number	captain_first_name	captain_last_name	quantity	item_type	description	weight
14	Dallas	45	Gordon	Hayward	12	Screen	We can watch games with it	90
15	Beijing	200	Gordon	Hayward	99	Medicine	Help people stay healthy	4000
16	New York	200	Daniel	Theis	1	Mouse	Something or a kind of animal	300
17	Houston	10	Brad	Wanamaker	1	Panda	Rare animal	4700
18	New Jersey	2000	Brad	Wanamaker	13	Toys	Make babies happy	9000
19	New Jersey	2000	Brad	Wanamaker	15	CD	Give me a piece of old CD	1
20	New York	1997	Brad	Wanamaker	100	Toiletroll	Famous item recently	9999
21	New York	1997	Brad	Wanamaker	17	bamboorat	$3 for 1, $10 for 3	100
22	New York	4500	Grant	Williams	199	iPad	Can protect eyes	10
23	New York	20	Romeo	Langford	150	Screen	We can watch games with it	90
24	New York	20	Romeo	Langford	12	Basketball	A smaller globe	4000
25	Houston	990	Romeo	Langford	17	Orange	Yellow and delicious	1
26	Houston	990	Romeo	Langford	15	Watches	Rolex makes millionnaires life boring	4000
27	New York	5	Robert	Williams	100	Basketball	A smaller globe	4000
28	New York	20	Kemba	Walker	12	Screen	We can watch games with it	90
29	New York	20	Kemba	Walker	150	Toys	Make babies happy	9000
30	Charlotte	190	Kemba	Walker	14	Screen	We can watch games with it	90

@localhost: 30 行自 ghw3_problem_2.details 保存到 ghw3_problem_2_details.csv (5 分钟 之前)

Figure 10. Screenshot for problem 2-b.

3. Here is the screenshot of the solution SQL and query result for problem 2-c.



```sql
select table_1.captain_id,
       table_1.captain_first_name,
       table_1.captain_last_name,
       count(table_1.captain_id) as "assigned_times"
from yq_captain table_1
join yq_shipment ys on table_1.captain_id = ys.captain_id
group by table_1.captain_id
```

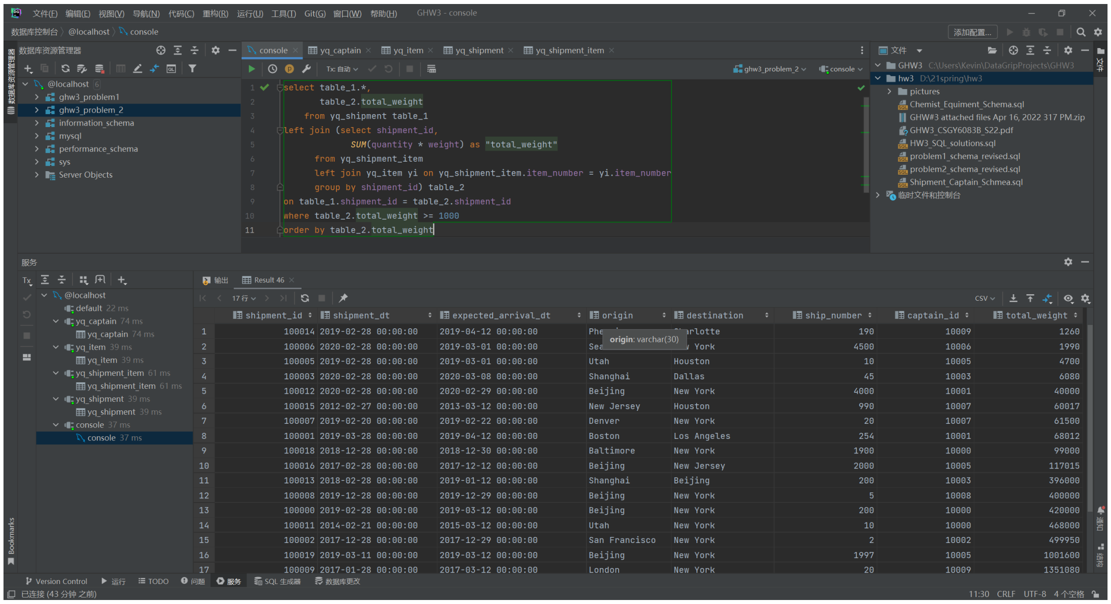
	captain_id	captain_first_name	captain_last_name	assigned_times
1	10000	Jayson	Tatum	5
2	10001	Jaylen	Brown	2
3	10002	Marcus	Smart	1
4	10003	Gordon	Hayward	2
5	10004	Daniel	Theis	1
6	10005	Brad	Wanamaker	3
7	10006	Grant	Williams	1
8	10007	Romeo	Langford	2
9	10008	Robert	Williams	1
10	10009	Kemba	Walker	2

在 49 ms (execution: 9 ms, fetching: 40 ms) 内检索到从 1 开始的 10 行

Figure 11. Screenshot for problem 2-c.

Homework 3

4. Here is the screenshot of the solution SQL and query result for problem 2-d.

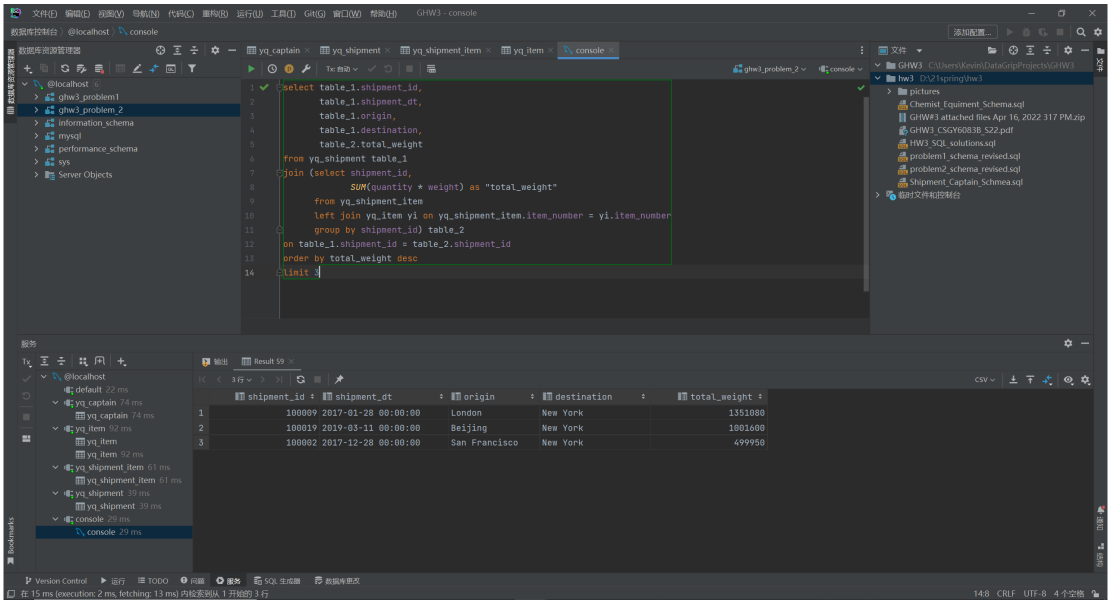


```sql
select table_1.*,
       table_2.total_weight
    from yq_shipment table_1
left join (select shipment_id,
                  SUM(quantity * weight) as "total_weight"
      from yq_shipment_item
      left join yq_item yi on yq_shipment_item.item_number = yi.item_number
      group by shipment_id) table_2
on table_1.shipment_id = table_2.shipment_id
where table_2.total_weight >= 1000
order by table_2.total_weight
```

	shipment_id	shipment_dt	expected_arrival_dt	origin	destination	ship_number	captain_id	total_weight
1	100014	2019-02-28 00:00:00	2019-04-12 00:00:00	Phe	arlotte	190	10009	1260
2	100006	2020-02-28 00:00:00	2019-03-01 00:00:00	Sea	York	4500	10006	1990
3	100005	2019-02-28 00:00:00	2019-03-01 00:00:00	Utah	Houston	10	10005	4700
4	100003	2020-02-28 00:00:00	2020-03-08 00:00:00	Shanghai	Dallas	45	10003	6080
5	100012	2020-02-28 00:00:00	2020-02-29 00:00:00	Beijing	New York	4000	10001	40000
6	100015	2012-02-27 00:00:00	2013-03-12 00:00:00	New Jersey	Houston	990	10007	60017
7	100007	2019-02-20 00:00:00	2019-02-22 00:00:00	Denver	New York	20	10007	61500
8	100001	2019-03-28 00:00:00	2019-04-12 00:00:00	Boston	Los Angeles	254	10001	68012
9	100018	2018-12-28 00:00:00	2018-12-30 00:00:00	Baltimore	New York	1900	10000	99000
10	100016	2017-02-28 00:00:00	2017-12-12 00:00:00	Beijing	New Jersey	2000	10005	117015
11	100013	2018-02-28 00:00:00	2019-01-12 00:00:00	Shanghai	Beijing	200	10003	396000
12	100008	2019-12-28 00:00:00	2019-12-29 00:00:00	Beijing	New York	5	10008	400000
13	100000	2019-02-28 00:00:00	2019-03-12 00:00:00	Beijing	New York	200	10000	420000
14	100011	2014-02-21 00:00:00	2015-03-12 00:00:00	Utah	New York	10	10000	468000
15	100002	2017-12-28 00:00:00	2017-12-29 00:00:00	San Francisco	New York	2	10002	499950
16	100019	2019-03-11 00:00:00	2019-03-12 00:00:00	Beijing	New York	1997	10005	1001600
17	100009	2017-01-28 00:00:00	2017-03-12 00:00:00	London	New York	20	10009	1351080

Figure 12. Screenshot for problem 2-d.

5. Here is the screenshot of the solution SQL and query result for problem 2-e.



```sql
select table_1.shipment_id,
       table_1.shipment_dt,
       table_1.origin,
       table_1.destination,
       table_2.total_weight
from yq_shipment table_1
join (select shipment_id,
             SUM(quantity * weight) as "total_weight"
      from yq_shipment_item
      left join yq_item yi on yq_shipment_item.item_number = yi.item_number
      group by shipment_id) table_2
on table_1.shipment_id = table_2.shipment_id
order by total_weight desc
limit 3
```

	shipment_id	shipment_dt	origin	destination	total_weight
1	100009	2017-01-28 00:00:00	London	New York	1351080
2	100019	2019-03-11 00:00:00	Beijing	New York	1001600
3	100002	2017-12-28 00:00:00	San Francisco	New York	499950

Figure 13. Screenshot for problem 2-e.