

# 关卡 3：数据仓库建模及数据分析



华为技术有限公司

## 课堂思考题

---

### 课堂思考题（20 分）

1. 对刚刚提到的三种架构做一个总结并谈谈对其应用场景的理解。（10 分）

答：

辐射状企业信息工厂 Inmon 架构：自上而下，建立规范化表以便后续管理。初期需要较多精力建立规范化数据仓库，后期管理成本较低。

Kimball 的 DW/BI 架构：自下而上，调用链路较少，数据具有较好完整性与一致性，使用一致性维度，更加稳定、高效。能较快投入使用，但由于没有强规范型要求，后期容易出现冗余字段、不稳定数据等。

独立数据集市架构：数据的局部化处理，在跨部门处理数据时具有较高效率，但造成大量数据冗余。适合较低成本的快速开发，但从全局看不具有部门之间的兼容性

2. 粒度举例。（5 分）

答：

以网聊场景为例，事实表的原子粒度可定义为：用户的某个聊天窗口中的每条消息，在此基础上可以定义高粒度的聚集事实表，例如一天内用户在某个聊天窗口中的所有消息

3. 维度举例。（5 分）

答：

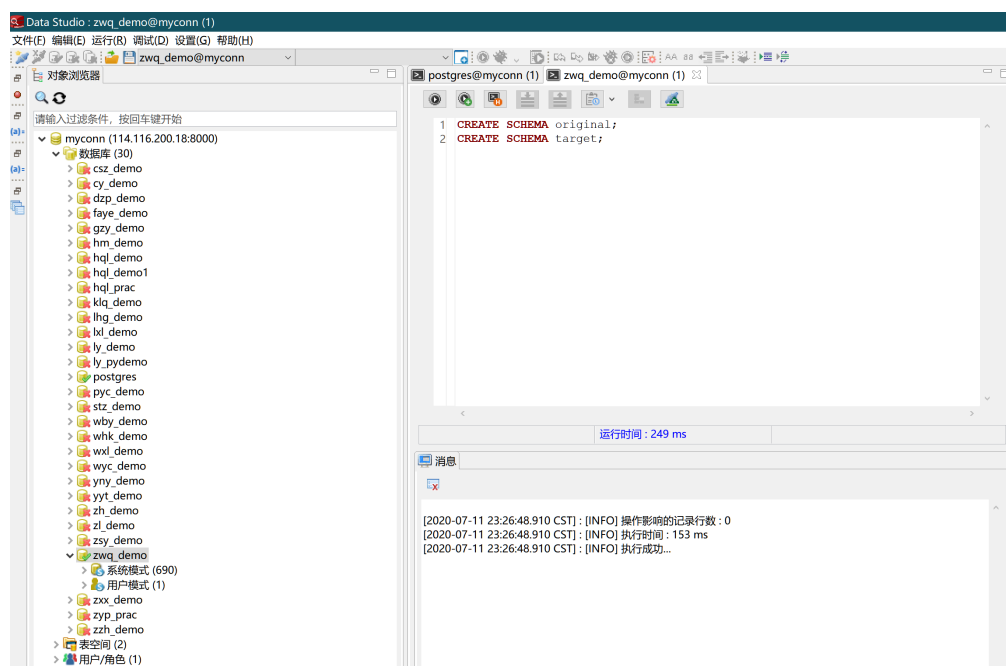
同样以网聊场景为例，维度包括：时间维度、用户维度、IP 地址维度、设备维度等

# 实验

## 数据库对象创建（15 分）

1. 截图内容：创建的个人数据库及两个模式。（5 分）

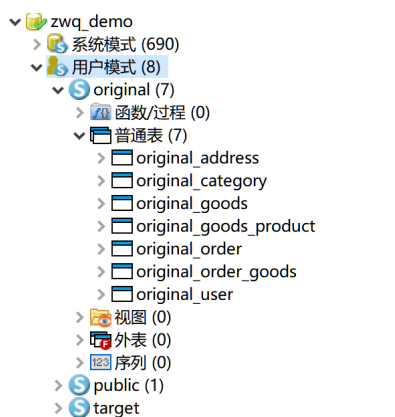
请在下方附上图片



请在上方附上图片

2. 截图内容：original 模式下所创建的表。（5 分）

请在下方附上图片



请在上方附上图片

3. 截图内容：target 模式下所创建的表和序列。（5 分）

请在下方附上图片

请在上方附上图片

## 数据转换（25 分）

1. 截图内容：正确连接到数据库 rdsformysql 的测试提示信息以及正确连接到数据库 dws 的测试提示信息。（5 分）

请在下方附上图片

请在上方附上图片

2. 在 original 模式下一共创建了 7 张表，需要利用 kettle 实现 7 个转换，完成数据从 RDS for MySQL 到 DWS 的导入工作。截图内容：每个转换的“步骤度量”信息。（2 分/个，共 14 分）

goods 转换

请在下方附上图片

执行结果

#	步骤名称	复制的记录行数	读	写	输入	输出	更新	拒绝	错误	激活	时间	速度 (条记录/秒)	Pri/in/out
1	litemall_goods	0	0	31	31	0	0	0	0	已完成	0.2s	188	-
2	original_goods	0	31	31	0	31	0	0	0	已完成	0.4s	81	-

请在上方附上图片

## goods\_product 转换 (3 分)

-----请在下方附上图片-----

## 执行结果

执行历史 日志 步骤度量 性能图 Metrics Preview data													
#	步骤名称	复制的记录行数	读	写	输入	输出	更新	拒绝	错误	激活	时间	速度 (条记录/秒)	Pri/in/out
1	litemall_goods_product	0	0	31	31	0	0	0	0	已完成	0.1s	274	-
2	original_goods_product	0	31	31	0	31	0	0	0	已完成	0.2s	133	-

-----请在上方附上图片-----

## order 转换 (3 分)

-----请在下方附上图片-----

## 执行结果

执行历史 日志 步骤度量 性能图 Metrics Preview data													
#	步骤名称	复制的记录行数	读	写	输入	输出	更新	拒绝	错误	激活	时间	速度 (条记录/秒)	Pri/in/out
1	litemall_order	0	0	60531	60531	0	0	0	0	已完成	51.3s	1,180	-
2	original_order	0	60531	60531	0	60531	0	0	0	已完成	1mn 3s	959	-

-----请在上方附上图片-----

## order\_goods 转换 (3 分)

-----请在下方附上图片-----

## 执行结果

执行历史 日志 步骤度量 性能图 Metrics Preview data													
#	步骤名称	复制的记录行数	读	写	输入	输出	更新	拒绝	错误	激活	时间	速度 (条记录/秒)	Pri/in/out
1	litemall_order_goods	0	0	63701	63701	0	0	0	0	已完成	49.9s	1,277	-
2	original_order_goods	0	63701	63701	0	63701	0	0	0	已完成	59.3s	1,075	-

-----请在上方附上图片-----

## address 转换

-----请在下方附上图片-----

## 执行结果

执行历史 日志 步骤度量 性能图 Metrics Preview data													
#	步骤名称	复制的记录行数	读	写	输入	输出	更新	拒绝	错误	激活	时间	速度 (条记录/秒)	Pri/in/out
1	litemall_address	0	0	15001	15001	0	0	0	0	已完成	5.5s	2,750	-
2	original_address	0	15001	15001	0	15001	0	0	0	已完成	14.2s	1,058	-

-----请在上方附上图片-----

## user 转换

-----请在下方附上图片-----

## 执行结果

执行历史 日志 步骤度量 性能图 Metrics Preview data													
#	步骤名称	复制的记录行数	读	写	输入	输出	更新	拒绝	错误	激活	时间	速度 (条记录/秒)	Pri/in/out
1	litemall_user	0	0	15001	15001	0	0	0	0	已完成	5.5s	2,736	-
2	original_user	0	15001	15001	0	15001	0	0	0	已完成	13.8s	1,084	-

-----请在上方附上图片-----

## category 转换

-----请在下方附上图片-----

## 执行结果

执行历史

日志

步骤度量

性能图

Metrics

Preview data

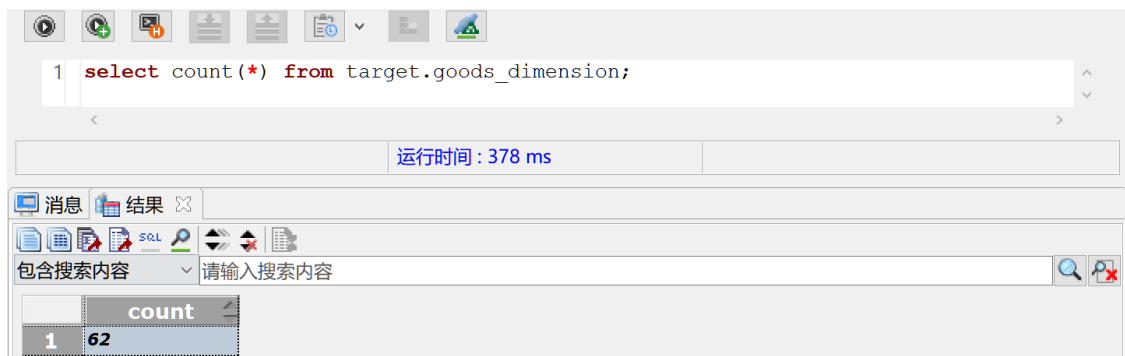
#	步骤名称	复制的记录行数	读	写	输入	输出	更新	拒绝	错误	激活	时间	速度 (条记录/秒)	Pri/in/out
1	litmall_category	0	0	21	21	0	0	0	0	已完成	0.1s	189	-
2	original_category	0	21	21	0	21	0	0	0	已完成	0.2s	84	-

-----请在上方附上图片-----

3. 在 target 模式下创建了 8 张表，需要利用 SQL 语句将 original 模式下的源数据加载到 target 模式下的事实表和维度表中。截图内容：select count(\*) from tablename;语句分别查询 target 模式下除了 date\_dimention 和 time\_dimension 外其余 6 张表的数据量。（1 分/个，共 6 分）

## goods\_dimension 表

-----请在下方附上图片-----



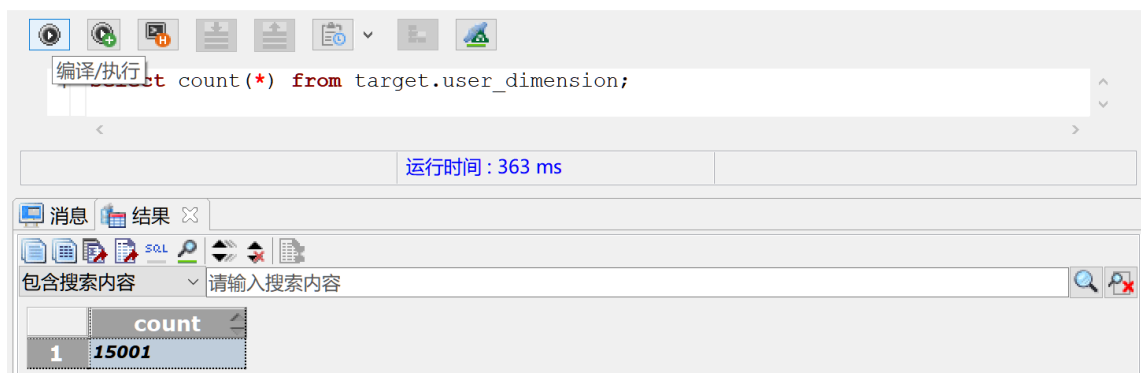
The screenshot shows a SQL query execution interface. The query is: `select count(*) from target.goods_dimension;`. The execution time is 378 ms. The results pane shows a single row with the value 62.

count
62

-----请在上方附上图片-----

## user\_dimension 表

-----请在下方附上图片-----



The screenshot shows a SQL query execution interface. The query is: `select count(*) from target.user_dimension;`. The execution time is 363 ms. The results pane shows a single row with the value 15001.

count
15001

-----请在上方附上图片-----

address\_dimension 表

请在下方附上图片



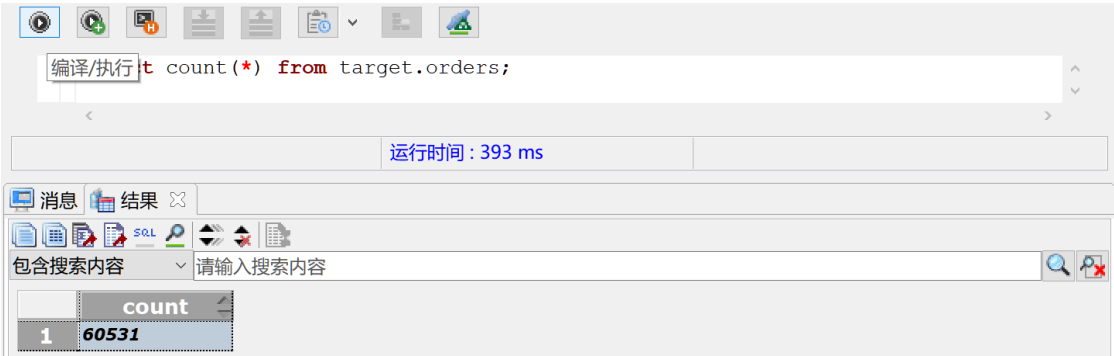
The screenshot shows a SQL query execution window. The query is: `select count(*) from target.address_dimension;`. The execution time is 379 ms. The result is displayed in a table with one row and one column, showing a count of 15001.

	count
1	15001

请在上方附上图片

orders 表

请在下方附上图片



The screenshot shows a SQL query execution window. The query is: `select count(*) from target.orders;`. The execution time is 393 ms. The result is displayed in a table with one row and one column, showing a count of 60531.

	count
1	60531

请在上方附上图片

order\_goods 表

请在下方附上图片



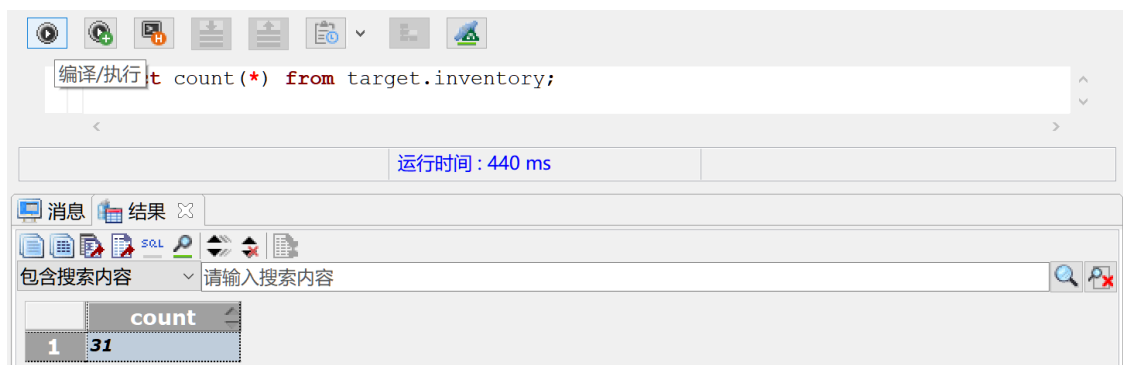
The screenshot shows a SQL query execution window. The query is: `select count(*) from target.order_goods;`. The execution time is 443 ms. The result is displayed in a table with one row and one column, showing a count of 63701.

	count
1	63701

请在上方附上图片

inventory 表

请在下方附上图片



请在上方附上图片

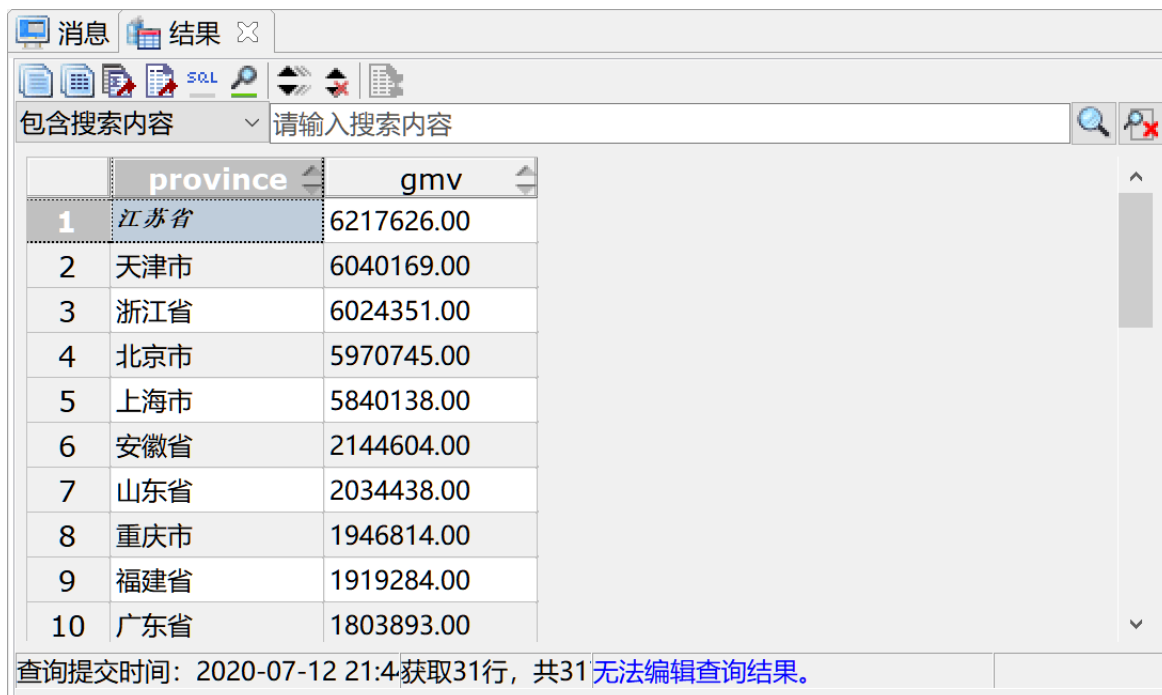


## 数据分析（40 分）

请补全查询语句，将补全的 SQL 语句以及结果截图贴到以下对应地方（结果截图，SQL 语句直接粘贴）。

1. 查看 2020 年 3 月各省 GMV，降序输出（5 分）

```
SELECT ad.province AS province, sum(o.actual_price) AS GMV
FROM target.orders o,
target.address_dimension ad,
target.date_dimension dd
WHERE o.address_key = ad.address_key
AND o.add_date = dd.date_key
AND dd.year = 2020
AND dd.month = 3
GROUP BY province
ORDER BY GMV DESC;
```

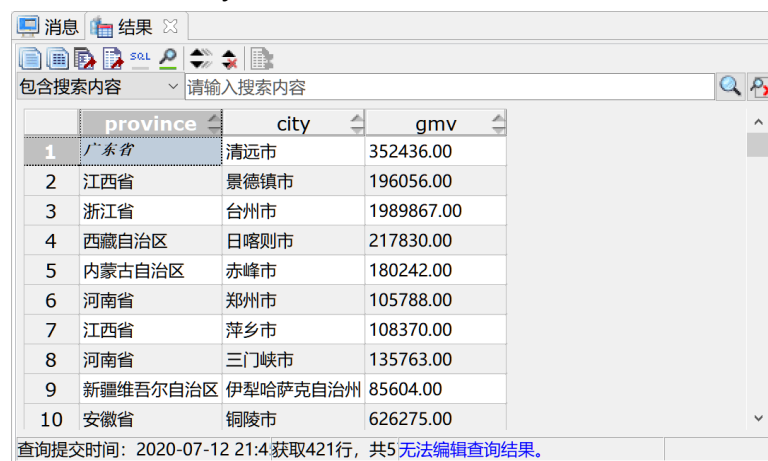


	province	gmv
1	江苏省	6217626.00
2	天津市	6040169.00
3	浙江省	6024351.00
4	北京市	5970745.00
5	上海市	5840138.00
6	安徽省	2144604.00
7	山东省	2034438.00
8	重庆市	1946814.00
9	福建省	1919284.00
10	广东省	1803893.00

查询提交时间: 2020-07-12 21:4 获取31行, 共31 无法编辑查询结果。

2. 查看全国各省市 GMV，时间范围为年初到今天（5 分）

```
SELECT ad.province AS province,  
ad.city AS city,  
sum(o.actual_price) AS GMV  
FROM target.orders o,  
target.address_dimension ad,  
target.date_dimension dd  
WHERE o.address_key = ad.address_key  
AND o.add_date = dd.date_key  
AND to_date(dd.full_date) BETWEEN to_date('2020/1/1') AND current_date  
AND province not in('北京市', '上海市', '天津市', '重庆市')  
GROUP BY province, city  
UNION ALL  
SELECT ad.province AS province,  
ad.county AS city,  
sum(o.actual_price) AS GMV  
FROM target.orders o,  
target.address_dimension ad,  
target.date_dimension dd  
WHERE o.address_key = ad.address_key  
AND o.add_date = dd.date_key  
AND to_date(dd.full_date) BETWEEN to_date('2020/1/1') AND current_date  
AND province in('北京市', '上海市', '天津市', '重庆市')  
GROUP BY province, ad.county;
```

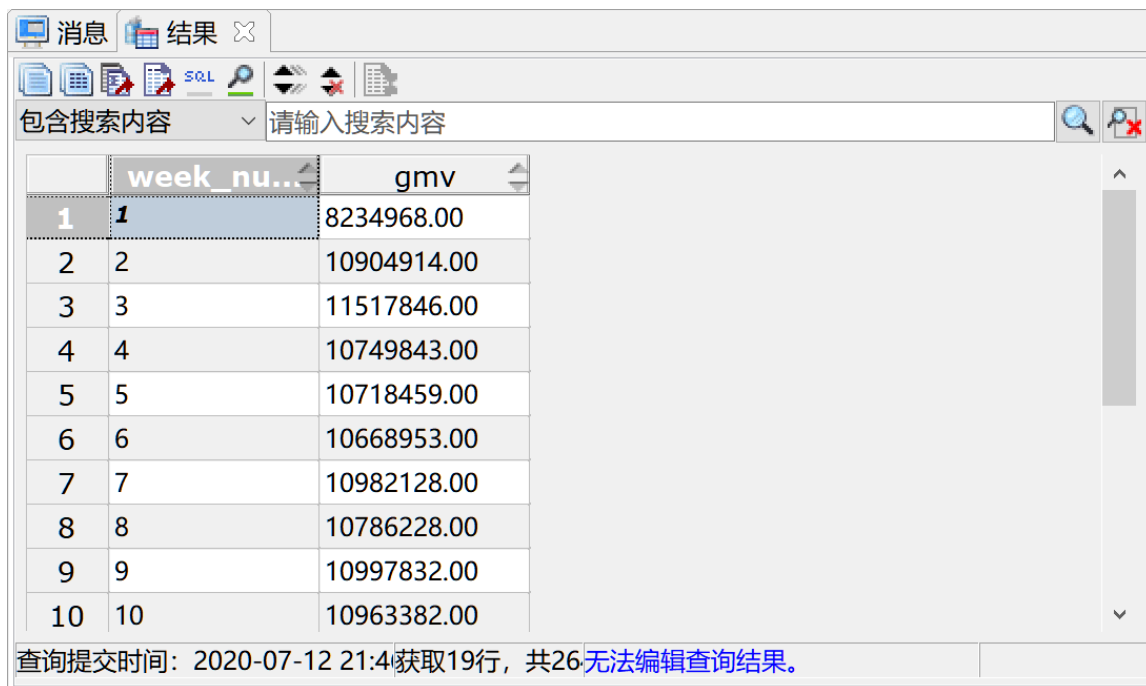


	province	city	gmv
1	广东省	清远市	352436.00
2	江西省	景德镇市	196056.00
3	浙江省	台州市	1989867.00
4	西藏自治区	日喀则市	217830.00
5	内蒙古自治区	赤峰市	180242.00
6	河南省	郑州市	105788.00
7	江西省	萍乡市	108370.00
8	河南省	三门峡市	135763.00
9	新疆维吾尔自治区	伊犁哈萨克自治州	85604.00
10	安徽省	铜陵市	626275.00

查询提交时间: 2020-07-12 21:4 获取421行, 共5 无法编辑查询结果。

3. 查看每周的总体 GMV，按照星期升序输出（5 分）

```
SELECT dd.week_num_in_year, sum(o.actual_price) AS GMV
FROM target.orders o, target.date_dimension dd
WHERE o.add_date = dd.date_key
GROUP BY dd.week_num_in_year
ORDER BY dd.week_num_in_year ASC;
```



The screenshot shows a database query result interface. At the top, there are tabs for '消息' (Messages) and '结果' (Results). Below the tabs is a toolbar with icons for file operations, SQL editing, and search. A search bar is present with the text '包含搜索内容' and a placeholder '请输入搜索内容'. The main area displays a table with two columns: 'week\_num\_in\_year' and 'gmv'. The table contains 10 rows of data, with the first row highlighted. At the bottom, a status bar shows the query submission time and the number of rows retrieved.

	week_num_in_year	gmv
1	1	8234968.00
2	2	10904914.00
3	3	11517846.00
4	4	10749843.00
5	5	10718459.00
6	6	10668953.00
7	7	10982128.00
8	8	10786228.00
9	9	10997832.00
10	10	10963382.00

查询提交时间：2020-07-12 21:4 获取19行，共26 无法编辑查询结果。

4. 查看全国各省市今年购买过的用户数，降序输出（5分）

```
SELECT ad.province, ad.city, count(DISTINCT user_key) AS totaluser
FROM target.orders o,
target.address_dimension ad,
target.date_dimension dd
WHERE o.add_date = dd.date_key
AND o.address_key = ad.address_key
AND dd.year = 2020
GROUP BY province, city
ORDER BY totaluser DESC;
```

消息 结果			
包含搜索内容 请输入搜索内容			
	province	city	totaluser
1	天津市	市辖区	1947
2	上海市	市辖区	1878
3	北京市	市辖区	1860
4	重庆市	市辖区	314
5	重庆市	市辖区	296
6	浙江省	舟山市	193
7	浙江省	绍兴市	189
8	浙江省	湖州市	187
9	浙江省	金华市	182
10	浙江省	嘉兴市	181

查询提交时间：2020-07-12 21:4 获取340行，共3 无法编辑查询结果。

5. 查看用户消费金额（不考虑退货），降序输出（5分）

```
SELECT ud.username,
gender,
sum(o.actual_price) AS totalspend
FROM target.orders o, target.user_dimension ud
WHERE o.user_key = ud.user_key
GROUP BY username, gender
ORDER BY totalspend DESC;
```

消息 结果			
包含搜索内容 请输入搜索内容			
	username	gender	totalspend
1	华小轩	1	41721.00
2	纪晓娟	2	39232.00
3	贾长丽	2	37261.00
4	贝成刚	1	37194.00
5	齐严刚	1	36383.00
6	宣高强	1	36084.00
7	林小轩	1	36084.00
8	沈严强	1	35783.00
9	韦兰丽	2	35681.00
10	危春娟	2	35443.00

查询提交时间: 2020-07-12 21:4 已获取1,000行, [无法编辑查询结果。](#)

6. 查看各商品的总体销量（不考虑退货），降序输出（5分）

```
SELECT gd.goods_name,
sum(og.number) AS totalnum,
sum(og.number * og.price) AS totalsales
FROM target.order_goods og
LEFT JOIN target.goods_dimension gd ON og.goods_key = gd.goods_key GROUP BY
gd.goods_key
ORDER BY totalnum DESC;
```

消息 结果			
包含搜索内容 请输入搜索内容			
	goods_name	totalnum	totalsales
1	荣耀FlyPods青春版 真无线耳机	2941	967589.00
2	HUAWEI X Gentle Monster Eyewear 智能眼镜	2935	5867065.00
3	荣耀 FlyPods 3真无线耳机	2916	2184084.00
4	HUAWEI FreeLace 无线耳机	2901	1157499.00
5	HUAWEI FreeBuds 3 无线耳机	2838	2977062.00
6	HUAWEI MateBook 13 2020款	2647	15850236.00
7	HUAWEI MateBook X Pro 2019款 13.9英寸	2635	20023365.00
8	华为平板 M6 10.8英寸	2584	9041416.00
9	HUAWEI MateBook 14 2020款	2546	14764254.00
10	HUAWEI MateBook D 2018款 15.6英寸	2537	13161956.00

查询提交时间：2020-07-12 21:5 获取30行，共36 无法编辑查询结果。

7. 查看各商品的月度销量（不考虑退货），按照商品及月份升序输出（5分）

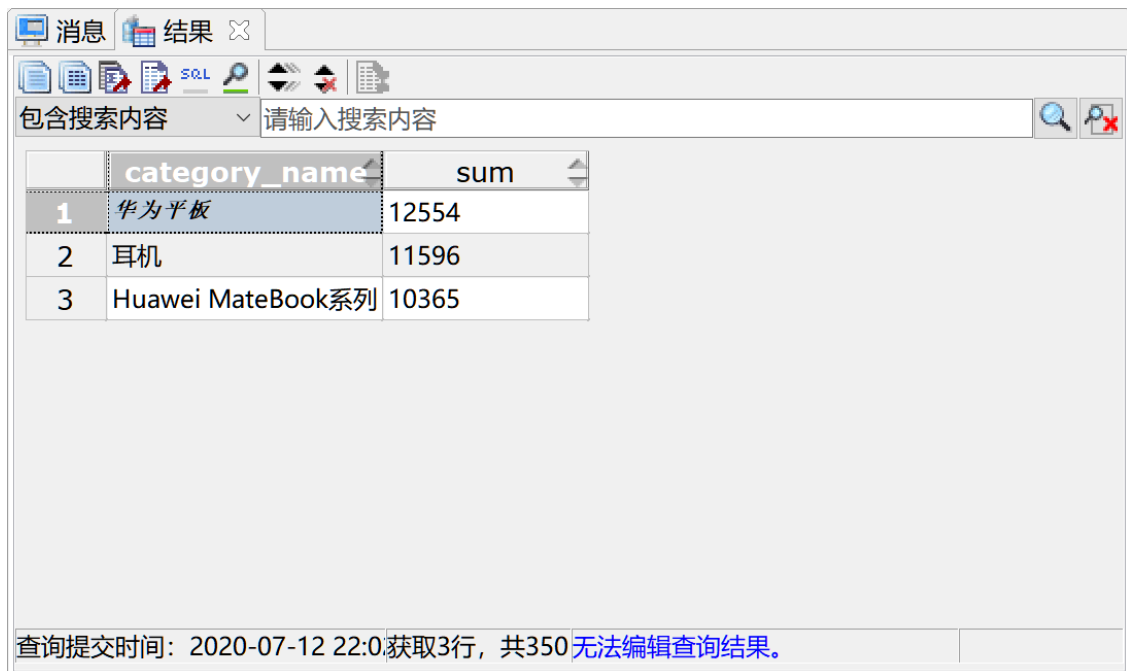
```
SELECT gd.goods_name, dd.month, sum(og.number) AS monthnum
FROM target.order_goods og
LEFT JOIN target.goods_dimension gd ON og.goods_key = gd.goods_key
LEFT JOIN target.date_dimension dd ON og.add_date = dd.date_key
GROUP BY gd.goods_name, dd.month
ORDER BY gd.goods_name, dd.month ASC;
```

消息 结果			
包含搜索内容 请输入搜索内容			
	goods_name	month	monthnum
1	HHUAWEI Mate 30 Pro 5G	1	463
2	HHUAWEI Mate 30 Pro 5G	2	407
3	HHUAWEI Mate 30 Pro 5G	3	460
4	HHUAWEI Mate 30 Pro 5G	4	429
5	HUAWEI FreeBuds 3 无线耳机	1	746
6	HUAWEI FreeBuds 3 无线耳机	2	636
7	HUAWEI FreeBuds 3 无线耳机	3	760
8	HUAWEI FreeBuds 3 无线耳机	4	696
9	HUAWEI FreeLace 无线耳机	1	719
10	HUAWEI FreeLace 无线耳机	2	669

查询提交时间：2020-07-12 22:00 获取117行，共2 [无法编辑查询结果。](#)

## 8. 查看销量最高的前三个类目（仅从数量上考虑销量）（5分）

```
SELECT gd.category_name, sum(og.number) AS sum
FROM target.order_goods og
LEFT JOIN target.goods_dimension gd ON og.goods_key = gd.goods_key
GROUP BY gd.category_name
ORDER BY sum DESC
LIMIT 3;
```



The screenshot shows a database query result interface. At the top, there are tabs for '消息' (Messages) and '结果' (Results). Below the tabs is a toolbar with icons for SQL, search, and other functions. A search bar is present with the text '包含搜索内容' and a placeholder '请输入搜索内容'. The main area displays a table with the following data:

	category_name	sum
1	华为平板	12554
2	耳机	11596
3	Huawei MateBook系列	10365

At the bottom of the interface, there is a status bar that reads: '查询提交时间: 2020-07-12 22:0 获取3行, 共350 无法编辑查询结果。'