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）

这三种架构都是由源事务，后端，前端三部分组成。架构是数据仓库建设的总体规划，从整体视角描述了解决方案的高层模型，描述了各个子系统的功能以及关系，描述了数据从源系统到决策系统的数据流程。业务需求回答了要做什么，架构就是回答怎么做的问题。数据仓库的核心功能从源系统抽取数据，通过清洗、转换、标准化，将数据加载到 BI 平台，进而满足业务用户的数据分析和决策支持。

数据仓库架构包含三个部分：数据架构、应用程序架构、底层设施。

Inmon 架构中企业数据仓库规范化是强制性的构件。但是这种规范化仅仅是建立在实现多对一关系的物理表。在应用中比较适用于专家团队，初期投入较高，维护容易。

Kimball 分工明确，资源占用更加合理，调用链路少，整个 DW / BI 系统更加稳定、高效、有保障。在应用中，一般适用于一般开发团队，初期投入较少，但不容易维护。

Inmon 架构包含聚集数据，不够灵活。Kimball 架构关注解决数据不一致性，但并未明确提出需要规范化。

独立数据集市架构不需要考虑跨组织的数据控制和协调的问题，从短期效果来看，有利于较低成本实现快速开发。但从长远来看，从相同的数据源重复获取数据，由于分析数据的冗余存储造成浪费和低效。

2. 粒度举例。（5 分）

粒度由大到小比如商品和货品、肉和食物，此外，客户的零售单据订单上的每个条目，每个订单所代表的一次交易，同类型的商家信息，相似偏好的买家信息都是不同的粒度单位。

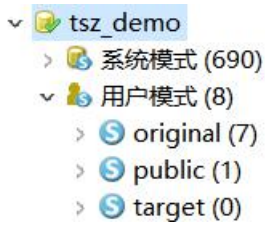
3. 维度举例。（5 分）

大多数维度都具有一个或多个层次如日期维度有年、季度、月和日四级，生物学上有界门纲目科属种七个维度来归类划分不同生物，而订单相关的维度有销售渠道、销售门店、销售对象、商品等。

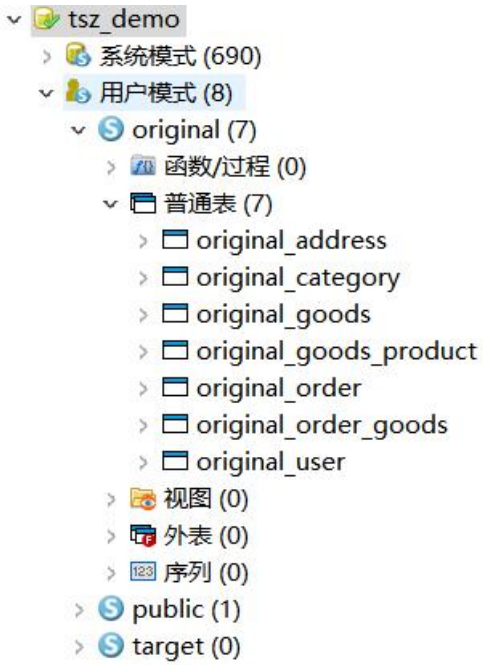
实验

数据库对象创建（15 分）

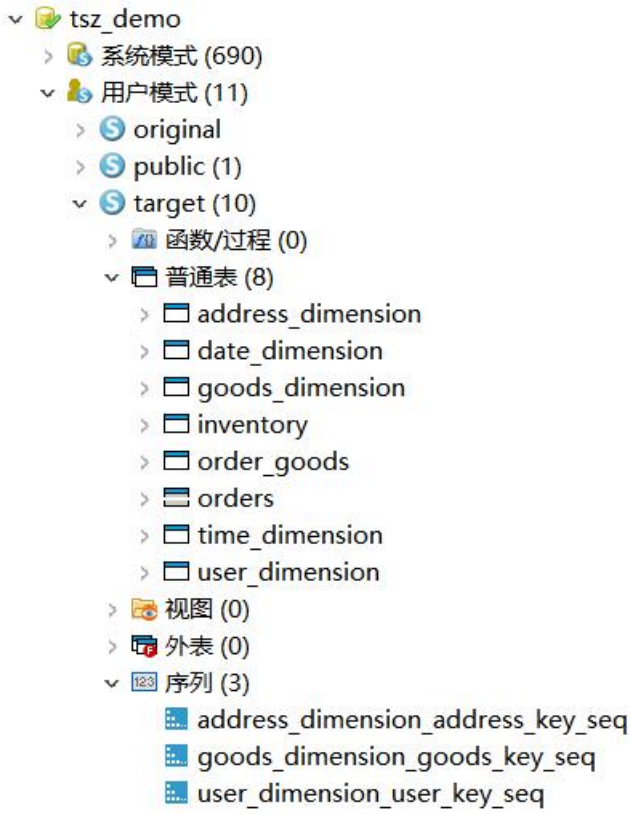
1. 截图内容：创建的个人数据库及两个模式。（5 分）



2. 截图内容：original 模式下所创建的表。（5 分）

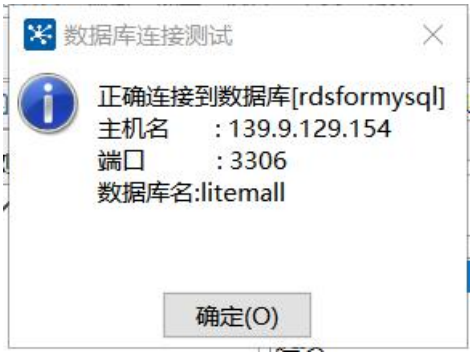


3. 截图内容：target 模式下所创建的表和序列。（5 分）



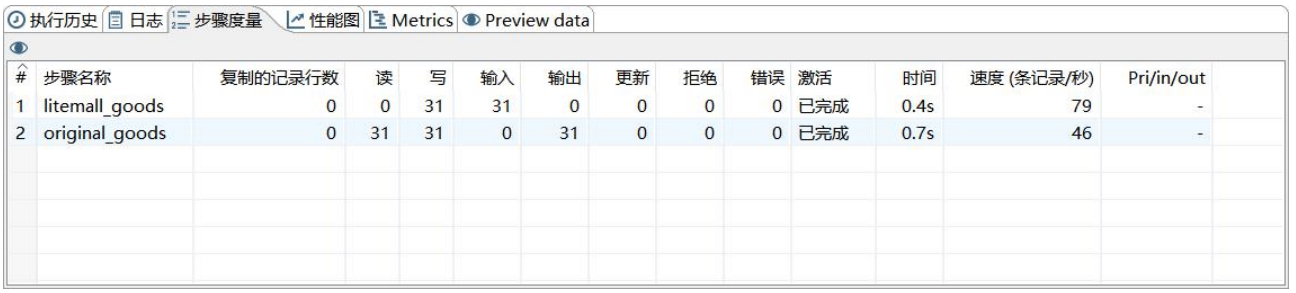
数据转换（25 分）

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）



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



#	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	0.4s	79	-
2	original_goods	0	31	31	0	31	0	0	0	已完成	0.7s	46	-

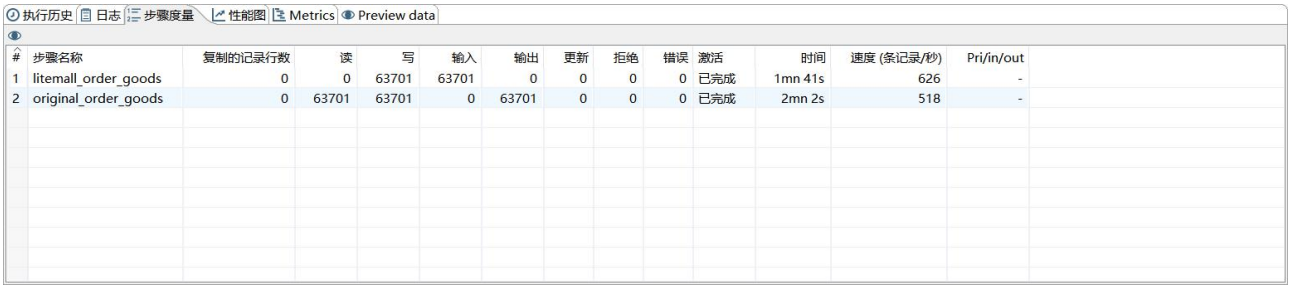
goods_product 转换（3 分）

#	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	0.2s	174	-
2	original_goods_product	0	31	31	0	31	0	0	0	已完成	0.4s	71	-

order 转换（3 分）

#	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	60530	60530	0	0	0	0	已完成	2mn 8s	470	-
2	original_order	0	60530	60530	0	60530	0	0	0	已完成	2mn 28s	408	-

order_goods 转换（3 分）



#	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	1mn 41s	626	-
2	original_order_goods	0	63701	63701	0	63701	0	0	0	已完成	2mn 2s	518	-

address 转换

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	9.8s	1,534	-
2	original_address	0	15001	15001	0	15001	0	0	0	已完成	24.8s	605	-

user 转换

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	8.8s	1,706	-
2	original_user	0	15001	15001	0	15001	0	0	0	已完成	24.5s	613	-

category 转换

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_category	0	0	21	21	0	0	0	0	已完成	0.3s	84	-
2	original_category	0	21	21	0	21	0	0	0	已完成	0.5s	46	-

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）

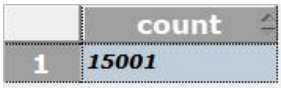
goods_dimension 表

	count
1	31

user_dimension 表

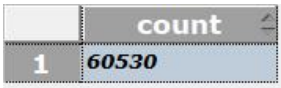
	count
1	15001

address_dimension 表



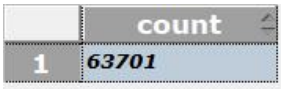
	count
1	15001

orders 表



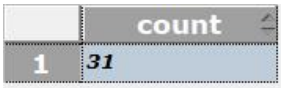
	count
1	60530

order_goods 表



	count
1	63701

inventory 表



	count
1	31

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
11	甘肃省	491934.00

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, city;
```

	province	city	gmv
1	西藏自治区	林芝市	266164.00
2	广西壮族自治区	百色市	197052.00
3	新疆维吾尔自治区	克拉玛依市	88740.00
4	广东省	清远市	352436.00
5	江西省	吉安市	124275.00
6	云南省	迪庆藏族自治州	80484.00
7	浙江省	台州市	1989867.00
8	浙江省	绍兴市	2361149.00

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;
```

	week_nu...	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

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 o.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 ad.province, ad.city
```

ORDER BY totaluser DESC;

	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	1859
4	重庆市	市辖区	314
5	重庆市	市辖县	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

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 ud.username, gender
ORDER BY totalspend DESC;

	username	gender	totalspend
1	华小轩	1	41721.00
2	纪晓娟	2	39232.00
3	贲长丽	2	37261.00
4	贝成刚	1	37194.00
5	齐严刚	1	36383.00
6	林小轩	1	36084.00
7	宣高强	1	36084.00
8	沈严强	1	35783.00

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.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_name
ORDER BY totalnum DESC;

	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

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;
```

	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

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;
```

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