- 首先创建贴源层 ODS 库

Create database game_ods;

- 为原始数据创建表 origin_data：

```
create table `origin_data`(
user_id bigint, register_time timestamp, wood_add_value double, wood_reduce_value double,
stone_add_value double, stone_reduce_value double, ivory_add_value double,
ivory_reduce_value double, meat_add_value double, meat_reduce_value double,
magic_add_value double, magic_reduce_value double, infantry_add_value int,
infantry_reduce_value int, cavalry_add_value int, cavalry_reduce_value int, shaman_add_value
int, shaman_reduce_value int, wound_infantry_add_value int, wound_infantry_reduce_value int,
wound_cavalry_add_value int, wound_cavalry_reduce_value int, wound_shaman_add_value int,
wound_shaman_reduce_value int, general_acceleration_add_value int,
general_acceleration_reduce_value int, building_acceleration_add_value int,
building_acceleration_reduce_value int, reaserch_acceleration_add_value int,
reaserch_acceleration_reduce_value int, training_acceleration_add_value int,
training_acceleration_reduce_value int, treatment_acceleraion_add_value int,
treatment_acceleration_reduce_value int, bd_training_hut_level int, bd_healing_lodge_level int,
bd_stronghold_level int, bd_outpost_portal_level int, bd_barrack_level int,
bd_healing_spring_level int, bd_dolmen_level int, bd_guest_cavern_level int,
bd_warehouse_level int, bd_watchtower_level int, bd_magic_coin_tree_level int,
bd_hall_of_war_level int, bd_market_level int, bd_hero_gacha_level int,
bd_hero_strengthen_level int, bd_hero_pve_level int, sr_scout_level int, sr_training_speed_level
int, sr_infantry_tier_2_level int, sr_cavalry_tier_2_level int, sr_shaman_tier_2_level int,
sr_infantry_atk_level int, sr_cavalry_atk_level int, sr_shaman_atk_level int,
sr_infantry_tier_3_level int, sr_cavalry_tier_3_level int, sr_shaman_tier_3_level int,
sr_troop_defense_level int, sr_infantry_def_level int, sr_cavalry_def_level int,
sr_shaman_def_level int, sr_infantry_hp_level int, sr_cavalry_hp_level int, sr_shaman_hp_level
int, sr_infantry_tier_4_level int, sr_cavalry_tier_4_level int, sr_shaman_tier_4_level int,
sr_troop_attack_level int, sr_construction_speed_level int, sr_hide_storage_level int,
sr_troop_consumption_level int, sr_rss_a_prod_levell int, sr_rss_b_prod_level int,
sr_rss_c_prod_level int, sr_rss_d_prod_level int, sr_rss_a_gather_level int, sr_rss_b_gather_level
int, sr_rss_c_gather_level int, sr_rss_d_gather_level int, sr_troop_load_level int,
sr_rss_e_gather_level int, sr_rss_e_prod_level int, sr_outpost_durability_level int,
sr_outpost_tier_2_level int, sr_healing_space_level int, sr_gathering_hunter_buff_level int,
sr_healing_speed_level int, sr_outpost_tier_3_level int, sr_alliance_march_speed_level int,
sr_pvp_march_speed_level int, sr_gathering_march_speed_level int, sr_outpost_tier_4_level int,
sr_guest_troop_capacity_level int, sr_march_size_level int, sr_rss_help_bonus_level int,
pvp_battle_count int, pvp_lanch_count int, pvp_win_count int, pve_battle_count int,
pve_lanch_count int, pve_win_count int, avg_online_minutes double, pay_price double,
pay_count int, prediction_pay_price double)
row format delimited
fields terminated by    ','
tblproperties(
"skip.header.line.count"="1"
);
```

- 查出全表记录条数和独立用户记录数，发现每条记录都是独立的用户，(因此后续不会使用 distinct user_id 以提升计算速度)

```
select count(1) as total_records, count(distinct user_id) as unique_users from origin_data;
```

```
total_records   unique_users
2288007 2288007
Time taken: 14.76 seconds, Fetched: 1 row(s)
```

- 综合监控指标脑图里所有指标，梳理发现需要添加两个基础属性以方便后续处理分析：添加流失用户、非流失用户标记`user_type`

添加消费程度标记`consumption_class`
添加日期字段`date`，(从原来的 register_time 转换过来)
故补充上这两个属性，创建 DWD 库，user_info 表：

```
create table user_info as
select *,
to_date(register_time) as `date`,
case when (wood  reduce  value<0.00001 and meat  reduce_value<0.00001 and
avg_online_minutes<=2.0) or pay_count=0 then 'lost'
else 'remain'
end as user_type,
case when pay_count=0 then 'zero'
when pay  price<=68.0 then 'low'
when pay_price<=202.0 then 'middle'
else 'high'
end as consumption_class
from game_ods.origin_data;
```

- 创建 DWS 库，开始聚合汇总计算相关指标

按日期统计每日新用户、流失用户、付费用户、新用户付费率、ARPU、ARPPU、新用户流失率

```
create table user_quality_table as
select `date`, count(1) as new_users, count(if (user_type='lost',1,null)) as lost_users,
concat(round(count(if (pay_count!=0,1,null))/count(1)*100,2),'%') as pay_percentage,
count(if (pay_count>0,1,null)) as paying_users,
round(sum(pay_price)/count(1),2) as ARPU,
round(sum(pay_price)/count(if (pay_count>0,1,null)),2) as ARPPU,
concat(round(count(if (user_type='lost',1,null))/count(1)*100,2),'%') as lost_percentage
from game_dwd.user_info group by `date` order by `date` asc;
```

date	new_users	lost_users	pay_percentage	paying_users	arpu	arppu	lost_percentag
2018-01-26	70250	68622	2.33%	1636	0.83	35.58	97.68%
2018-01-27	70417	68904	2.16%	1519	0.41	19.09	97.85%
2018-01-28	79227	77721	1.91%	1513	0.42	22.23	98.1%
2018-01-29	63803	62311	2.35%	1498	0.57	24.08	97.66%
2018-01-30	50201	48968	2.46%	1236	0.78	31.55	97.54%
2018-01-31	56522	55494	1.83%	1032	0.35	19.38	98.18%
2018-02-01	83245	82120	1.35%	1126	0.3	22.53	98.65%
2018-02-02	60173	58782	2.31%	1393	0.85	36.58	97.69%
2018-02-03	51659	50460	2.32%	1201	0.74	31.88	97.68%
2018-02-04	60421	59267	1.91%	1155	0.44	23.03	98.09%
2018-02-05	60998	59642	2.22%	1356	0.68	30.77	97.78%
2018-02-06	57203	56213	1.73%	992	0.41	23.83	98.27%
2018-02-07	71576	70601	1.36%	976	0.45	33.34	98.64%
2018-02-08	72402	71389	1.4%	1016	0.48	34.09	98.6%
2018-02-09	50143	49032	2.22%	1111	0.65	29.5	97.78%
2018-02-10	53521	52483	1.94%	1039	0.68	35.14	98.06%
2018-02-11	54014	53054	1.78%	962	0.41	22.76	98.22%
2018-02-12	52231	51028	2.31%	1206	0.82	35.52	97.7%
2018-02-13	50638	49797	1.66%	841	0.46	27.69	98.34%
2018-02-14	54419	53722	1.29%	703	0.48	37.53	98.72%
2018-02-15	78707	78024	0.87%	687	0.25	28.95	99.13%
2018-02-16	56355	55366	1.75%	989	0.81	46.21	98.25%
2018-02-17	44477	43728	1.68%	749	0.4	23.74	98.32%
2018-02-18	59447	58663	1.32%	784	0.4	30.08	98.68%
2018-02-19	117311	116261	0.9%	1052	0.37	41.32	99.1%
2018-02-20	92860	92022	0.9%	839	0.27	30.02	99.1%
2018-02-21	43720	42699	2.34%	1021	0.44	18.65	97.66%
2018-02-22	42110	41278	1.98%	834	0.48	24.12	98.02%
2018-02-23	44635	43628	2.26%	1007	1.04	46.18	97.74%
2018-02-24	45648	44783	1.9%	866	0.7	37.01	98.11%
2018-02-25	49835	48980	1.72%	855	0.52	30.02	98.28%
2018-02-26	42647	41613	2.42%	1034	0.96	39.46	97.58%
2018-02-27	39140	38445	1.78%	695	0.41	23.27	98.22%
2018-02-28	42928	42189	1.72%	740	0.53	30.58	98.28%
2018-03-01	36226	35560	1.84%	666	0.5	27.38	98.16%
2018-03-02	42775	41743	2.41%	1032	0.7	29.1	97.59%
2018-03-03	48970	48068	1.85%	904	0.39	21.12	98.16%
2018-03-04	50989	49996	1.95%	994	0.38	19.57	98.05%
2018-03-05	44726	43557	2.61%	1169	0.7	26.6	97.39%
2018-03-06	41438	40427	2.44%	1011	0.81	33.11	97.56%

Time taken: 0.168 seconds, Fetched: 40 row(s)

按照日期、按照用户消费程度统计平均在线时长和数量分布：

```
create table user_consumption_table as
select    `date`,
consumption_class,
round(avg(avg_online_minutes),2) as avg_online_time,
count(1) as count
from game_dwd.user_info
group by `date`, consumption_class
order by `date`asc , avg(avg_online_minutes) desc;
```

date	consumption_class	avg_online_time	count
2018-01-26	high	452.5	48
2018-01-26	middle	318.13	68
2018-01-26	low	134.58	1520
2018-01-26	zero	8.0	68614
2018-01-27	high	394.24	29
2018-01-27	middle	341.62	38
2018-01-27	low	131.02	1452
2018-01-27	zero	8.38	68898
2018-01-28	high	336.32	21
2018-01-28	middle	303.68	47
2018-01-28	low	130.21	1445
2018-01-28	zero	7.79	77714
2018-01-29	high	323.72	38
2018-01-29	middle	298.61	47
2018-01-29	low	143.86	1413
2018-01-29	zero	8.49	62305
2018-01-30	high	374.18	30
2018-01-30	middle	305.63	48
2018-01-30	low	133.52	1158
2018-01-30	zero	9.58	48965
2018-01-31	high	347.07	16
2018-01-31	middle	265.61	45
2018-01-31	low	120.51	971
2018-01-31	zero	8.69	55490
2018-02-01	middle	255.73	36
2018-02-01	high	251.99	22
2018-02-01	low	125.59	1068

加速卷获取使用情况

```
create table user_acceleration_situation_table as
select    `date`,
consumption_class,
round(avg(general_acceleration_add_value),1) as avg_general_add,
round(avg(general_acceleration_reduce_value),1) as avg_general_reduce,
round(avg(building_acceleration_add_value),1) as avg_building_add,
round(avg(building_acceleration_reduce_value),1) as avg_building_reduce,
round(avg(reaserch_acceleration_add_value),1) as avg_research_add,
round(avg(reaserch_acceleration_reduce_value),1) as avg_research_reduce,
round(avg(training_acceleration_add_value),1) as avg_training_add,
round(avg(training_acceleration_reduce_value),1) as avg_training_reduce,
round(avg(treatment_acceleraion_add_value),1) as avg_treatment_add,
round(avg(treatment_acceleration_reduce_value),1) as avg_treatment_reduce
from game_dwd.user_info
group by `date`, consumption_class
order by `date`asc;
```

2018-01-27	high	23351.0	21251.4	8533.8	6944.7	7507.0	7389.6	8877.0	9922.9	135.4	31.7
2018-01-27	zero	31.7	27.9	7.6	6.2	8.4	3.2	27.2	4.3	2.2	0.0
2018-01-27	middle	8430.7	7650.1	3094.8	2909.7	2235.0	1958.7	4377.3	2721.8	137.0	19.8
2018-01-28	low	1520.8	1129.6	670.5	643.5	685.4	515.5	766.3	511.4	61.1	16.0
2018-01-28	middle	16312.9	14349.4	3225.3	2932.2	3964.9	3723.6	4769.0	4730.5	159.0	65.4
2018-01-28	zero	55.9	42.8	11.7	7.8	12.4	4.3	37.2	7.2	2.6	0.2
2018-01-28	high	51813.0	49610.1	5004.2	7507.0	14028.9	13430.9	28094.1	29922.8	982.7	805.7
2018-01-29	middle	15023.4	11550.6	6607.3	5494.5	4466.4	5226.2	9846.2	6541.3	260.3	67.1
2018-01-29	zero	91.7	70.4	16.3	12.4	15.6	6.7	57.4	9.1	3.7	0.1
2018-01-29	high	42396.9	37398.3	14131.6	11675.0	8381.8	11104.4	22945.4	25375.6	376.2	70.9
2018-01-29	low	2448.2	1860.8	1098.6	1001.0	985.9	811.1	1044.6	768.4	91.5	13.1
2018-01-30	high	88397.6	81646.9	26667.1	26596.4	33008.0	32425.0	35575.7	37979.0	530.9	80.4
2018-01-30	low	3092.4	2185.8	1507.4	1317.7	1475.1	1172.3	1502.5	925.7	104.1	10.9
2018-01-30	middle	18576.7	14152.4	5317.7	5751.3	6398.8	5546.4	12396.0	8301.7	300.5	67.0
2018-01-30	zero	132.1	105.6	32.8	36.4	22.9	10.3	107.3	14.3	5.9	0.1
2018-01-31	low	3580.5	2638.5	1924.4	1830.2	2208.0	1746.0	1870.2	1076.3	135.4	6.8
2018-01-31	middle	24447.8	21957.2	7750.3	8240.7	11213.1	9906.5	16582.5	10474.0	353.2	35.7
2018-01-31	zero	150.0	110.9	82.7	85.5	32.7	11.8	168.4	14.3	6.6	0.0
2018-01-31	high	79459.2	70756.5	23742.8	24502.1	25637.8	23747.1	30123.4	26095.1	488.8	11.8
2018-02-01	middle	29149.9	25339.8	13707.1	12870.0	16266.6	13064.2	15710.7	10779.5	386.9	63.0
2018-02-01	high	88243.2	81083.8	32468.0	31829.5	37567.1	35195.9	60137.5	51538.4	813.0	26.7
2018-02-01	zero	183.2	95.6	230.1	150.5	50.0	10.8	281.8	16.5	5.6	0.0
2018-02-01	low	4587.1	3108.2	3875.5	2949.7	3842.9	2380.6	2635.0	1271.1	165.8	7.8
2018-02-02	high	108293.0		93315.0	51378.0	45863.9	48459.0	43138.2	56340.9	45413.6	651.0	131.3
2018-02-02	middle	32765.0	24922.6	18836.8	16908.4	16162.9	12752.8	19746.9	11192.3	469.9	71.5
2018-02-02	low	5578.3	3840.1	4591.3	3607.3	4366.9	2859.3	3110.5	1617.0	190.6	18.1
2018-02-02	zero	219.1	122.2	179.1	125.5	58.8	18.1	289.3	23.6	8.3	0.1
2018-02-03	middle	28141.7	23012.5	13516.7	12354.5	15259.8	11623.7	21179.0	13835.3	413.9	55.7
2018-02-03	zero	199.0	110.3	159.5	101.0	58.1	17.1	225.2	24.2	8.7	0.0
2018-02-03	high	107267.8		94139.7	43790.1	39829.4	40143.5	36744.3	60927.7	44444.4	834.9	131.4
2018-02-03	low	4592.3	3152.5	3903.9	3059.4	3822.8	2378.4	2703.1	1339.8	168.7	8.4
2018-02-04	high	83748.2	71696.7	29886.9	25402.1	53965.0	40320.0	46087.5	31916.2	449.1	59.5
2018-02-04	low	4694.7	3182.9	3967.0	2983.5	3967.9	2336.7	2795.1	1335.0	172.7	16.5
2018-02-04	zero	199.2	119.1	167.5	98.9	56.4	15.7	250.8	24.4	7.9	0.1
2018-02-04	middle	31175.8	25203.4	13778.7	12872.0	14954.5	12200.2	18784.0	11065.5	360.5	65.9
2018-02-05	low	5425.5	3608.2	4388.6	3416.9	4174.8	2677.0	3079.1	1711.0	191.8	19.1
2018-02-05	zero	247.5	151.6	177.3	111.0	58.1	16.1	323.7	25.3	8.1	0.1
2018-02-05	high	100070.2		80787.7	43722.2	38717.1	50316.4	41363.2	60950.3	45486.0	658.6	142.8
2018-02-05	middle	31304.7	24200.0	14963.1	12232.4	16865.0	12743.3	18967.0	10931.6	403.2	91.0
2018-02-06	middle	28535.8	24033.7	17571.2	15659.0	15513.3	12617.7	15096.8	12096.7	424.4	45.2
2018-02-06	high	99333.5	85891.6	34868.6	32813.8	37115.0	33753.1	51561.5	46714.2	929.0	172.0
2018-02-06	low	4690.1	3279.7	3940.5	2927.0	3888.4	2369.9	2825.8	1489.4	168.1	12.2

最后取出三张表去 Tableau 里进行可视化分析。


user_consumptio n_table.xlsx

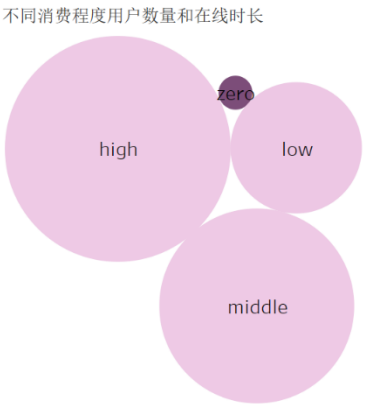

user_acceleratio n_situation_table.

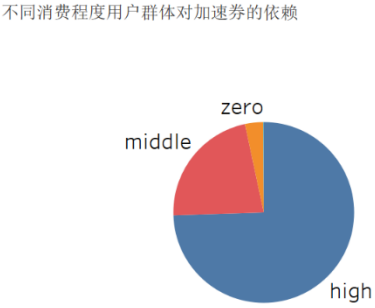

user_quality_tabl e.xlsx

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	date	consumption_class	avg_general_add	avg_general_reduce	avg_building_add	avg_building_reduce	avg_research_add	avg_research_red	avg_training_add	avg_training_reduc	avg_treatment_ad	avg_treatment_reduce	
2	2018/1/26	middle	4351.7	3037.8	2491.6	1931.2	546.5	557.6	3471.4	1780.9	58.2	11.5	
3	2018/1/26	zero	12.5	10	4.2	3.6	3.9	1.6	12.3	1.8	0.8	0	
4	2018/1/26	high	19950.4	18911.3	6635	6221	3993.2	5146.6	8853	8526.8	70.4	12.6	
5	2018/1/26	low	566	441.4	290.6	267.3	220.9	147.4	253	146.1	23.5	1.9	

	date	consumption_class	avg_online_time	count
2	2018/1/26	high	452.5	48
3	2018/1/26	middle	318.13	68
4	2018/1/26	low	134.58	1520
5	2018/1/26	zero	8	68614
6	2018/1/27	high	394.24	29
7	2018/1/27	middle	341.62	38
8	2018/1/27	low	131.02	1452
9	2018/1/27	zero	8.38	68898

	A	B	C	D	E	F	G	H
1	date	new_users	lost_users	pay_percentage	paying_users	arpu	arppu	lost_percentage
2	2018/1/26	70250	68622	2.33%	1636	0.83	35.58	97.68%
3	2018/1/27	70417	68904	2.16%	1519	0.41	19.09	97.85%
4	2018/1/28	79227	77721	1.91%	1513	0.42	22.23	98.10%
5	2018/1/29	63803	62311	2.35%	1498	0.57	24.08	97.66%
6	2018/1/30	50201	48968	2.46%	1236	0.78	31.55	97.54%
7	2018/1/31	56522	55494	1.83%	1032	0.35	19.38	98.18%
8	2018/2/1	83245	82120	1.35%	1126	0.3	22.53	98.65%
9	2018/2/2	60173	58782	2.31%	1393	0.85	36.58	97.69%





作图分析：

左上的【不同消费程度用户数量和在线时长】图里，圆圈越大代表群体平均在线时长越长，颜色越深代表群体人数越大，这个结果符合我们的常识。因此我们要想尽办法让用户多停留几分钟。设计各种巧妙的细节，让用户不经意多停留几分钟，增加转化率。

右边的【用户新增/留存/付费情况】图里，我们可以看到流失率相当高，还是说明了我们的产品很难短时间内抓住用户的 aha 时刻。特别注意到添加了红色参考线的四个日期，都是同时出现了新用户数正尖峰和流失率/付费率倒尖峰，从数学上这不难理解其实只是分母变大，数据是正常波动的。底部的 ARPPU 和 ARPU 也是呈现出较大的差异，说明我们的忠诚(付费)用户明显比不花钱用户更敢于花钱。但这还不够，我们得让付费用户持续付费，付更多的费，同时尽量转化不付费用户。

左下角的【不同消费程度用户群体对加速券的依赖】图里，可以发现一个用户对加速券的依赖可以侧面反映出他是哪个消费程度的用户。那么加速券就是一个需要业务人员深挖的点，究竟如何设计加速券的获得和使用能够更好的黏住用户，刺激用户消费欲，还待探究和实验。是一个好的突破点。此外根据用户对加速券的依赖程度随时间的变化预测该用户对本款游戏的心理预期，适当推荐相应的游戏套餐和充值计划，以及微妙的游戏内环节设计，更好的黏住用户，提高消费。