DB HW1 Report

0716234 蕭彧

patient_info

Field	Type	Null	Key	Default	Extra
patient_id	varchar(10)	NO	PRI	NULL	
sex	varchar(10)	YES		NULL	
age	int	YES		NULL	
province	varchar(20)	YES		NULL	
city	varchar(20)	YES		NULL	
infection_case	varchar(100)	YES		NULL	

search_trend

Field	Type	Null	Key	Default	Extra
date	date	NO	PRI	NULL	
cold	float	YES		NULL	
flu	float	YES		NULL	
pneumonia	float	YES		NULL	
coronavirus	float	YES		NULL	

time

Field	Type	Null	Key	Default	Extra
date	date	NO	PRI	NULL	
test	int	YES		NULL	
negative	int	YES		NULL	
confirmed	int	YES		NULL	
released	int	YES		NULL	
deceased	int	YES		NULL	

time_age

Field	Type	Null	Key	Default	Extra
date	date	NO	PRI	NULL	
age	int	NO	PRI	NULL	
confirmed	int	YES		NULL	
deceased	int	YES		NULL	

time_gender

Field	Type	Null	Key	Default	Extra
date	date	NO	PRI	NULL	
sex	varchar(10)	NO	PRI	NULL	
confirmed	int	YES		NULL	
deceased	int	YES		NULL	

time_province

Field	Type	Null	Key	Default	Extra
date	date	NO	PRI	NULL	
province	varchar(20)	NO	PRI	NULL	
confirmed	int	YES		NULL	
released	int	YES		NULL	
deceased	int	YES		NULL	

Region

Field	Type	Null	Key	Default	Extra
code	int	NO	PRI	NULL	
province	varchar(20)	YES		NULL	
city	varchar(20)	YES		NULL	
elementary_school_count	int	YES		NULL	
kindergarten_count	int	YES		NULL	
university_count	int	YES		NULL	
elderly_population_ratio	float	YES		NULL	
elderly_alone_ratio	float	YES		NULL	
nursing_home_count	int	YES		NULL	

weather

Field	Type	Null	Key	Default	Extra
code	int	NO	PRI	NULL	
date	date	NO	PRI	NULL	
avg_temp	float	YES		NULL	
most_wind_direction	int	YES		NULL	
avg_relative_humidity	float	YES		NULL	

QUESTIONS

1. (3%) What is the difference between type “char” and type “varchar”? 譯：變數型態 “char” 和 “varchar” 有什麼不同？

A: The main difference between these two is that char is fix length but varchar is variable length. For example, if we store ‘ABC’ using char[10] and varchar[10], Char[10] will use 3 spaces store ‘ABC’ and the rest space will be filled with space character, but varchar[10] will use 4 spaces, 3 for ‘ABC’ and 1 for the length of ‘ABC. The same is that when the length of the string is over 10, both of them will cut the string off to length of 10. Char has better loading speed than varchar, since char is fix size, but char has worse space utilization than varchar, since varchar is variable size. We need to trade of between speed of loading data and space utilization when we consider which one to use.

2. (3%) How many bytes it should take for “tinyint”, “smallint”, “mediumint”, “”? (e.g. 8 bytes for “bigint”) And what’s the range they can express? (e.g. from -1000 to 1000) 譯：“tinyint”, “smallint”, “mediumint”, “int” 各需要多少 bytes 來儲存？(e.g. 8 bytes for “bigint”) 還有他們的表示範圍可以從哪裡到哪裡？(e.g. from -1000 to 1000)

A: tinyint: 1 bytes, from -128 to 127
Smallint: 2 bytes, from -32768 to 32767
Mediumint: 3 bytes, from -8388608 to 8388607
Int: 4 bytes, from -2147483648 to 2147483647

3. (4%) What do you think about this DB schema? If you can change this table architecture, how would you modify it and why? 譯：你對這資料庫架構有什麼想法？如果你可以修改這架構，你會怎麼改？為什麼？

A: I would say that the three time_xxxx are too redundant. I will put these three together into a big table name time_attribute or something like that. In this table, age, gender, province and release will all in it. By this table, I can save a lot of

spaces, and the complexity of finding the data won't be increase a lot because I just literally combine some columns together in this table.

Task 11:

```
select  sum(case when m.home_team_score > m.away_team_score then 1 else 0 end) / count(*) as home_team_win_rate,
        sum(case when ((p.home_strong > p.away_strong) and (m.home_team_score > m.away_team_score)) then 1
                 when ((p.away_strong > p.home_strong) and (m.away_team_score > m.home_team_score)) then 1 else 0 end) / count(*) as strong_team_win_rate
from    (select  p.id,
                 sum(case when p.home_rate is not null then p.home_rate else 0 end) / count(p.home_rate) as home_strong,
                 sum(case when p.away_rate is not null then p.away_rate else 0 end) / count(p.away_rate) as away_strong
        from     (select  p.id, p.home_team_id, p.home_player, p.home_rate, p.away_team_id, p.away_player,  max(pa2.overall_rating) as away_rate
                  from    (select  p.id, p.home_team_id, p.home_player, max(pa1.overall_rating) as home_rate, p.away_team_id, p.away_player, p.date
                           from    (select id, home_team_id, home_player_1 as home_player, away_team_id, away_player_1 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_2 as home_player, away_team_id, away_player_2 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_3 as home_player, away_team_id, away_player_3 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_4 as home_player, away_team_id, away_player_4 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_5 as home_player, away_team_id, away_player_5 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_6 as home_player, away_team_id, away_player_6 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_7 as home_player, away_team_id, away_player_7 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_8 as home_player, away_team_id, away_player_8 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_9 as home_player, away_team_id, away_player_9 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_10 as home_player, away_team_id, away_player_10 as away_player, date from match_info
                                    union
                                    select id, home_team_id, home_player_11 as home_player, away_team_id, away_player_11 as away_player, date from match_info
                                    ) p
                           left join player_attributes pa1 on ((pa1.player_api_id = p.home_player)  and (pa1.overall_rating <= p.date))
                           group by p.id, p.home_team_id, p.home_player, p.away_team_id, p.away_player, p.date
                           ) p
                  left join player_attributes pa2 on ((pa2.player_api_id = p.away_player) and (pa2.overall_rating <= p.date))
                  group by p.id, p.home_team_id, p.home_player, p.home_rate, p.away_team_id, p.away_player, p.date
                  ) p
        where p.home_player is not null or p.away_player is not null
        group by p.id
        order by p.id ASC
        ) p
left join match_info m on (m.id = p.id);
```

In 11.sql, I calculate the winning rate of home team from all the match, and the winning rate of the team that their average overall_rating is higher. This can show which of them has more reliability on winning the game.

```
mysql> source 11.sql
+---------------------+-----------------------+
| home_team_win_rate | strong_team_win_rate |
+---------------------+-----------------------+
|              0.4580 |                0.5066 |
+---------------------+-----------------------+
1 row in set (12.86 sec)
```

The result shows that the team with the higher overall_rating score will have more opportunity to win the game than the one with home advantage.

Task 12:

```
select  sum(case when ((home_team_score > away_team_score) and (B365H < B365D) and (B365H < B365A)) then B365H
                 when ((away_team_score > home_team_score) and (B365A < B365D) and (B365A < B365H)) then B365A
                 when ((home_team_score <= away_team_score) and (B365H < B365D) and (B365H < B365A)) then -B365H
                 when ((away_team_score <= home_team_score) and (B365A < B365D) and (B365A < B365H)) then -B365A else 0 end) as B365_betsmall_win,
        sum(case when ((home_team_score > away_team_score) and (WHH < WHD) and (WHH < WHA)) then WHH
                 when ((away_team_score > home_team_score) and (WHA < WHH) and (WHA < WHD)) then WHA
                 when ((home_team_score <= away_team_score) and (WHH < WHD) and (WHH < WHA)) then -WHH
                 when ((away_team_score <= home_team_score) and (WHA < WHH) and (WHA < WHD)) then -WHA  else 0 end) as WH_betsmall_win,
        sum(case when ((home_team_score > away_team_score) and (SJH < SJD) and (SJH < SJA)) then SJH
                 when ((away_team_score > home_team_score) and (SJA < SJD) and (SJA < SJH)) then SJA
                 when ((home_team_score <= away_team_score) and (SJH < SJD) and (SJH < SJA)) then -SJH
                 when ((away_team_score <= home_team_score) and (SJA < SJD) and (SJA < SJH)) then -SJA   else 0 end) as SJ_betsmall_win from match_info
                 where B365H is not null and B365D is not null and B365A is not null and
                       WHH is not null and WHD is not null and WHA is not null and
                       SJH is not null and SJD is not null and SJA is not null and home_team_score is not null and away_team_score is not null;
```

```sql
select  sum(case when ((home_team_score > away_team_score) and (B365H > B365D) and (B365H > B365A)) then B365H
                 when ((away_team_score > home_team_score) and (B365A > B365D) and (B365A > B365H)) then B365A
                 when ((home_team_score <= away_team_score) and (B365H > B365D) and (B365H > B365A)) then -B365H
                 when ((away_team_score <= home_team_score) and (B365A > B365D) and (B365A > B365H)) then -B365A else 0 end) as B365_betbig_win,
        sum(case when ((home_team_score > away_team_score) and (WHH > WHD) and (WHH > WHA)) then WHH
                 when ((away_team_score > home_team_score) and (WHA > WHH) and (WHA > WHD)) then WHA
                 when ((home_team_score <= away_team_score) and (WHH > WHD) and (WHH > WHA)) then -WHH
                 when ((away_team_score <= home_team_score) and (WHA > WHH) and (WHA > WHD)) then -WHA  else 0 end) as WH_betbig_win,
        sum(case when ((home_team_score > away_team_score) and (SJH > SJD) and (SJH > SJA)) then SJH
                 when ((away_team_score > home_team_score) and (SJA > SJD) and (SJA > SJH)) then SJA
                 when ((home_team_score <= away_team_score) and (SJH > SJD) and (SJH > SJA)) then -SJH
                 when ((away_team_score <= home_team_score) and (SJA > SJD) and (SJA > SJH)) then -SJA   else 0 end) as SJ_betbig_win from match_info
                 where B365H is not null and B365D is not null and B365A is not null and
                       WHH is not null and WHD is not null and WHA is not null and
                       SJH is not null and SJD is not null and SJA is not null and home_team_score is not null and away_team_score is not null;

select  sum(case when ((home_team_score = away_team_score) and (B365D > B365H) and (B365D > B365A)) then B365D
                 when ((home_team_score != away_team_score) and (B365D > B365H) and (B365D > B365A)) then -B365D else 0 end) as B365_betbigdual_win,
        sum(case when ((home_team_score = away_team_score) and (WHD > WHH) and (WHD > WHA)) then WHD
                 when ((home_team_score != away_team_score) and (WHD > WHH) and (WHD > WHA)) then -WHD else 0 end) as WH_betbigdual_win,
        sum(case when ((home_team_score = away_team_score) and (SJD > SJH) and (SJD > SJA)) then SJD
                 when ((home_team_score != away_team_score) and (SJD > SJH) and (SJD > SJA)) then -SJD else 0 end) as SJ_betbigdual_win from match_info
                  where B365H is not null and B365D is not null and B365A is not null and
                        WHH is not null and WHD is not null and WHA is not null and
                        SJH is not null and SJD is not null and SJA is not null and home_team_score is not null and away_team_score is not null;

select  sum(case when ((home_team_score = away_team_score) and (B365D < B365H) and (B365D < B365A)) then B365D
                 when ((home_team_score != away_team_score) and (B365D < B365H) and (B365D < B365A)) then -B365D else 0 end) as B365_betsmalldual_win,
        sum(case when ((home_team_score = away_team_score) and (WHD < WHH) and (WHD < WHA)) then WHD
                 when ((home_team_score != away_team_score) and (WHD < WHH) and (WHD < WHA)) then -WHD else 0 end) as WH_betsmalldual_win,
        sum(case when ((home_team_score = away_team_score) and (SJD < SJH) and (SJD < SJA)) then SJD
                 when ((home_team_score != away_team_score) and (SJD < SJH) and (SJD < SJA)) then -SJD else 0 end) as SJ_betsmalldual_win from match_info
                  where B365H is not null and B365D is not null and B365A is not null and
                        WHH is not null and WHD is not null and WHA is not null and
                        SJH is not null and SJD is not null and SJA is not null and home_team_score is not null and away_team_score is not null;
```

In 12.sql, I use 4 sql queries to illustrate how to bet. The first one is I always bet the same amount of money (here is one dollar) on the teams that have lower odds of winning the game. The second one is I always bet the same amount of money on the teams that have higher odds of winning the game. The third one is when the odds of dual is the highest, I will always bet the same amount of money on the game become dual. The fourth one is when the odds of dual is the lowest, I will always bet the same amount of money on the game become dual. All of the above I bet on three companies respectively.

```
mysql> source 12.sql
+--------------------+--------------------+--------------------+
| B365_betsmall_win | WH_betsmall_win    | SJ_betsmall_win    |
+--------------------+--------------------+--------------------+
|  557.080002784729 | 665.7000126838684 | 546.4530019760132 |
+--------------------+--------------------+--------------------+
1 row in set (0.04 sec)

+----------------------+----------------------+--------------------+
| B365_betbig_win      | WH_betbig_win        | SJ_betbig_win      |
+----------------------+----------------------+--------------------+
| -55059.559908390045 | -51505.399946928024 | -54517.08996677399 |
+----------------------+----------------------+--------------------+
1 row in set (0.02 sec)

+----------------------+--------------------+--------------------+
| B365_betbigdual_win | WH_betbigdual_win  | SJ_betbigdual_win |
+----------------------+--------------------+--------------------+
|  -5774.140011310577 | -5690.910006046295 | -6170.94003367424 |
+----------------------+--------------------+--------------------+
1 row in set (0.02 sec)

+------------------------+----------------------+----------------------+
| B365_betsmalldual_win | WH_betsmalldual_win | SJ_betsmalldual_win |
+------------------------+----------------------+----------------------+
|   0.28999996185302734 |  -5.479999899864197 |  -4.044999837875366 |
+------------------------+----------------------+----------------------+
1 row in set (0.02 sec)
```

The result shows that only first query has all positive numbers. It means that the tactic is always betting the same amount of money on the team that has lowest odds (except dual), and we'll finally earn money!

xxx_betsmall_win: bet the smallest odds win on xxx company (except dual)
xxx_betbig_win: bet the largest odds win on xxx company (except dual)
xxx_betbigdual_win: bet on dual when odds of dual is the largest on xxx company
xxx_betsmalldual_win: bet on dual when odds of dual is the smallest on xxx company