HackMD Link: https://hackmd.io/@kkmelon/rJS99pvM6

champ

```
mysql> DESCRIBE champ;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| champion_name | varchar(15) | NO   |     | NULL    |       |
| champion_id   | int         | NO   | PRI | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

match_info

```
mysql> DESCRIBE match_info;;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| match_id | int         | NO   | PRI | NULL    |       |
| duration | int         | YES  |     | NULL    |       |
| version  | varchar(15) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

participant

```
mysql> DESCRIBE participant;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| player_id   | int         | NO   | PRI | NULL    |       |
| match_id    | int         | NO   | MUL | NULL    |       |
| player      | tinyint     | YES  |     | NULL    |       |
| champion_id | int         | NO   |     | NULL    |       |
| ss1         | varchar(15) | YES  |     | NULL    |       |
| ss2         | varchar(15) | YES  |     | NULL    |       |
| position    | varchar(13) | NO   |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

- Match_id is a foreign key:

```
| participant | CREATE TABLE `participant` (
  `player_id` int NOT NULL,
  `match_id` int NOT NULL,
  `player` tinyint DEFAULT NULL,
  `champion_id` int NOT NULL,
  `ss1` varchar(15) DEFAULT NULL,
  `ss2` varchar(15) DEFAULT NULL,
  `position` varchar(13) NOT NULL,
  PRIMARY KEY (`player_id`),
  KEY `match_id` (`match_id`),
  CONSTRAINT `participant_ibfk_1` FOREIGN KEY (`match_id`) REFERENCES `match_info` (`match_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
```

teamban

```
mysql> DESCRIBE teamban;
+-------------+---------+------+-----+---------+-------+
| Field       | Type    | Null | Key | Default | Extra |
+-------------+---------+------+-----+---------+-------+
| match_id    | int     | NO   | PRI | NULL    |       |
| team        | char(1) | NO   |     | NULL    |       |
| champion_id | int     | NO   |     | NULL    |       |
| banturn     | tinyint | NO   | PRI | NULL    |       |
+-------------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

stat

```
mysql> DESCRIBE stat;
+------------------------+------------+------+-----+---------+-------+
| Field                  | Type       | Null | Key | Default | Extra |
+------------------------+------------+------+-----+---------+-------+
| player_id              | int        | NO   | PRI | NULL    |       |
| win                    | tinyint(1) | YES  |     | NULL    |       |
| item1                  | smallint   | YES  |     | NULL    |       |
| item2                  | smallint   | YES  |     | NULL    |       |
| item3                  | smallint   | YES  |     | NULL    |       |
| item4                  | smallint   | YES  |     | NULL    |       |
| item5                  | smallint   | YES  |     | NULL    |       |
| item6                  | smallint   | YES  |     | NULL    |       |
| kills                  | tinyint    | YES  |     | NULL    |       |
| deaths                 | tinyint    | YES  |     | NULL    |       |
| assists                | tinyint    | YES  |     | NULL    |       |
| longesttimespentliving | smallint   | YES  |     | NULL    |       |
| doublekills            | tinyint    | YES  |     | NULL    |       |
| triplekills            | tinyint    | YES  |     | NULL    |       |
| quadrakills            | tinyint    | YES  |     | NULL    |       |
| pentakills             | tinyint    | YES  |     | NULL    |       |
| legendarykills         | tinyint    | YES  |     | NULL    |       |
| goldearned             | mediumint  | YES  |     | NULL    |       |
| firstblood             | tinyint(1) | YES  |     | NULL    |       |
+------------------------+------------+------+-----+---------+-------+
19 rows in set (0.01 sec)
```

1. (3%) What the difference between type "char" and type "varchar"?

- Storage Size:
  - CHAR: Fixed-length character string. It always occupies the same amount of space regardless of the actual content. If the specified length is 10 characters, it will always use 10 characters of storage.
  - VARCHAR: Variable-length character string. It only uses as much storage as needed for the actual content. If the specified length is 10 characters but the actual content is only 5 characters, it will use 5 characters of storage.
  - I think `CHAR` is like `char array` and `VARCHAR` is like `string` in C++.
- Trailing Spaces:
  - CHAR: Pads the data with spaces to reach the specified length. For example, if you store 'John' in a CHAR(10) column, it will be stored as 'John ' (padded with spaces).
  - VARCHAR: Does not pad the data with spaces. It only uses the necessary amount of space for the actual content.
- Performance:
  - CHAR: Can be more efficient for fixed-length data or when the column will often be fully populated.
  - VARCHAR: More efficient for variable-length data, as it doesn't waste space on padding.
- Use `CHAR` or `VARCHAR`?
  - Use CHAR when the length of the string is relatively fixed and known in advance (e.g., postal codes, phone numbers).
  - Use VARCHAR when the length of the string can vary significantly (e.g., names, descriptions).
- Maximum Length:
  - The maximum length for CHAR is 255 characters in most databases, while VARCHAR can typically go up to 65,535 characters.

2. (3%) Type "boolean" would be stored as which type in MySQL?

- Tinyint(1).

3. (4%) How many bytes it should take for "tinyint", "smallint", "mediumint", "int"? (e.g. 8 bytes for "bigint") And what's the range they can express? (e.g. from -1000 to 1000)

Type	Bytes	Range
TINYINT	1	$-2^7 \sim 2^7 - 1$
SMALLINT	2	$-2^{15} \sim 2^{15} - 1$
MEDIUMINT	3	$-2^{23} \sim 2^{23} - 1$
INT	4	$-2^{31} \sim 2^{31} - 1$
BIGINT	8	$-2^{63} \sim 2^{63} - 1$

4. (5%) What do you think about this table schema? If you can change this table architecture, how would you modify it and why?

- My opinions:
  - `longesttimespentliving` is unnecessary.
  - Add a table called `player` to store additional information about the account(e.g. created time, games played, ranking, friend lists...)
  - Move the `participant` into `stat`, because I think skills are tied to some attributes in `stat`
  - Merge `match_info` and `teamban` because they are associated. If there are no bans required, just set the columns empty.
  - I noticed that `item` are stored in `tinyint` maybe they can be used as foreign key to another table, but it is not necessary in this HW.

Data Count

```
mysql> SOURCE list_table_rows.sql
+-----------+
| champions |
+-----------+
|       138 |
+-----------+
1 row in set (0.00 sec)

+---------+
| matches |
+---------+
|  182527 |
+---------+
1 row in set (0.03 sec)

+--------------+
| participants |
+--------------+
|      1825270 |
+--------------+
1 row in set (0.27 sec)
```

```
+---------+
| bans    |
+---------+
| 1089969 |
+---------+
1 row in set (0.14 sec)

+---------+
| stats   |
+---------+
| 1825270 |
+---------+
1 row in set (0.22 sec)
```

Problem 1.

Result

```
mysql> SOURCE 1.sql
+-----+
| cnt |
+-----+
| 138 |
+-----+
1 row in set (0.01 sec)
```

Query

```sql
SELECT COUNT(DISTINCT champion_id)
AS cnt
FROM champ;
```

Problem 2.

Result

```
    -> ^C
mysql> source 2.sql;
+-----+
| cnt |
+-----+
|  74 |
+-----+
1 row in set (0.21 sec)
```

Query

```sql
SELECT COUNT(DISTINCT SUBSTRING_INDEX(version, '.', 2))
AS cnt
FROM match_info;
```

Problem 3.

Result:

```
mysql> source 3.sql;
+---------------+-------+
| champion_name | cnt   |
+---------------+-------+
| Lee Sin       | 56598 |
| Master Yi     | 23385 |
| Graves        | 19767 |
+---------------+-------+
3 rows in set (1.98 sec)
```

Query

```sql
SELECT champion_name, COUNT(*) as cnt
FROM champ c
JOIN participant p ON c.champion_id = p.champion_id
WHERE p.position = 'JUNGLE'
GROUP BY champion_name
ORDER BY cnt DESC
LIMIT 3;
```

Problem 4.

Result:

```
mysql> source 4.sql;
+----------+----------+
| match_id | time     |
+----------+----------+
|   146486 | 01:23:11 |
|    69303 | 01:20:14 |
|      581 | 01:16:59 |
|    70361 | 01:15:06 |
|   176628 | 01:13:34 |
+----------+----------+
5 rows in set (0.07 sec)
```

Query:

```sql
SELECT match_id,
       SEC_TO_TIME(duration) AS time
FROM match_info
ORDER BY duration DESC
LIMIT 5;
```

Problem 5.

Result:

```
mysql> source 5.sql
+----------+-----+
| win_lose | cnt |
+----------+-----+
| lose     | 338 |
| win      | 807 |
+----------+-----+
2 rows in set (18.67 sec)
```

Query:

```sql
SELECT
    CASE
        WHEN matches_stats.win = 1 THEN 'win'
        ELSE 'lose'
    END AS win_lose,
    COUNT(*) AS cnt
FROM (
    SELECT
        p.match_id,
        s.win,
        AVG(s.longesttimespentliving) AS avg_longest_time
    FROM participant p
    JOIN stat s ON p.player_id = s.player_id
    GROUP BY p.match_id, s.win
    HAVING AVG(s.longesttimespentliving) >= 1200
) AS matches_stats
GROUP BY win_lose
ORDER BY cnt;
```

Problem 6.

Result:

```
mysql> source 6.sql;
+--------------+----------------+
| position     | champion_name  |
+--------------+----------------+
| DUO_CARRY    | Caitlyn        |
| DUO_SUPPORT  | Thresh         |
| JUNGLE       | Lee Sin        |
| MID          | Ahri           |
| TOP          | Riven          |
+--------------+----------------+
5 rows in set (8.53 sec)
```

Query:

```sql
WITH champion_counts AS (
    SELECT
        p.position,
        c.champion_name,
        RANK() OVER (PARTITION BY p.position ORDER BY COUNT(*) DESC) AS rk
    FROM champ c, participant p, match_info m
    WHERE c.champion_id = p.champion_id
    AND p.match_id = m.match_id
    AND m.duration BETWEEN 2400 AND 3000
    AND p.position IN ('TOP', 'JUNGLE', 'MID', 'DUO_CARRY', 'DUO_SUPPORT')
    GROUP BY p.position, c.champion_name
)

SELECT position, champion_name
FROM champion_counts
WHERE rk = 1;
```

Problem 7.

Result:

```
mysql> source 7.sql;
+-------------+---------------+---------+
| position    | champion_name | kda     |
+-------------+---------------+---------+
| DUO_CARRY   | Shaco         | 19.0000 |
| DUO_SUPPORT | Janna         |  3.8330 |
| JUNGLE      | Ivern         |  3.8764 |
| MID         | Ivern         |  3.7015 |
| TOP         | Sona          |  3.1538 |
+-------------+---------------+---------+
5 rows in set (31.40 sec)
```

```sql
Run on active connection | Select block
WITH champion_stats AS(
    SELECT p.position,
        c.champion_name,
        ((SUM(s.kills) + SUM(s.assists)) / SUM(s.deaths)) AS kda,
        RANK() OVER (PARTITION BY p.position ORDER BY ((SUM(s.kills) + SUM(s.assists)) / SUM(s.deaths)) DESC) AS rk
    FROM champ c,
        participant p,
        stat s
    WHERE c.champion_id = p.champion_id
        AND p.player_id = s.player_id
        AND p.position IN (
            'TOP',
            'JUNGLE',
            'MID',
            'DUO_CARRY',
            'DUO_SUPPORT'
        )
    GROUP BY p.position,
        c.champion_name
    HAVING SUM(s.deaths) > 0
)

SELECT position, champion_name, kda
FROM champion_stats
WHERE rk = 1;
```

Problem 8.

Result:

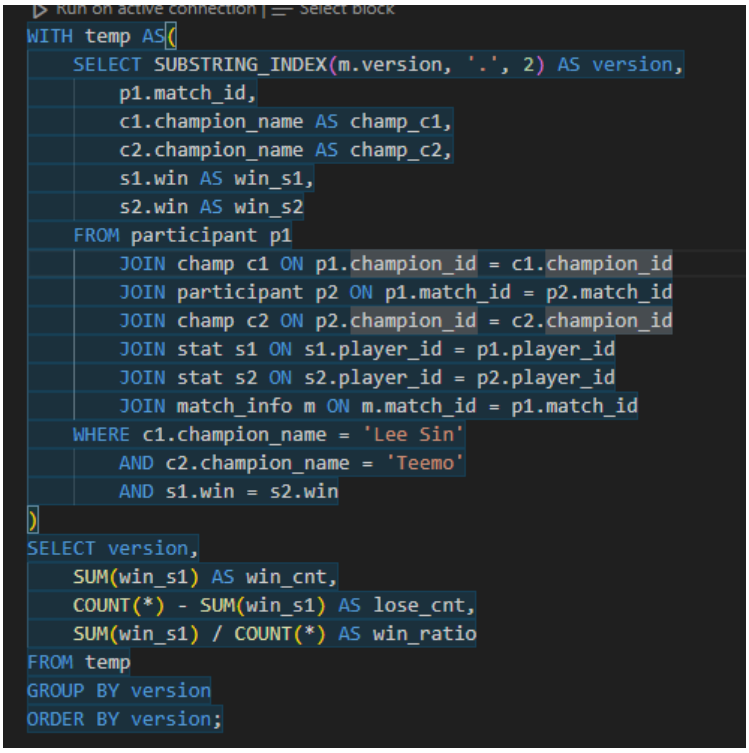
```
mysql> source 8.sql
+---------------+
| champion_name |
+---------------+
| Kayn          |
| Ornn          |
| Rakan         |
| RekSai        |
| Sion          |
| Xayah         |
+---------------+
6 rows in set (0.35 sec)
```

Query:

```sql
SELECT champion_name
FROM champ c
WHERE champion_id NOT IN (
    SELECT DISTINCT c.champion_id
    FROM champ c, teamban tb, match_info m
    WHERE tb.match_id = m.match_id
    AND tb.champion_id = c.champion_id
    AND SUBSTRING_INDEX(m.version, '.', 2) = '7.7'
)
ORDER BY champion_name;
```

Problem 9.

version	win_cnt	lose_cnt	win_ratio
4.10	2	1	0.6667
4.12	0	1	0.0000
4.15	1	1	0.5000
4.17	0	1	0.0000
4.18	0	1	0.0000
4.19	0	1	0.0000
4.21	1	1	0.5000
4.9	1	0	1.0000
5.1	1	2	0.3333
5.12	1	0	1.0000
5.13	0	1	0.0000
5.15	0	1	0.0000
5.19	1	0	1.0000
5.20	2	0	1.0000
5.21	0	2	0.0000
5.24	1	1	0.5000
5.5	1	0	1.0000
5.6	0	1	0.0000
5.7	1	0	1.0000
6.1	0	1	0.0000
6.13	1	0	1.0000
6.14	1	0	1.0000
6.18	1	1	0.5000
6.19	1	0	1.0000
6.2	1	1	0.5000
6.20	3	2	0.6000
6.21	0	2	0.0000
6.22	2	1	0.6667
6.23	3	2	0.6000
6.24	4	3	0.5714
6.5	1	0	1.0000
6.6	0	1	0.0000
6.8	1	0	1.0000
6.9	1	1	0.5000
7.10	282	304	0.4812
7.2	2	1	0.6667
7.3	0	1	0.0000
7.4	1	1	0.5000
7.5	2	2	0.5000
7.6	2	5	0.2857
7.7	32	29	0.5246
7.8	210	237	0.4698
7.9	527	464	0.5318



```sql
WITH temp AS(
    SELECT SUBSTRING_INDEX(m.version, '.', 2) AS version,
        p1.match_id,
        c1.champion_name AS champ_c1,
        c2.champion_name AS champ_c2,
        s1.win AS win_s1,
        s2.win AS win_s2
    FROM participant p1
        JOIN champ c1 ON p1.champion_id = c1.champion_id
        JOIN participant p2 ON p1.match_id = p2.match_id
        JOIN champ c2 ON p2.champion_id = c2.champion_id
        JOIN stat s1 ON s1.player_id = p1.player_id
        JOIN stat s2 ON s2.player_id = p2.player_id
        JOIN match_info m ON m.match_id = p1.match_id
    WHERE c1.champion_name = 'Lee Sin'
        AND c2.champion_name = 'Teemo'
        AND s1.win = s2.win
)
SELECT version,
    SUM(win_s1) AS win_cnt,
    COUNT(*) - SUM(win_s1) AS lose_cnt,
    SUM(win_s1) / COUNT(*) AS win_ratio
FROM temp
GROUP BY version
ORDER BY version;
```

Problem 10.

```
mysql> source 10.sql
+-----------------+-----------+----------+---------------+------------------+-----------+---------------+---------------+
| self_champ_name | win_ratio | self_kda | self_avg_gold | enemy_champ_name | enemy_kda | enemy_avg_gold | battle_record |
+-----------------+-----------+----------+---------------+------------------+-----------+---------------+---------------+
| Yasuo           |    0.6042 |   1.7255 |    12501.5833 | Gragas           |    2.4916 |     10755.1042 |           288 |
| Darius          |    0.5881 |   2.1526 |    12017.6165 | Gragas           |    2.2283 |     10681.5057 |           352 |
| Jax             |    0.5825 |   1.7231 |    12132.2330 | Gragas           |    2.4101 |     10994.5340 |           206 |
| Teemo           |    0.5820 |   1.8196 |    12376.0899 | Gragas           |    2.4378 |     11256.6878 |           189 |
| Nasus           |    0.5528 |   2.0772 |    11905.5366 | Gragas           |    2.3986 |     10985.0000 |           123 |
| Pantheon        |    0.5515 |   2.2233 |    11616.3015 | Gragas           |    2.1137 |     10785.1176 |           136 |
| Illaoi          |    0.5481 |   1.8441 |    12178.4904 | Gragas           |    2.2791 |     10610.6827 |           104 |
| Kled            |    0.5478 |   2.6554 |    11211.7070 | Gragas           |    2.3155 |     10211.3439 |           157 |
| Galio           |    0.5421 |   2.4855 |    10766.2421 | Gragas           |    2.5829 |     10377.9737 |           190 |
| Gangplank       |    0.5412 |   2.4881 |    13863.7599 | Gragas           |    2.5465 |     11519.4337 |           279 |
| Rumble          |    0.5391 |   2.0767 |    11800.5703 | Gragas           |    2.7543 |     10990.9844 |           128 |
| Gnar            |    0.5333 |   2.0596 |    10513.8095 | Gragas           |    2.2646 |     10027.0286 |           105 |
| Shen            |    0.5333 |   3.4077 |    11040.8722 | Gragas           |    2.3261 |     10910.1444 |           180 |
| Tryndamere      |    0.5303 |   1.6813 |    12052.4343 | Gragas           |    2.5882 |     10333.8333 |           198 |
| Garen           |    0.5294 |   2.3300 |    11935.0588 | Gragas           |    2.2147 |     10763.0588 |           102 |
| Riven           |    0.5291 |   1.9934 |    11894.6620 | Gragas           |    2.3340 |     10782.8648 |           429 |
| Renekton        |    0.5251 |   1.9700 |    11890.7916 | Gragas           |    2.4054 |     10861.9789 |           379 |
| Irelia          |    0.5243 |   2.0192 |    11796.8932 | Gragas           |    2.5975 |     10778.9223 |           206 |
| Fiora           |    0.5150 |   1.8389 |    11766.3375 | Gragas           |    2.2258 |     10495.7400 |           400 |
| Fizz            |    0.4965 |   1.9594 |    11573.7021 | Gragas           |    2.2930 |     10840.8582 |           141 |
| Malphite        |    0.4825 |   2.6718 |    10515.2797 | Gragas           |    2.3934 |     10776.5105 |           143 |
| Kennen          |    0.4653 |   1.6964 |    11550.3889 | Gragas           |    2.8311 |     11029.6250 |           144 |
| Nautilus        |    0.4493 |   2.5975 |    10402.3333 | Gragas           |    2.6741 |     10887.7826 |           138 |
| Jayce           |    0.4416 |   2.0161 |    11570.1364 | Gragas           |    2.7776 |     11072.3442 |           154 |
+-----------------+-----------+----------+---------------+------------------+-----------+---------------+---------------+
24 rows in set (3.86 sec)
```

Run on active connection | Select block

```sql
SELECT c1.champion_name AS self_champ_name,
    SUM(s1.win) / COUNT(*) AS win_ratio,
    (SUM(s1.kills) + SUM(s1.assists)) / SUM(s1.deaths) AS self_kda,
    AVG(s1.goldearned) AS self_avg_gold,
    c2.champion_name AS enemy_champ_name,
    (SUM(s2.kills) + SUM(s2.assists)) / SUM(s2.deaths) AS enemy_kda,
    AVG(s2.goldearned) AS enemy_avg_gold,
    COUNT(*) AS battle_record
FROM participant p1
    JOIN champ c1 ON p1.champion_id = c1.champion_id
    JOIN participant p2 ON p1.match_id = p2.match_id AND p1.player_id != p2.player_id
    JOIN champ c2 ON p2.champion_id = c2.champion_id
    JOIN stat s1 ON s1.player_id = p1.player_id
    JOIN stat s2 ON s2.player_id = p2.player_id
WHERE c2.champion_name = 'Gragas'
    AND p1.position = 'TOP'
    AND p1.position = p2.position
GROUP BY c1.champion_name, c2.champion_name
HAVING COUNT(*) > 100
ORDER BY win_ratio DESC;
```

Problem 11.

Idea

- First I put the summoner's skill in order, so that SQL can consider (Skill A, Skill B) and (Skill B, Skill A) the same.
- Second, group the data by the skills pair, and calculate the win ratio where the skill combination is selected for more than 1000 times.

Result 1:

```
+-----------+-----------+-----------+----------+---------------+
| skill_one | skill_two | win_ratio | self_kda | battle_record |
+-----------+-----------+-----------+----------+---------------+
| Haste     | Ignite    |    0.5238 |   1.8590 |          1594 |
| Flash     | Ignite    |    0.5221 |   2.0738 |         39665 |
| Flash     | Haste     |    0.5116 |   2.1121 |          2418 |
| Ignite    | Teleport  |    0.5016 |   2.0619 |          5668 |
| Flash     | Teleport  |    0.5006 |   2.0655 |        278156 |
| Haste     | Teleport  |    0.4940 |   1.8317 |          7261 |
| Exhaust   | Flash     |    0.4877 |   2.1054 |          6623 |
| Flash     | Heal      |    0.4580 |   1.8133 |          1214 |
+-----------+-----------+-----------+----------+---------------+
8 rows in set (17.71 sec)
```

Query 1:

```sql
SELECT CASE
        WHEN p.ss1 < p.ss2 THEN p.ss1
        ELSE p.ss2
    END AS skill_one,
    CASE
        WHEN p.ss1 < p.ss2 THEN p.ss2
        ELSE p.ss1
    END AS skill_two,
    SUM(s.win) / COUNT(*) AS win_ratio,
    (SUM(s.kills) + SUM(s.assists)) / SUM(s.deaths) AS self_kda,
    COUNT(*) AS battle_record
FROM participant p
    JOIN champ c ON p.champion_id = c.champion_id
    JOIN stat s ON s.player_id = p.player_id
WHERE p.position = 'TOP'
    AND p.champion_id IN (
        SELECT DISTINCT c.champion_id
        FROM stat s
            JOIN participant p ON s.player_id = p.player_id
            JOIN champ c ON p.champion_id = c.champion_id
        WHERE p.position = 'TOP'
        GROUP BY c.champion_id
        HAVING COUNT(*) > 1000
    )
GROUP BY skill_one,
    skill_two
HAVING COUNT(*) > 1000
ORDER BY (SUM(s.win) / COUNT(*)) DESC;
```

Problem

- I think the combination are related to the champion picked, so I wrote another Query for those who want to pick the best skill combination according to the champion.

Result 2:

self_champ_name	skill_one	skill_two	win_ratio	self_kda	battle_record
Pantheon	Flash	Ignite	0.5473	2.3701	4917
Darius	Flash	Haste	0.5431	2.2577	1230
Wukong	Flash	Ignite	0.5420	2.4352	1035
Teemo	Flash	Ignite	0.5363	1.8852	4671
Yasuo	Flash	Ignite	0.5362	1.7031	1671
Tryndamere	Flash	Ignite	0.5288	1.9301	3232
Yasuo	Flash	Teleport	0.5276	1.6233	11965
Aatrox	Flash	Teleport	0.5265	2.0038	2602
Kayle	Flash	Teleport	0.5256	1.8722	3225
Warwick	Flash	Teleport	0.5243	2.0783	1194
Renekton	Flash	Ignite	0.5208	2.1864	3487
Darius	Flash	Ignite	0.5198	2.0224	1741
Swain	Flash	Teleport	0.5190	2.2831	3946
Tryndamere	Flash	Teleport	0.5160	1.7935	7029
Illaoi	Flash	Teleport	0.5152	1.7203	6046
Teemo	Flash	Teleport	0.5148	1.7495	6185
Wukong	Flash	Teleport	0.5136	2.2517	2691
Irelia	Flash	Teleport	0.5135	2.1121	9839
Riven	Flash	Teleport	0.5132	2.0154	17038
Darius	Flash	Teleport	0.5128	1.9826	14642
Riven	Flash	Ignite	0.5128	2.0056	2147
Akali	Flash	Ignite	0.5126	2.0735	1432
Garen	Flash	Ignite	0.5108	2.2148	1386
Fizz	Ignite	Teleport	0.5107	2.1788	3362
Quinn	Flash	Ignite	0.5095	2.0673	1058
Akali	Flash	Teleport	0.5093	1.9652	2623
Malphite	Flash	Teleport	0.5081	2.7510	7109
Yorick	Flash	Teleport	0.5074	1.9614	3877
Fiora	Flash	Teleport	0.5073	1.9043	17354

Code for the second query in problem 11.

```sql
-- Calculate how many times each summoner skills were picked in the top lane when the champion was in the previous query.
SELECT c.champion_name AS self_champ_name,
    CASE
        WHEN p.ss1 < p.ss2 THEN p.ss1
        ELSE p.ss2
    END AS skill_one,
    CASE
        WHEN p.ss1 < p.ss2 THEN p.ss2
        ELSE p.ss1
    END AS skill_two,
    SUM(s.win) / COUNT(*) AS win_ratio,
    (SUM(s.kills) + SUM(s.assists)) / SUM(s.deaths) AS self_kda,
    COUNT(*) AS battle_record
FROM participant p
    JOIN champ c ON p.champion_id = c.champion_id
    JOIN stat s ON s.player_id = p.player_id
WHERE p.position = 'TOP'
    AND p.champion_id IN (
        SELECT DISTINCT c.champion_id
        FROM stat s
            JOIN participant p ON s.player_id = p.player_id
            JOIN champ c ON p.champion_id = c.champion_id
        WHERE p.position = 'TOP'
        GROUP BY c.champion_id
        HAVING COUNT(*) > 1000
    )
GROUP BY self_champ_name,
    skill_one,
    skill_two
HAVING COUNT(*) > 1000
ORDER BY (SUM(s.win) / COUNT(*)) DESC
LIMIT 30;
```

Problem 12.

Goals:

- To provide player win rate of champions in each position so that player could choose those "OP" champions with high win rate when they don't know what to play.

Query:

- This query follows the structure of Problem 7. Only the third column is changed to `win_ratio` and it provides 5 data for each position.
- `win_ratio` is calculated by the formula in Problem 9. Only champions that were picked for more than 300 times in each position are considered(Or there will be some champions with `win_ratio` 1.

```
Run on active connection | Select block
WITH champion_stats AS(
    SELECT p.position,
        c.champion_name,
        SUM(s.win) / COUNT(*) AS win_ratio,
        RANK() OVER (PARTITION BY p.position ORDER BY (SUM(s.win) / COUNT(*)) DESC) AS rk
    FROM champ c,
        participant p,
        stat s
    WHERE c.champion_id = p.champion_id
        AND p.player_id = s.player_id
        AND p.position IN (
            'TOP',
            'JUNGLE',
            'MID',
            'DUO_CARRY',
            'DUO_SUPPORT'
        )
    GROUP BY p.position,
        c.champion_name
    HAVING COUNT(*) > 300
)

SELECT position, champion_name, win_ratio
FROM champion_stats
WHERE rk <= 5;
```

Result:

```
mysql> source 12.sql;
+-------------+---------------+-----------+
| position    | champion_name | win_ratio |
+-------------+---------------+-----------+
| DUO_CARRY   | Ziggs         |    0.5392 |
| DUO_CARRY   | Twitch        |    0.5352 |
| DUO_CARRY   | KogMaw        |    0.5304 |
| DUO_CARRY   | Draven        |    0.5294 |
| DUO_CARRY   | Miss Fortune  |    0.5264 |
| DUO_SUPPORT | Sion          |    0.5398 |
| DUO_SUPPORT | Sona          |    0.5360 |
| DUO_SUPPORT | Janna         |    0.5325 |
| DUO_SUPPORT | Blitzcrank    |    0.5270 |
| DUO_SUPPORT | Leona         |    0.5231 |
| JUNGLE      | Ivern         |    0.5638 |
| JUNGLE      | Nunu          |    0.5339 |
| JUNGLE      | Skarner       |    0.5315 |
| JUNGLE      | Xin Zhao      |    0.5289 |
| JUNGLE      | Amumu         |    0.5275 |
| MID         | Pantheon      |    0.5544 |
| MID         | Zilean        |    0.5485 |
| MID         | Anivia        |    0.5415 |
| MID         | Xerath        |    0.5381 |
| MID         | Annie         |    0.5341 |
| TOP         | Karthus       |    0.5343 |
| TOP         | Annie         |    0.5340 |
| TOP         | Pantheon      |    0.5307 |
| TOP         | Kayle         |    0.5270 |
| TOP         | Yasuo         |    0.5268 |
+-------------+---------------+-----------+
25 rows in set (28.73 sec)
```