Question 1. [14 MARKS]

Given R and S are relations. Which of the following statements are true? Circle one answer for each and give a brief justification of your reasoning.

Part (a) [4 MARKS]

If R and S have no attributes in common, $R \times S = R \bowtie S$.

TRUE **FALSE**

True.

Part (b) [4 MARKS]

If R and S have at least one attribute in common, it cannot be true that $R \times S = R \bowtie S$.

TRUE **FALSE**

True because the schemas of the two relations are necessarily different, $R \times S$ includes each common attribute twice and $R \bowtie S$ contains the common attribute once. Ie, $R \times S$ has 1 more column than $R \bowtie S$ even if the tuples are the same.

Part (c) [4 MARKS]

Let L be a list of attributes of R and c be a boolean expression involving attributes from R. It is true that $\Pi_L(\sigma_c R) = \sigma_c(\Pi_L R)$.

TRUE **FALSE**

FALSE. σ might prune some tuples off based on columns that no longer exist if Π applied first.

Part (d) [2 MARKS]

In MySQL, it is preferable to perform a join with only `where`,

$$R \texttt{ inner join } S \texttt{ where } \ldots$$

than `on` and `where`:

$$R \texttt{ inner join } S \texttt{ on } \ldots \texttt{ where } \ldots$$

because we can put both the `on` and `where` constraints all into one `where` condition simplifying the query.

TRUE **FALSE**

False. prefer to do `on` then `where`. More efficient (main reason) but also more reflective of the meaning of the join.

Question 2. [14 MARKS]

Consider the following database consisting of two relations A and B:

A

Q	R	S
1	5	0
4	6	5
2	8	0
3	4	0
1	2	3
3	3	2

B

S	T
0	5
2	6
4	5

Part (a) [4 MARKS]

What does the following query return. Represent the result as a table including the schema (table headings).

$$A \bowtie B$$

Q	R	S	T
1	5	0	5
2	8	0	5
3	4	0	5
3	3	2	6

Part (b) [4 MARKS]

What does the following query return. Represent the result as a table including the schema (table headings).

$$\Pi_{A.Q,A.R,B.T}(A \bowtie_{A.Q=B.S} B)$$

Q	R	B.T
4	6	5
2	8	6

Part (c) [4 MARKS]

What does the following query return. Represent the result as a table including the schema (table headings).

$$(\Pi_S A - \Pi_S B) \cap (\Pi_S A - \Pi_S (A \bowtie (\sigma_{T=5} B)))$$

S
5
3

Part (d) [2 MARKS]

Give a MySQL query that retrieves the same information as part (c). *Note: this looks harder than it is - think about what the query above is returning.*

```
select A.S from A where A.S not in (select S from B where T = 5)
```

Question 3. [16 MARKS]

Consider the following database.

R:

A	B
1	2
3	4
1	3

S:

B	C
1	3
2	4

T:

B	C
2	5
2	3
2	4

Give the results (schema and data) produced by each of the following queries.

Part (a) [4 MARKS]

```
select R1.A from R R1 inner join R R2 where R1.B < R2.B;
```

Solution.

A
1
1
1

Part (b) [4 MARKS]

```
select * from R inner join S where C in (select C from S natural join T);
```

Solution.

R.A	R.B	S.B	S.C
1	2	2	4
3	4	2	4
1	3	2	4

Part (c) [4 MARKS]

```
select * from S full outer join T on S.B = T.B;
```

Solution.

S.B	S.C	T.B	T.C
1	3	null	null
2	4	2	5
2	4	2	3
2	4	2	4

Part (d) [4 MARKS]

```
select * from (R natural join S) natural join T;
```

Solution.

A	B	C
1	2	4

Question 4. [25 MARKS]

Consider the following baseball database. You may assume the following facts about the database:

- Players may have stats from many different years and may play for different teams in different years (or even the same year).
- You may assume id's are unique and that each players full name (first and last together) is unique.

Your queries below will be graded on correctness not on efficiency.

player(id, first_name, last_name)
team(team_id, city, name)
plays_for(id, team_id, position, year)
bat_stats(id, HR, AVG, year)

Part (a) [2 MARKS]

Underline a reasonable primary key(s) for each relation.

Solution.
player(id, first_name, last_name)
team(team_id, city, name)
plays_for(id, team_id, position, year) or depending if they assume multiple positions on one team **plays_for**(id, team_id, position, year)
bat_stats(id, HR, AVG, year)

Part (b) [2 MARKS]

List the foreign key relations.

Solutions.
plays_for(id) foreign key for player.
plays_for(team_id) foreign key for team.
bat_stats(id) foreign key for player.

Part (c) [3 MARKS]

Write a query that returns all players first and last names who played position 'pitcher' in 2015.

```
select first_name, last_name from player natural join plays_for where
position = 'pitcher' and year = '2015';
```

Part (d) [3 MARKS]

Create a view called '**Jays**' that contains all players first and last name and id who played for the team with name 'Blue Jays' in the year 2016.

```
create view Jays as (select first_name, last_name, id from player natural join
plays_for natural join team where name = 'Blue Jays' and year = '2016');
```

Question 4. (CONTINUED)

The database schema for your reference:

player(id, first_name, last_name)
team(team_id, city, name)
plays_for(id, team_id, position, year)
bat_stats(id, HR, AVG, year)
view: Jays(id, first_name, last_name)

Part (e) [3 MARKS]

Use the view **Jays** from part (d) (the schema is listed above - you may use it even if you were unable to complete part(d)) to write a query that returns the total number of home runs (HR) combined of all players playing for the 'Blue Jays' in 2016.

```
select sum(HR) from Jays natural join bat_stats where year = 2016;
```

Part (f) [3 MARKS]

Write a query that returns the first and last names of each player and the number of years that they played. The title of the column containing the number of years should have title `career_length`.

```
select first_name, last_name, count(year) as 'career_length' from player
natural join plays_for group by id;
```

Part (g) [3 MARKS]

Write a query that returns the first and last names of each player and the number of years that they played for all players whose average (over all years) of AVG was greater than 0.275. The title of the column containing the number of years should have title `career_length`.

```
select first_name, last_name, count(year) as 'career_length' from
player natural join plays_for group by id having avg(AVG) > 0.275;
```

Question 4. (CONTINUED)

The database schema for your reference:

player(id, first_name, last_name)
team(team_id, city, name)
plays_for(id, team_id, position, year)
bat_stats(id, HR, AVG, year)

Part (h) [3 MARKS]

Suppose that the player 'Edwin Encarnacion' gets traded to the team with team_id = 22 in 2017. Write an `insert` statement to add him with his new team to the **plays_for** table - you may assume that his position remains the same from 2016.

```
insert into plays\_for values
(select player.id, 22, plays\_for.position, 2017 from
player inner join plays_for on player.id = plays_for.id
where first_name = 'Edwin' and last_name = 'Encarnacion' and year = 2016);
```

Part (i) [3 MARKS]

Suppose we are in the 2017 season and 'Jose Batista' hits another home run. Update his HR statistic in the **bat_stats** relation.

```
update bat_stats set HR = HR + 1 where year = '2017' and id in
(select id from player where first_name = 'Jose' and last_name = 'Batista')
```

Question 5. [1 MARK]

Make sure your student number is on every page and that you have filled in your lab day and time.