Mini Project – DBMS1 and DBMS2

Part A

ICC Test Cricket

Data Set	
Data Dictionary	
Tasks Performed with Answers	

Dataset: ICC Test Batting Figures.csv

Test cricket is the form of the sport of cricket with the longest match duration and is considered the game's highest standard. Test matches are played between national representative teams that have been granted ‘Test status’, as determined and conferred by the International Cricket Council (ICC). The term Test stems from the fact that the long, gruelling matches are mentally and physically testing. Two teams of 11 players each play a four-innings match, which may last up to five days (or longer in some historical cases). It is generally considered the most complete examination of a team's endurance and ability.
The Data consists of runs scored by the batsmen from 1877 to 2019 December.

Data Dictionary:

Sl No	Column Name	Column Description
1	Player	Name of the player and country the player belongs to
2	Span	The duration of years between which the player was active
3	Mat	No of matches played by the player
4	Inn	No of innings played by the player
5	NO	No of matches the player was NOT OUT by the end of the match.
6	Runs	Total number of runs scored by the player
7	HS	Highest Score of the player
8	Avg	Average runs scored by the player in all the matches
9	100	No of centuries scored by the player
10	50	No of fifties scored by the player
11	0	No of Duck outs of the player
12	Player Profile	Link to the profiles of the players

Tasks Performed with Answers:

1. Import the csv file to a table in the database.
2. Remove the column 'Player Profile' from the table.

```
ALTER TABLE `icc_test_batting_figures`  CHANGE COLUMN `Player
Profile` `PlayerProfile` TEXT NULL DEFAULT NULL ;
    ALTER TABLE icc_test_batting_figures DROP COLUMN PlayerProfile;
```

3. Extract the country name and player names from the given data and store it in separate columns for further usage

```
ALTER TABLE icc_test_batting_figures ADD COLUMN Country TEXT ;
update  icc_test_batting_figures set country =
replace(SUBSTRING(player,POSITION('(' IN player)+1,
LENGTH(player)),')',"");


ALTER TABLE icc_test_batting_figures ADD COLUMN player_name TEXT;
update  icc_test_batting_figures set player_name =
SUBSTR(player,1,POSITION('(' IN player)-1);
```

4. From the column 'Span' extract the start_year and end_year and store them in separate columns for further usage.

```
    ALTER TABLE icc_test_batting_figures ADD COLUMN start_year TEXT;
     update  icc_test_batting_figures set start_year = SUBSTR(span,
1,4 ) ;

       ALTER TABLE icc_test_batting_figures ADD COLUMN end_year TEXT;
       update  icc_test_batting_figures set end_year = SUBSTR(span,
       6,4 ) ;
```

5. The column 'HS' has the highest score scored by the player so far in any given match. The column also has details if the player had completed the match in a NOT OUT status. Extract the data and store the highest runs and the NOT OUT status in different columns.

```
ALTER TABLE icc_test_batting_figures ADD COLUMN HS_notOut TEXT;
update  icc_test_batting_figures set HS_notOut =
SUBSTR(HS,1,length(HS)-1) where SUBSTR(HS,length(HS),length(HS))='*' ;
```

6. Using the data given, considering the players who were active in the year of 2019, create a set of batting order of best 6 players using the selection criteria of those who have a good average score across all matches for India.

```
SELECT *
 FROM `batting figures`
 WHERE player LIKE '%(INDIA)' AND
```

```
        span LIKE '%2019'
ORDER BY avg DESC
LIMIT 6
```

7. Using the data given, considering the players who were active in the year of 2019, create a set of batting order of best 6 players using the selection criteria of those who have the highest number of 100s across all matches for India.

```
SELECT *
 FROM `batting figures`
 WHERE player LIKE '%(INDIA)' AND
       span LIKE '%2019'
ORDER BY '100' DESC
LIMIT 6
```

8. Using the data given, considering the players who were active in the year of 2019, create a set of batting order of best 6 players using 2 selection criteria of your own for India.

```
SELECT *
 FROM `batting figures`
 WHERE player LIKE '%(INDIA)' AND
       span LIKE '%2019' AND hs > 150 AND avg > 40
ORDER BY '100' DESC
LIMIT 6
```

9. Create a View named 'Batting_Order_GoodAvgScorers_SA' using the data given, considering the players who were active in the year of 2019, create a set of batting order of best 6 players using the selection criteria of those who have a good average score across all matches for South Africa.

```
  CREATE VIEW Batting_Order_GoodAvgScorers_SA_VW AS
SELECT *
 FROM `batting figures`
 WHERE player LIKE '%(SA)' AND
        span LIKE '%2019'
ORDER BY avg DESC
LIMIT 6
```

10. Create a View named 'Batting_Order_HighestCenturyScorers_SA' Using the data given, considering the players who were active in the year of 2019, create a set of batting order of best 6 players using the selection criteria of those who have highest number of 100s across all matches for South Africa.

```
CREATE VIEW Batting_Order_HighestCenturyScorers_SA_VW AS
SELECT *
 FROM `batting figures`
 WHERE player LIKE '%(SA)' AND
       span LIKE '%2019'
```

```
ORDER BY '100' DESC
LIMIT 6
```

11. Using the data given, Give the number of player_played for each country.

```
SELECT SUBSTRING(player,POSITION('(' IN player), LENGTH(player)) AS
country_name, COUNT(player) AS Number_of_player
FROM `batting figures`
GROUP BY SUBSTRING(player,POSITION('(' IN player), LENGTH(player))
```

12. Using the data given, Give the number of player_played for Asian and Non-Asian continent

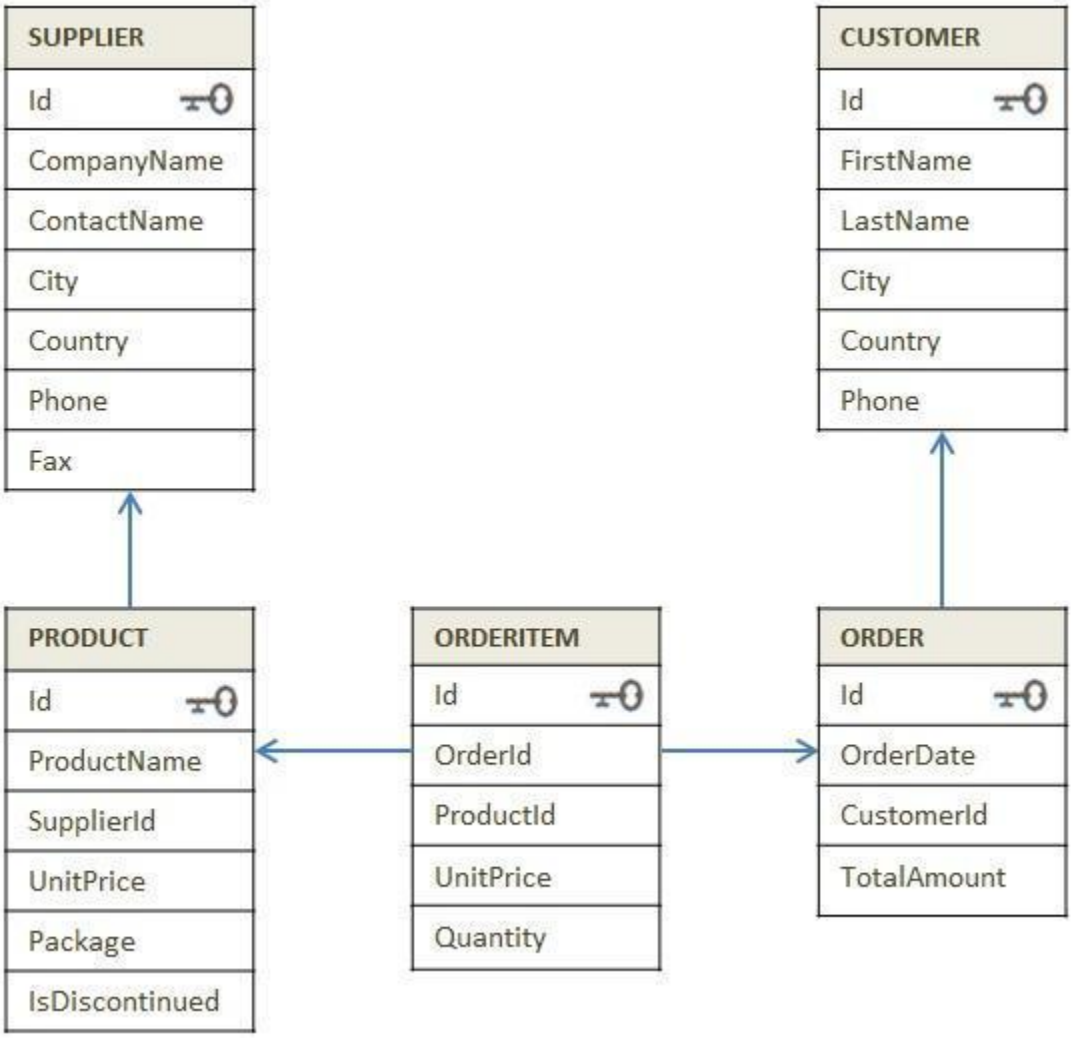
```
SELECT COUNT(player) AS no_of_asian_cricketer
FROM `batting figures`
WHERE SUBSTRING(player,POSITION('(' IN player), LENGTH(player))
LIKE '%INDIA%' OR
      SUBSTRING(player,POSITION('(' IN player), LENGTH(player))
LIKE '%PAK%' OR
       SUBSTRING(player,POSITION('(' IN player),
LENGTH(player)) LIKE '%BDESH%' OR
       SUBSTRING(player,POSITION('(' IN player),
LENGTH(player)) LIKE '%SL%' OR
SUBSTRING(player,POSITION('(' IN player),LENGTH(player)) LIKE
'%AFG%'

#Non Asian Continent
SELECT COUNT(player) AS no_of_non_asian_cricketer
FROM `batting figures`
WHERE SUBSTRING(player,POSITION('(' IN player), LENGTH(player))
LIKE '%AUS%' OR
      SUBSTRING(player,POSITION('(' IN player), LENGTH(player))
LIKE '%SA%' OR
       SUBSTRING(player,POSITION('(' IN player),
LENGTH(player)) LIKE '%ZIM%' OR
       SUBSTRING(player,POSITION('(' IN player),
LENGTH(player)) LIKE '%WI%' OR
       SUBSTRING(player,POSITION('(' IN player),
LENGTH(player)) LIKE '%ENG%' OR
       SUBSTRING(player,POSITION('(' IN player),
LENGTH(player)) LIKE '%NZ%' OR
       SUBSTRING(player,POSITION('(' IN player),
LENGTH(player)) LIKE '%IRE%'
```

PART B :

Richard's Supply,

"Richard's Supply" is a company which deals with different food products. The company is associated with a pool of suppliers. Every Supplier supplies different types of food products to Richard's supply. This company also receives orders for the food products from various customers. Each order may have multiple products mentioned along with the quantity. The company has been maintaining the database for 2 years.

Refer to the following Entity-Relationship diagram of the database.



Instruction: Execute the SQL files in the sequence given below.

1. 1_DDL_Case Study
2. 2_Data
3. 3_Data Constraints

1. Company sells the product at different discounted rates. Refer actual product price in product table and selling price in the order item table. Write a query to find out total amount saved in each order then display the orders from highest to lowest amount saved.

```
select orderId,p.unitprice as p_unitprice,oi.unitprice as
o_unitprice
from OrderItem oi inner join Product p
on oi.ProductId=p.ID)
select orderID, sum(p_unitprice-o_unitprice) as amount_saved
from cte
group by orderId
order by 2 desc;
```

2. Mr. Kavin want to become a supplier. He got the database of "Richard's Supply" for reference. Help him to pick:

    a. List few products that he should choose based on demand.

    b. Who will be the competitors for him for the products suggested in above questions.

```
-- a)
Select productname, count(*) as demand
from Product p inner join OrderItem oi
on p.id=oi.productId
group by productid
order by 2 desc
limit 10;

-- b)
With cte as
(Select supplierid,productname, count(*) as demand
from Product p inner join OrderItem oi
on p.id=oi.productId
group by productid
order by 3 desc
limit 10)
select cte.productName,s.*
from cte inner join Supplier s
```

```
on cte.supplierid=s.id ;
```

3. Create a combined list to display customers and suppliers details considering the following criteria
    - Both customer and supplier belong to the same country
    - Customer who does not have supplier in their country
    - Supplier who does not have customer in their country

```
-- a)
SELECT *
FROM supplier s INNER JOIN
product p ON
s.id = p.supplierid INNER JOIN
orderitem oi ON p.id = oi.productid
INNER JOIN orders o ON
o.id = oi.orderid INNER JOIN customer c
ON o.customerid = c.id
WHERE s.country = c.country
GROUP BY s.companyname

-- b)
SELECT *
FROM supplier s INNER JOIN
product p ON
s.id = p.supplierid INNER JOIN
orderitem oi ON p.id = oi.productid
INNER JOIN orders o ON
o.id = oi.orderid INNER JOIN customer c
ON o.customerid = c.id
WHERE s.country != c.country
GROUP BY s.companyname

-- c)
SELECT *
FROM supplier s INNER JOIN
product p ON
s.id = p.supplierid INNER JOIN
orderitem oi ON p.id = oi.productid
INNER JOIN orders o ON
o.id = oi.orderid INNER JOIN customer c
ON o.customerid = c.id
WHERE s.country != c.country
GROUP BY c.firstname
```

4. Every supplier supplies specific products to the customers. Create a view of suppliers and total sales made by their products and write a query on this view to find out top 2 suppliers (using windows function) in each country by total sales done by the products.

```
create or replace view supp_total_sales
```

```
as (
select p.supplierID, s.CompanyName as Supplier_Company_name,
s.country as supplier_country, p.ID as ProductID,
p.ProductName,count(*) as total_sales
from Product p inner join Supplier s
on p.SupplierId=s.ID inner join OrderItem oi
on p.ID=oi.ProductID
group by p.ID
);

select *
from (select Supplier_Company_Name, Supplier_Country,
    rank() over(partition by supplier_country order by total_sales)
as rn
     from supp_total_sales) a
where rn <= 2;
```

5. Find out for which products, UK is dependent on other countries for the supply. List the countries which are supplying these products in the same list.

```
select distinct p.ProductName,s.Country
 from Customer c inner join Orders o
 on c.ID=o.customerId and c.Country = 'UK'
 inner join OrderItem oi
 on o.id=oi.OrderId
 inner join Product p
 on oi.ProductId = p.ID
 inner join Supplier s
on p.SupplierId = s.ID and s.Country != 'UK';
```

6. Create two tables as ‘customer’ and ‘customer_backup’ as follow -

   ‘customer’ table attributes -
   Id, FirstName,LastName,Phone
   ‘customer_backup’ table attributes -
   Id, FirstName,LastName,Phone

   Create a trigger in such a way that It should insert the details into the ‘customer_backup’ table when you delete the record from the ‘customer’ table automatically.

```
-- 1.
create table customer as
(select id,firstname,lastname,phone from Supply_chain.Customer);
-- 2.
create table customer_backup as
(select * from customer where 1!=1);
-- creating trigger
-- 3.
DELIMITER $$
CREATE TRIGGER ins_backup
AFTER DELETE ON customer
FOR EACH ROW
```