



American International University-Bangladesh (AIUB)

ADVANCE DATABASE MANAGEMENT SYSTEM

SECTION: B

GROUP NAME: AVENGERS

**PROJECT TITLE: CRICKET CLUB MANAGEMENT
SYSTEM**

Submitted by:

SL	NAME	ID
01	HASAN, SHADIK	19-39357-1
02	HOSSEN, DIDAR	18-39084-3
03	PARVEEN, SUMAIYA	19-41542-3
04	RAHMAN, RAHAT	20-43401-1

Submitted to:

REZWAN AHMED

Table of Contents

INTRODUCTION:	4
SYSTEM SUMMARY:	4
ER DIAGRAM:	5
CLASS DIAGRAM:	6
USE CASE AND ACTIVITY DIAGRAM:	6
USE CASE:	7
ACTIVITY:	8
SCHEMA DIAGRAM:	8
ADMIN PANEL:	9
TABLE CREATION :	11
FOR CLUB :	11
FOR CLUB ADDRESS:	12
FOR OWNER:	12
FOR MANAGER:	13
FOR KIT:	13
FOR PLAYER:	13
FOR MEDICAL STUFF:	14
FOR STADIUM:	14
FOR STADIUM ADDRESS:	15
FOR GROUND STUFF:	15
FOR COACH:	15
FOR FOREIGN KEY OWNER PLAYER:	16
FOR FOREIGN KEY PLAYER MEDICAL STUFF:	16
FOR FOREIGN KEY OWNER STADIUM:	17
FOR FOREIGN KEY STADIUM GROUND STUFF:	17
FOR FOREIGN KEY MANAGER COACH:	18
DATA INSERTION:	18
FOR CLUB :	18
FOR CLUB ADDRESS:	19
FOR OWNER:	19

FOR MANAGER:.....	20
FOR KIT:	20
FOR PLAYER:	21
FOR MEDICAL STUFF:	22
FOR STADIUM:	22
FOR STADIUM ADDRESS:.....	23
FOR GROUND STUFF:	23
FOR COACH:	24
FOR FOREIGN KEY OWNER PLAYER:.....	24
FOR FOREIGN KEY PLAYER MEDICAL STUFF:.....	25
FOR FOREIGN KEY OWNER STADIUM:	26
FOR FOREIGN KEY STADIUM GROUND STUFF:	26
FOR FOREIGN KEY MANAGER COACH:.....	27
QUERY WRITING:.....	28
VIEW:.....	30
PROCEDURE AND FUNCTION:	34
PROCEDURE.....	34
FUNCTION:	37
TRIGGER:	42
CONCLUSION :	48
IN THE FUTURE :	48

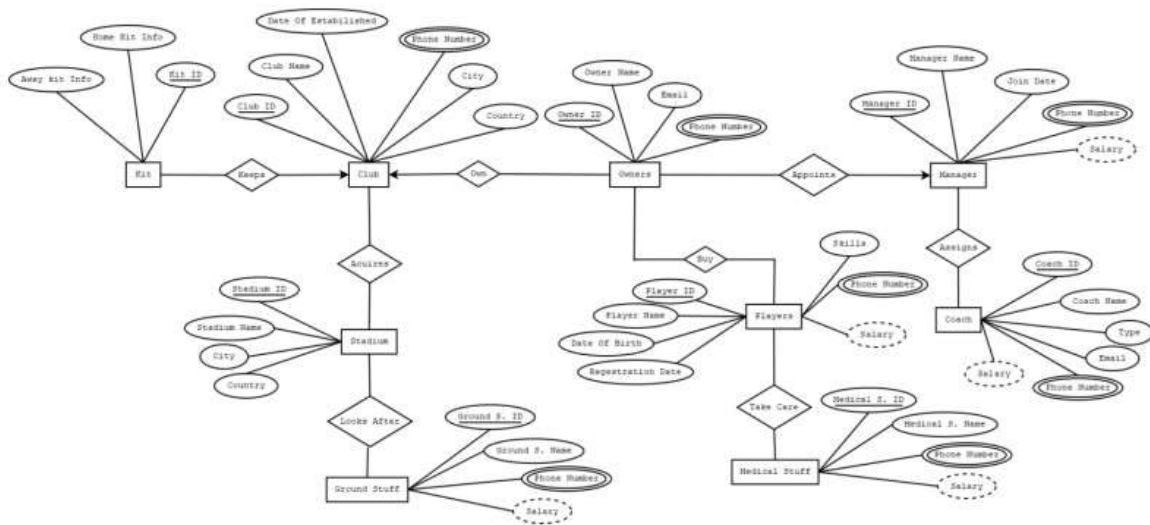
INTRODUCTION:

The title of the project is “Cricket club management system”. Cricket club management project is designed with the motive of managing a cricket club. This software system consists of various online booking and management functionalities needed by a cricket club. This software system assures efficient management and maintains the functioning on a cricket club. The main aim of cricket club management is to manage the details of cricket, players, events, fees, schedules etc. . We do believe that this project will be beneficial to a lot of new or updatable Cricket club management system.

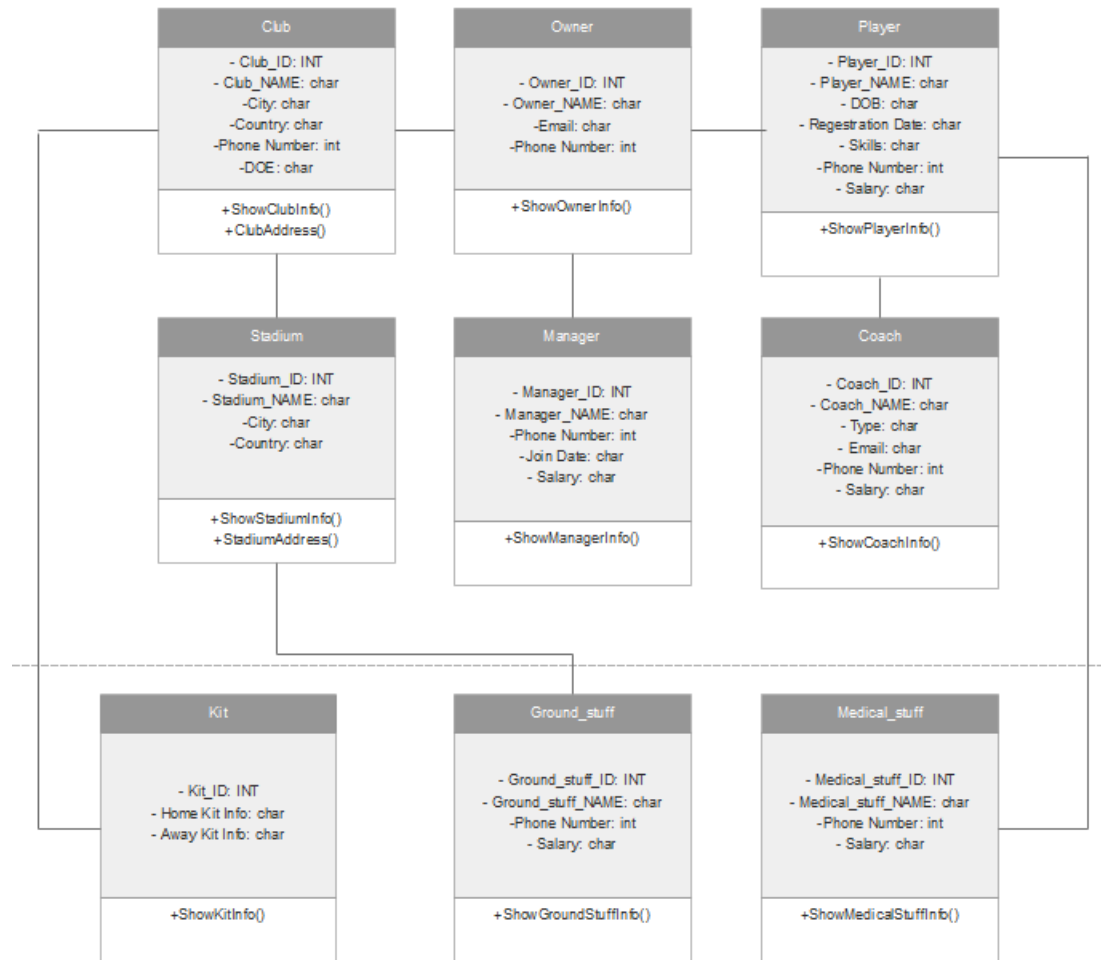
SYSTEM SUMMARY:

In Cricket Club Management System To identify a club the system stores club name, club ID, established date, location and phone numbers. Every Club acquires one or many stadiums. Every stadium has stadium name, location and one stadium ID. Ground stuffs look after those stadiums. One ground stuff has only one Id but may have more than one phone number. Club may have many Owners. A owner is identified by Owner_Id. An Owner information composed of name, email and one or more phone number. Owners appoints only one Manager for the club. Owners can buy many players. Every Player can be identified by their Player ID. Club also store players data like Player name, date of birth, registration date, skills, phone numbers and salary. Manager can assign coaches. To take care of the players club assigns medical stuffs. A medical Stuff have one medical_s.ID. The system also stores their name and phone numbers. Every coach has their individual one Id and may have more than one phone numbers. The System also store their name, address, Email address. Club keeps their Kits. kit information consists of home kit info, away kit info and one kit ID.

ER DIAGRAM:

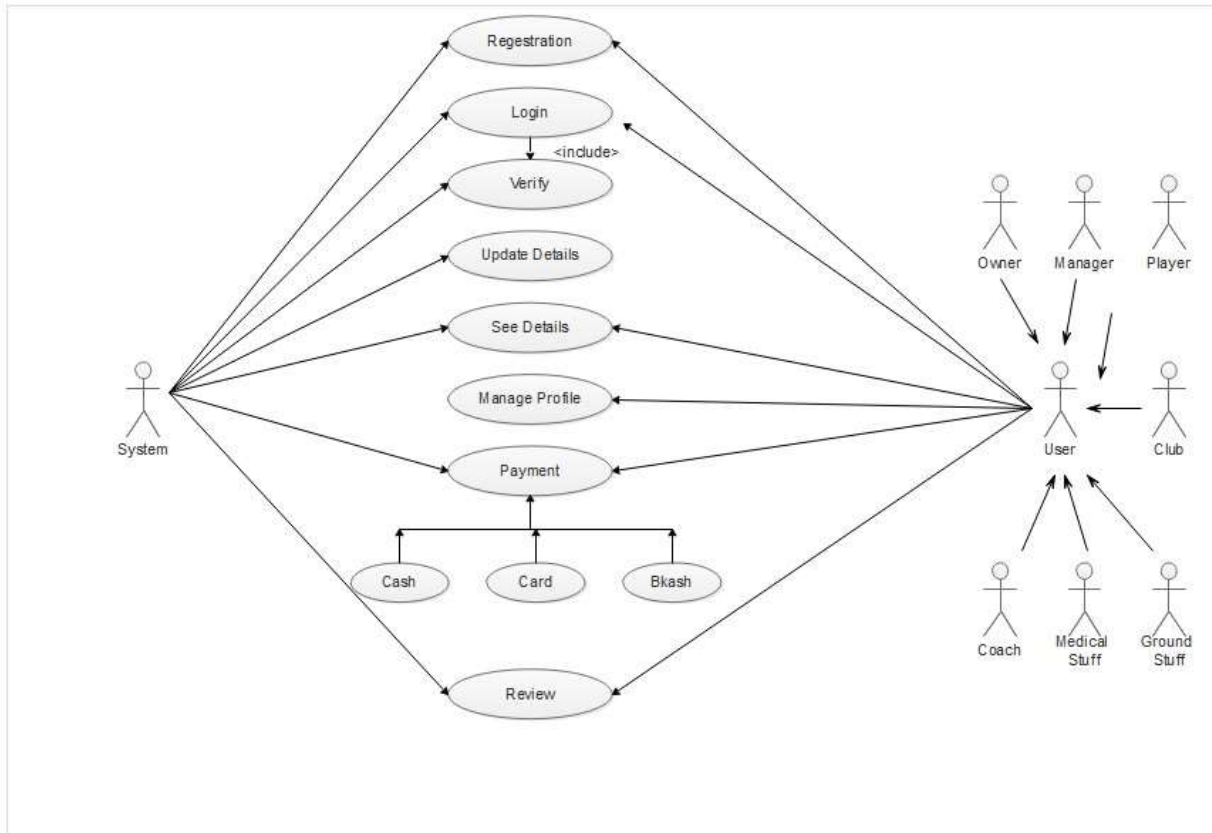


CLASS DIAGRAM:

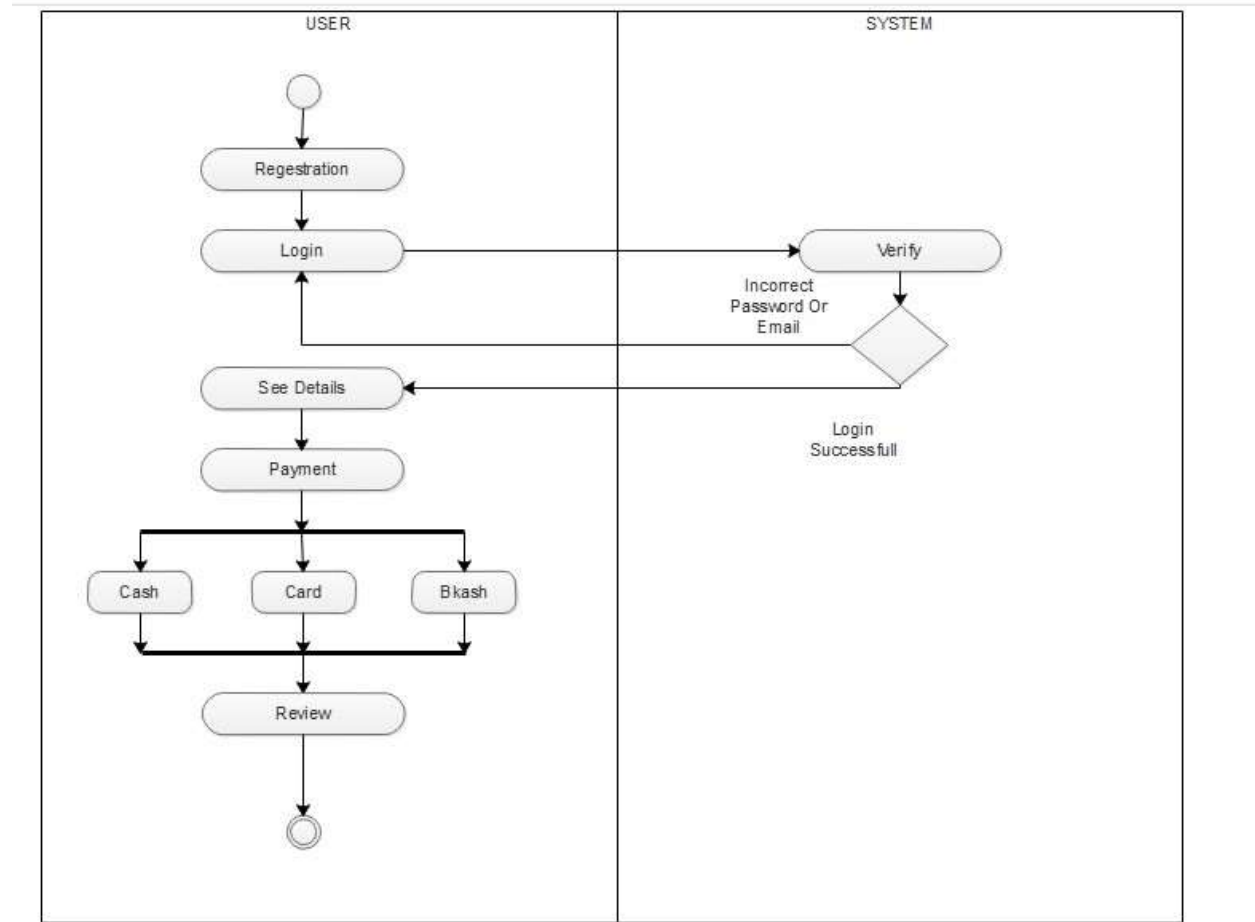


USE CASE AND ACTIVITY DIAGRAM:

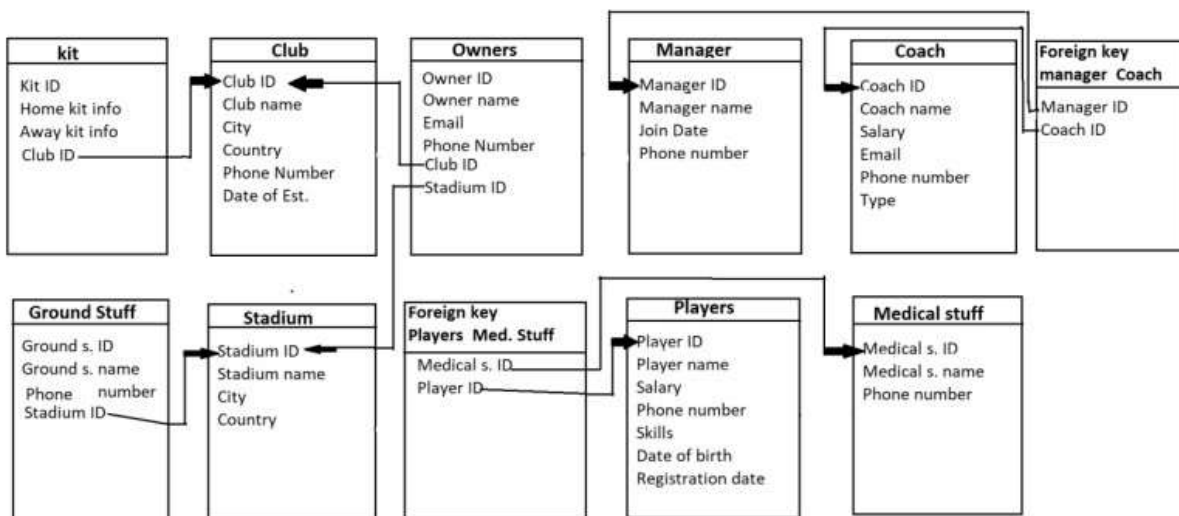
USE CASE:



ACTIVITY:



SCHEMA DIAGRAM:



ADMIN PANEL:

LET'S CREATE ACCOUNT

USER NAME

E-MAIL ADDRESS

PASSWORD SHOW

CONFIRM PASSWORD

CREATE ACCOUNT

I ALREADY HAVE AN ACCOUNT? [LOG IN](#)

This is the registration page with validation and connected with oracle 10g.

LET'S LOG-IN

E-MAIL ADDRESS

PASSWORD SHOW

FORGOT YOUR PASSWORD??

LOG-IN

I DON'T HAVE AN ACCOUNT? [CREATE ACCOUNT NOW](#)

This is the Login page with validation and connected with oracle 10g.

TABLES				
SEARCH A TABLE...				
CLUB				
OWNER				
PLAYER				
COACH				
MANAGER				
STADIUM				
KIT				
MEDICAL STUFF				
GROUND STUFF				
5 CLUB	6 OWNER	5 PLAYER	5 COACH	7 MANAGER

This is a admin dashboard page. This is also connected with oracle 10g. Here the admin can search for a table and view the table page. And for the table, a count was shown that is done by the counting from oracle.

CLUB DATA						
<div> <div>SEARCH BY ID...</div> <div>SEARCH</div> </div>						
CLUB_ID	CLUB_NAME	DATE_OF_ESTABLISHED	CLUB_PHONE_NUMBER_1	CLUB_PHONE_NUMBER_2	CLUB_PHONE_NUMBER_3	ACTION
1001	DHAKA PLATOON	2013	94652641	58412658	179654166	EDIT DELETE
1051	RAJSHAHI ROYELS	2012	9545463	1785621365	1941635126	EDIT DELETE
1101	RANGPUR RANGERS	2012	54308523	241574521	1815461265	EDIT DELETE
1151	CUMILLA WARRIORS	2015	84165155	1485126352	1354218518	EDIT DELETE
1201	BARISHAL BULLS	2012	87486542	98864697	179655448	EDIT DELETE

CLUB ID

CLUB NAME

DATE OF ESTABLISHED

CLUB PHONE NUMBER 1

CLUB PHONE NUMBER 2

CLUB PHONE NUMBER 3

SAVE

This is the page for a table and here admin can see the table which is on oracle. And also here admin can edit data for update and delete. Also if the admin can search for data he/she can search by id.

TABLE CREATION :

FOR CLUB :

```
CREATE TABLE CLUB (CLUB_ID NUMBER(10) PRIMARY KEY, CLUB_NAME VARCHAR(25),
DATE_OF_ESTABLISHED VARCHAR(10), CLUB_PHONE_NUMBER_1 NUMBER(11),
CLUB_PHONE_NUMBER_2 NUMBER(11), CLUB_PHONE_NUMBER_3 NUMBER(11));
```

```
CREATE SEQUENCE CLUB_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50
NOCACHE NOCYCLE;
```

```
DESC CLUB;
```

Table	Column	Size	Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CLUB	CLUB_ID	10	Number	10		0	✓	✓		
	CLUB_NAME	25	Varchar2	25				✓		
	DATE_OF_ESTABLISHED	10	Varchar2	10				✓		
	CLUB_PHONE_NUMBER_1	11	Number	11		0		✓		
	CLUB_PHONE_NUMBER_2	11	Number	11		0		✓		
	CLUB_PHONE_NUMBER_3	11	Number	11		0		✓		

FOR CLUB ADDRESS:

```
CREATE TABLE CLUB_ADDRESS (A_ID NUMBER(10) PRIMARY KEY, CLUB_CITY VARCHAR(30), CLUB_COUNTRY VARCHAR(30));
```

```
CREATE SEQUENCE A_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50 NOCACHE NOCYCLE;
```

```
DESC CLUB;
```



The screenshot shows the 'Structure' tab for the 'CLUB_ADDRESS' table. The table has three columns: A_ID (NUMBER(10), Primary Key), CLUB_CITY (VARCHAR(30)), and CLUB_COUNTRY (VARCHAR(30)).

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
A_ID	NUMBER	10		0	✓	✓		
CLUB_CITY	VARCHAR2	30				✓		
CLUB_COUNTRY	VARCHAR2	30				✓		

FOR OWNER:

```
CREATE TABLE OWNER (OWNER_ID NUMBER(10) PRIMARY KEY, OWNER_NAME VARCHAR(30), EMAIL VARCHAR(30), OWNER_PHONE_NUMBER_1 NUMBER(11), OWNER_PHONE_NUMBER_2 NUMBER(11), OWNER_PHONE_NUMBER_3 NUMBER(11), CLUB_ID NUMBER(10), A_ID NUMBER(10), S_ID NUMBER(10), STADIUM_ID NUMBER(10));
```

```
ALTER TABLE OWNER ADD CONSTRAINT FK FOREIGN KEY (A_ID) REFERENCES CLUB_ADDRESS (A_ID);
```

```
ALTER TABLE OWNER ADD CONSTRAINT FK FOREIGN KEY (S_ID) REFERENCES STADIUM_ADDRESS (S_ID);
```

```
CREATE SEQUENCE OWNER_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50 NOCACHE NOCYCLE;
```

```
DESC OWNER;
```



The screenshot shows the 'Structure' tab for the 'OWNER' table. The table has ten columns: OWNER_ID (NUMBER(10), Primary Key), OWNER_NAME (VARCHAR(30)), EMAIL (VARCHAR(30)), OWNER_PHONE_NUMBER_1 (NUMBER(11)), OWNER_PHONE_NUMBER_2 (NUMBER(11)), OWNER_PHONE_NUMBER_3 (NUMBER(11)), CLUB_ID (NUMBER(10)), A_ID (NUMBER(10)), S_ID (NUMBER(10)), and STADIUM_ID (NUMBER(10)).

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
OWNER_ID	NUMBER	10		0	✓	✓		
OWNER_NAME	VARCHAR2	30				✓		
EMAIL	VARCHAR2	30				✓		
OWNER_PHONE_NUMBER_1	NUMBER	11		0		✓		
OWNER_PHONE_NUMBER_2	NUMBER	11		0		✓		
OWNER_PHONE_NUMBER_3	NUMBER	11		0		✓		
CLUB_ID	NUMBER	10		0		✓		
A_ID	NUMBER	10		0		✓		
S_ID	NUMBER	10		0		✓		
STADIUM_ID	NUMBER	10		0		✓		

FOR MANAGER:

```
CREATE TABLE MANAGER (MANAGER_ID NUMBER(10) PRIMARY KEY, MANAGER_NAME  
VARCHAR(30), JOIN_DATE VARCHAR(10), MANAGER_PHONE_NUMBER_1 NUMBER(11),  
MANAGER_PHONE_NUMBER_2 NUMBER(11), MANAGER_PHONE_NUMBER_3 NUMBER(11),  
MANAGER_SALARY NUMBER(20));
```

```
CREATE SEQUENCE MANAGER_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50  
NOCACHE NOCYCLE;
```

```
DESC MANAGER;
```



The screenshot shows the 'Describe' tab for the MANAGER table. The table has 7 columns: MANAGER_ID (Number, 10, Primary Key), MANAGER_NAME (Varchar2, 30), JOIN_DATE (Varchar2, 10), MANAGER_PHONE_NUMBER_1 (Number, 11), MANAGER_PHONE_NUMBER_2 (Number, 11), MANAGER_PHONE_NUMBER_3 (Number, 11), and MANAGER_SALARY (Number, 20). The primary key is indicated by a key icon in the Primary Key column for MANAGER_ID.

Name	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
MANAGER_ID	MANAGER_ID	Number	10		0	✓			
MANAGER_NAME	MANAGER_NAME	Varchar2	30				✓		
JOIN_DATE	JOIN_DATE	Varchar2	10				✓		
MANAGER_PHONE_NUMBER_1	MANAGER_PHONE_NUMBER_1	Number	11		0		✓		
MANAGER_PHONE_NUMBER_2	MANAGER_PHONE_NUMBER_2	Number	11		0		✓		
MANAGER_PHONE_NUMBER_3	MANAGER_PHONE_NUMBER_3	Number	11		0		✓		
MANAGER_SALARY	MANAGER_SALARY	Number	20		0		✓		

FOR KIT:

```
CREATE TABLE KIT (KIT_ID NUMBER(10) PRIMARY KEY, HOME_KIT_INFO VARCHAR(30),  
AWAY_KIT_INFO VARCHAR(30), CLUB_ID NUMBER (10), A_ID NUMBER(10));
```

```
CREATE SEQUENCE KIT_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50  
NOCACHE NOCYCLE;
```

```
DESC KIT;
```



The screenshot shows the 'Describe' tab for the KIT table. The table has 5 columns: KIT_ID (Number, 10, Primary Key), HOME_KIT_INFO (Varchar2, 30), AWAY_KIT_INFO (Varchar2, 30), CLUB_ID (Number, 10), and A_ID (Number, 10). The primary key is indicated by a key icon in the Primary Key column for KIT_ID.

Name	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
KIT_ID	KIT_ID	Number	10		0	✓			
HOME_KIT_INFO	HOME_KIT_INFO	Varchar2	30				✓		
AWAY_KIT_INFO	AWAY_KIT_INFO	Varchar2	30				✓		
CLUB_ID	CLUB_ID	Number	10		0		✓		
A_ID	A_ID	Number	10		0		✓		

FOR PLAYER:

```
CREATE TABLE PLAYER (PLAYER_ID NUMBER(10) PRIMARY KEY, PLAYER_NAME VARCHAR(30),  
DATE_OF_BIRTH VARCHAR(10), REGISTRATION_DATE VARCHAR(10),SKILLS  
VARCHAR(10),PLAYER_PHONE_NUMBER_1 NUMBER(11),PLAYER_PHONE_NUMBER_2  
NUMBER(11), PLAYER_PHONE_NUMBER_3 NUMBER(11), PLAYER_SALARY NUMBER(20));
```

```
CREATE SEQUENCE PLAYER_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50  
NOCACHE NOCYCLE;
```

DESC PLAYER;

Results Explain Describe Saved SQL History

Object Type: TABLE Object: PLAYER

Table	Columns	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLAYER	PLAYER_ID	Number	10		0	1	✓		
	PLAYER_NAME	Varchar2	30				✓		
	DATE_OF_BIRTH	Varchar2	10				✓		
	HEIGHT_IN_CM	Varchar2	10				✓		
	WEIGHT	Varchar2	10				✓		
	PLAYER_PHONE_NUMBER_1	Number	11		0		✓		
	PLAYER_PHONE_NUMBER_2	Number	11		0		✓		
	PLAYER_PHONE_NUMBER_3	Number	11		0		✓		
	PLAYER_SALARY	Number	20		0		✓		

1-8

FOR MEDICAL STUFF:

CREATE TABLE MEDICAL_STUFF (MEDICAL_S_ID NUMBER(10) PRIMARY KEY, MEDICAL_S_NAME VARCHAR(30), MEDICAL_S_PHONE_NUMBER_1 NUMBER(11), MEDICAL_S_PHONE_NUMBER_2 NUMBER(11), MEDICAL_S_PHONE_NUMBER_3 NUMBER(11), MEDICAL_S_SALARY NUMBER(20));

CREATE SEQUENCE MEDICAL_S_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50 NOCACHE NOCYCLE;

DESC MEDICAL_STUFF;

Results Explain Describe Saved SQL History

Object Type: TABLE Object: MEDICAL_STUFF

Table	Columns	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MEDICAL_STUFF	MEDICAL_S_ID	Number	10		0	1	✓		
	MEDICAL_S_NAME	Varchar2	30				✓		
	MEDICAL_S_PHONE_NUMBER_1	Number	11		0		✓		
	MEDICAL_S_PHONE_NUMBER_2	Number	11		0		✓		
	MEDICAL_S_PHONE_NUMBER_3	Number	11		0		✓		
	MEDICAL_S_SALARY	Number	20		0		✓		

1-8

Application Express 2.1.0.0.19
Copyright © 2009, 2010, 2011 Oracle. All rights reserved.

FOR STADIUM:

CREATE TABLE STADIUM (STADIUM_ID NUMBER(10) PRIMARY KEY, STADIUM_NAME VARCHAR(30));

CREATE SEQUENCE STADIUM_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50 NOCACHE NOCYCLE;

DESC STADIUM;

Results Explain Describe Saved SQL History

Object Type: TABLE Object: STADIUM

Table	Columns	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STADIUM	STADIUM_ID	Number	10		0	1	✓		
	STADIUM_NAME	Varchar2	30				✓		

1-2

FOR STADIUM ADDRESS:

```
CREATE TABLE STADIUM_ADDRESS (S_ID NUMBER(10) PRIMARY KEY, STADIUM_CITY  
VARCHAR(30), STADIUM_COUNTRY VARCHAR(30));
```

```
CREATE SEQUENCE S_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50  
NOCACHE NOCYCLE;
```

```
DESC STADIUM_ADDRESS;
```



The screenshot shows the 'TABLE: STADIUM_ADDRESS' structure in SQL Developer. The table has three columns: S_ID (NUMBER(10), Primary Key), STADIUM_CITY (VARCHAR(30)), and STADIUM_COUNTRY (VARCHAR(30)).

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
STADIUM_ADDRESS	S_ID	Number	10		0	✓	✓		
STADIUM_ADDRESS	STADIUM_CITY	Varchar2	30				✓		
STADIUM_ADDRESS	STADIUM_COUNTRY	Varchar2	30				✓		

FOR GROUND STUFF:

```
CREATE TABLE GROUND_STUFF (GROUND_S_ID NUMBER(10) PRIMARY KEY, GROUND_S_NAME  
VARCHAR(30), GROUND_S_PHONE_NUMBER_1 NUMBER(11), GROUND_S_PHONE_NUMBER_2  
NUMBER(11), GROUND_S_PHONE_NUMBER_3 NUMBER(11), GROUND_S_SALARY  
NUMBER(20), S_ID NUMBER(10), STADIUM_ID NUMBER(10));
```

```
ALTER TABLE GROUND_STUFF ADD CONSTRAINT GK FOREIGN KEY (S_ID) REFERENCES  
STADIUM_ADDRESS (S_ID);
```

```
CREATE SEQUENCE GROUND_S_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50  
NOCACHE NOCYCLE;
```

```
DESC GROUND_STUFF;
```



The screenshot shows the 'TABLE: GROUND_STUFF' structure in SQL Developer. The table has eight columns: GROUND_S_ID (NUMBER(10), Primary Key), GROUND_S_NAME (VARCHAR(30)), GROUND_S_PHONE_NUMBER_1 (NUMBER(11)), GROUND_S_PHONE_NUMBER_2 (NUMBER(11)), GROUND_S_PHONE_NUMBER_3 (NUMBER(11)), GROUND_S_SALARY (NUMBER(20)), S_ID (NUMBER(10)), and STADIUM_ID (NUMBER(10)).

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
GROUND_STUFF	GROUND_S_ID	Number	10		0	✓	✓		
GROUND_STUFF	GROUND_S_NAME	Varchar2	30				✓		
GROUND_STUFF	GROUND_S_PHONE_NUMBER_1	Number	11		0		✓		
GROUND_STUFF	GROUND_S_PHONE_NUMBER_2	Number	11		0		✓		
GROUND_STUFF	GROUND_S_PHONE_NUMBER_3	Number	11		0		✓		
GROUND_STUFF	GROUND_S_SALARY	Number	20		0		✓		
GROUND_STUFF	S_ID	Number	10		0		✓		
GROUND_STUFF	STADIUM_ID	Number	10		0		✓		

FOR COACH:

```
CREATE TABLE COACH (COACH_ID NUMBER(10) PRIMARY KEY, COACH_NAME  
VARCHAR(30), TYPE VARCHAR(30), EMAIL VARCHAR(30), COACH_PHONE_NUMBER_1  
NUMBER(11), COACH_PHONE_NUMBER_2 NUMBER(11), COACH_PHONE_NUMBER_3  
NUMBER(11), COACH_SALARY NUMBER(20));
```

```
CREATE SEQUENCE COACH_ID_SEQ START WITH 1001 MAXVALUE 90000 INCREMENT BY 50
NOCACHE NOCYCLE;
```

```
DESC COACH;
```

The screenshot shows the 'TABLE Object: COACH' in SQL Developer. The table has the following columns:

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
COACH_ID	NUMBER	10		0	✓	✓		
COACH_NAME	VARCHAR2	30				✓		
DOB	DATE					✓		
EMAIL	VARCHAR2	30				✓		
COACH_PHOTO_PATH	NUMBER	11		0		✓		
COACH_PHOTO_PATH_2	NUMBER	11		0		✓		
COACH_PHOTO_PATH_3	NUMBER	11		0		✓		
COACH_PHOTO_PATH_4	NUMBER	11		0		✓		
COACH_PHOTO_PATH_5	NUMBER	11		0		✓		

FOR FOREIGN KEY OWNER PLAYER:

```
CREATE TABLE FOREIGN_KEY_OWNER_PLAYER (OWNER_ID NUMBER(10), PLAYER_ID
NUMBER(10));
```

```
ALTER TABLE FOREIGN_KEY_OWNER_PLAYER ADD CONSTRAINT IF FOREIGN KEY (OWNER_ID)
REFERENCES OWNER (OWNER_ID);
```

```
ALTER TABLE FOREIGN_KEY_OWNER_PLAYER ADD CONSTRAINT LF FOREIGN KEY (PLAYER_ID)
REFERENCES PLAYER (PLAYER_ID);
```

```
DESC FOREIGN_KEY_OWNER_PLAYER;
```

The screenshot shows the 'TABLE Object: FOREIGN_KEY_OWNER_PLAYER' in SQL Developer. The table has the following columns:

Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
OWNER_ID	NUMBER	10		0	✓	✓		
PLAYER_ID	NUMBER	10		0		✓		

FOR FOREIGN KEY PLAYER MEDICAL STUFF:

```
CREATE TABLE FOREIGN_KEY_PLAYER_MS (PLAYER_ID NUMBER(10), MEDICAL_S_ID
NUMBER(10));
```

```
ALTER TABLE FOREIGN_KEY_PLAYER_MS ADD CONSTRAINT MF FOREIGN KEY (PLAYER_ID)
REFERENCES PLAYER (PLAYER_ID);
```

```
ALTER TABLE FOREIGN_KEY_PLAYER_MS ADD CONSTRAINT AF FOREIGN KEY (MEDICAL_S_ID)
REFERENCES MEDICAL_STUFF (MEDICAL_S_ID);
```

```
DESC FOREIGN_KEY_PLAYER_MS;
```


Results	Explain	Describe	Save SQL	History					
Object Type: TABLE Object: FOREIGN_KEY_PLAYER_MS									
Name	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
FOREIGN_KEY_PLAYER_MS	PLAYER_ID	Number	10	0			✓	-	-
	STADIUM_ID	Number	10	0			✓	-	-
1-2									

Language: sql

Application Express 4.1.5.0.18
Copyright © 1996, 2018, Oracle. All rights reserved.

FOR FOREIGN KEY OWNER STADIUM:

```
CREATE TABLE FOREIGN_KEY_OWNER_STADIUM (OWNER_ID NUMBER(10), STADIUM_ID
NUMBER(10));
```

```
ALTER TABLE FOREIGN_KEY_OWNER_STADIUM ADD CONSTRAINT HF FOREIGN KEY
(OWNER_ID) REFERENCES OWNER (OWNER_ID);
```

```
ALTER TABLE FOREIGN_KEY_OWNER_STADIUM ADD CONSTRAINT BF FOREIGN KEY
(STADIUM_ID) REFERENCES STADIUM (STADIUM_ID);
```

```
DESC FOREIGN_KEY_OWNER_STADIUM;
```

Results Explain Describe Save SQL History

Object Type: TABLE Object: FOREIGN_KEY_OWNER_STADIUM

Name	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
FOREIGN_KEY_OWNER_STADIUM	OWNER_ID	Number	10	0			✓		
	STADIUM_ID	Number	10	0			✓		

1 - 2

Language: sql

Application Express 4.1.5.0.18
Copyright © 1996, 2018, Oracle. All rights reserved.

FOR FOREIGN KEY STADIUM GROUND STUFF:

```
CREATE TABLE FOREIGN_KEY_STADIUM_GS (STADIUM_ID NUMBER(10), GROUND_S_ID
NUMBER(10));
```

```
ALTER TABLE FOREIGN_KEY_STADIUM_GS ADD CONSTRAINT UF FOREIGN KEY (STADIUM_ID)
REFERENCES STADIUM (STADIUM_ID);
```

```
ALTER TABLE FOREIGN_KEY_STADIUM_GS ADD CONSTRAINT VF FOREIGN KEY (GROUND_S_ID)
REFERENCES GROUND_STUFF (GROUND_S_ID);
```

```
DESC FOREIGN_KEY_STADIUM_GS;
```

Results	Explain	Describe	Save SQL	History					
Object Type: TABLE Object: FOREIGN_KEY_STADIUM_GS									
Name	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
FOREIGN_KEY_STADIUM_GS	STADIUM_ID	Number	10	0			✓		
	GROUND_S_ID	Number	10	0			✓		
1/2									

FOR FOREIGN KEY MANAGER COACH:

```
CREATE TABLE FOREIGN_KEY_MANAGER_COACH (MANAGER_ID NUMBER(10), COACH_ID  
NUMBER(10));
```

```
ALTER TABLE FOREIGN_KEY_MANAGER_COACH ADD CONSTRAINT CF FOREIGN KEY  
(MANAGER_ID) REFERENCES MANAGER (MANAGER_ID);
```

```
ALTER TABLE FOREIGN_KEY_MANAGER_COACH ADD CONSTRAINT DF FOREIGN KEY  
(COACH_ID) REFERENCES COACH (COACH_ID);
```

```
DESC FOREIGN_KEY_MANAGER_COACH;
```



The screenshot shows a database tool interface with a table structure view for 'FOREIGN_KEY_MANAGER_COACH'. The table has two columns: 'MANAGER_ID' and 'COACH_ID', both of type 'NUMBER' with a length of 10. The 'MANAGER_ID' column is marked as the primary key. The interface includes tabs for 'Results', 'Caption', 'Describe', 'Event SQL', and 'History'.

Column Name	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comments
MANAGER_ID	NUMBER	10	0		Yes	Yes		
COACH_ID	NUMBER	10	0		No	Yes		

DATA INSERTION:

FOR CLUB :

```
INSERT INTO CLUB VALUES (CLUB_ID_SEQ.NEXTVAL, 'DHAKA PLATOON', '2015', '094652641',  
'058412656', '0179654166');
```

```
INSERT INTO CLUB VALUES (CLUB_ID_SEQ.NEXTVAL, 'RAJSHAHI ROYELS', '2012', '09545463',  
'01785621265', '01941635126');
```

```
INSERT INTO CLUB VALUES (CLUB_ID_SEQ.NEXTVAL, 'RANGPUR RANGERS', '2012', '054308523',  
'0241574521', '01815461265');
```

```
INSERT INTO CLUB VALUES (CLUB_ID_SEQ.NEXTVAL, 'CUMILLA WARRIORS', '2015', '0841651',  
'01485126352', '0135421851');
```

```
INSERT INTO CLUB VALUES (CLUB_ID_SEQ.NEXTVAL, 'BARISHAL BULLS', '2012', '87486542',  
'9886464', '01796554486');
```

```
SELECT * FROM CLUB;
```

Results Explain Describe Saved SQL History

5 rows

CLUB_ID	CLUB_NAME	DATE_OF_ESTABLISHED	CLUB_PHONE_NUMBER_1	CLUB_PHONE_NUMBER_2	CLUB_PHONE_NUMBER_3
1001	DHAKA PLATOON	2010	94022941	99422999	079054199
1051	RAJSHAHI ROYALE	2012	8645401	078021289	0941939120
1101	RANGPUR RANGERS	2012	94209221	241518221	1015481200
1151	CUMILLA MARCOES	2010	941030	140512582	030421001
1201	BARISHAL BULLS	2012	0700542	990004	078024480

5 rows returned in 0.00 seconds

CSV Export

Application Express 2.1.1.18.19

Copyright © 1996-2005, Oracle. All rights reserved.

FOR CLUB ADDRESS:

```

INSERT INTO CLUB_ADDRESS VALUES (A_ID_SEQ.NEXTVAL, 'DHAKA', 'BANGLADESH');

INSERT INTO CLUB_ADDRESS VALUES (A_ID_SEQ.NEXTVAL, 'RAJSHAHI', 'BANGLADESH');

INSERT INTO CLUB_ADDRESS VALUES (A_ID_SEQ.NEXTVAL, 'RANGPUR', 'BANGLADESH');

INSERT INTO CLUB_ADDRESS VALUES (A_ID_SEQ.NEXTVAL, 'CUMILLA', 'BANGLADESH');

INSERT INTO CLUB_ADDRESS VALUES (A_ID_SEQ.NEXTVAL, 'BARISHAL', 'BANGLADESH');

SELECT * FROM CLUB_ADDRESS;

```

Results Explain Describe Saved SQL History

A_ID	CLUB_CITY	CLUB_COUNTRY
1001	DHAKA	BANGLADESH
1051	RAJSHAHI	BANGLADESH
1101	RANGPUR	BANGLADESH
1151	CUMILLA	BANGLADESH
1201	BARISHAL	BANGLADESH

5 rows returned in 0.00 seconds CSV Export

FOR OWNER:

```

INSERT INTO OWNER VALUES (OWNER_ID_SEQ.NEXTVAL, 'BEXIMCO GROUP',
'beximco@gmail.com', '065416953', '06526316', '01865326541', '1001', '1001', '1151', '1201');

INSERT INTO OWNER VALUES (OWNER_ID_SEQ.NEXTVAL, 'BANGLADESH CRICKET BOARD',
'bcb@gmail.com', '0695165', '09654152', '036525463', '1051', '1051', '1201', '1251');

INSERT INTO OWNER VALUES (OWNER_ID_SEQ.NEXTVAL, 'BANGLADESH CRICKET BOARD',
'bcb@gmail.com', '84541206321', '5841665', '014584169', '1201', '1101', '1251', '1301');

```

```
INSERT INTO OWNER VALUES (OWNER_ID_SEQ.NEXTVAL, 'MUSTAFA RAFIQUUL ISLAM',
'mustafa@gmail.com', '01986526', '01965219', '0106526341', '1251', '1101', '1301', '1351');
```

```
INSERT INTO OWNER VALUES (OWNER_ID_SEQ.NEXTVAL, 'NAFISA KAMAL', 'nafisa@gmail.com',
'08562652', '048541652', '05849652', '1301', '1201', '1351', '1401');
```

```
SELECT * FROM OWNER;
```

OWNER_ID	OWNER_NAME	EMAIL	OWNER_PHONE_NUMBER_1	OWNER_PHONE_NUMBER_2	OWNER_PHONE_NUMBER_3	CLUB_ID	A_ID	S_ID	STADIUM_ID
1051	BEKMACO GROUP	beemco@gmail.com	6541895	6526318	186526541	1001	1001	1101	1201
1101	BANGLADESH CRICKET BOARD	bcb@gmail.com	666165	9664152	36625463	1051	1051	1201	1251
1251	BANGLADESH CRICKET BOARD	bcb@gmail.com	85541284221	5841665	16884159	1201	1101	1101	1101
1301	MUSTAFA RAFIQUUL ISLAM	mustafa@gmail.com	1986526	1965219	106526341	1251	1101	1101	1101
1351	NAFISA KAMAL	nafisa@gmail.com	8562652	48541952	5849652	1101	1201	1101	1201

3 rows returned in 0.00 seconds

FOR MANAGER:

```
INSERT INTO MANAGER VALUES (MANAGER_ID_SEQ.NEXTVAL, 'FARHAN', '15-10-2017',
'01085165846', '018542966', '08541965', '50000');
```

```
INSERT INTO MANAGER VALUES (MANAGER_ID_SEQ.NEXTVAL, 'FARIZ', '18-12-2019',
'086529652', '0874296635', '056412965', '40000');
```

```
INSERT INTO MANAGER VALUES (MANAGER_ID_SEQ.NEXTVAL, 'SADI', '01-06-2020',
'058654296', '0541896526', '01875496', '50000');
```

```
INSERT INTO MANAGER VALUES (MANAGER_ID_SEQ.NEXTVAL, 'MUSTAFA', '03-02-2018',
'06526952', '056296635', '089645296', '45000');
```

```
INSERT INTO MANAGER VALUES (MANAGER_ID_SEQ.NEXTVAL, 'NAFI', '09-08-2021',
'085461265', '096412966', '056429652', '50000');
```

```
SELECT * FROM MANAGER;
```

MANAGER_ID	MANAGER NAME	JOIN DATE	MANAGER_PHONE NUMBER_1	MANAGER_PHONE NUMBER_2	MANAGER_PHONE NUMBER_3	MANAGER SALARY
1001	FARHAN	15-10-2017	100165846	1842966	841965	48000
1051	FARIZ	18-12-2019	86529652	87429635	5412965	40000
1101	SADI	01-06-2020	5864296	54189626	1875496	50000
1251	MUSTAFA	03-02-2018	6526952	56296635	89645296	45000
1301	NAFI	09-08-2021	85461265	96412966	56429652	50000
1351	NAFI	09-08-2021	8481264	9412966	56429652	48000

6 rows returned in 0.00 seconds

FOR KIT:

```
INSERT INTO KIT VALUES (KIT_ID_SEQ.NEXTVAL, 'YELLOW', 'GREEN', CLUB_ID_SEQ.NEXTVAL,
A_ID_SEQ.NEXTVAL);
```

```
INSERT INTO KIT VALUES (KIT_ID_SEQ.NEXTVAL, 'BLUE', 'ORANGE', CLUB_ID_SEQ.NEXTVAL,
A_ID_SEQ.NEXTVAL);
```

```

INSERT INTO KIT VALUES (KIT_ID_SEQ.NEXTVAL, 'BLACK', 'RED', CLUB_ID_SEQ.NEXTVAL,
A_ID_SEQ.NEXTVAL);

INSERT INTO KIT VALUES (KIT_ID_SEQ.NEXTVAL, 'WHITE', 'PURPLE', CLUB_ID_SEQ.NEXTVAL,
A_ID_SEQ.NEXTVAL);

INSERT INTO KIT VALUES (KIT_ID_SEQ.NEXTVAL, 'BLACK','ORANGE', CLUB_ID_SEQ.NEXTVAL,
A_ID_SEQ.NEXTVAL);

SELECT * FROM KIT;

```

Results	Explain	Describe	Saved SQL	History
KIT_ID	HOME_KIT_INFO	AWAY_KIT_INFO	CLUB_ID	A_ID
1101	YELLOW	GREEN	1251	1251
1151	BLUE	ORANGE	1301	1301
1201	BLACK	RED	1351	1351
1251	WHITE	PURPLE	1401	1401
1301	BLACK	ORANGE	1451	1451

5 rows returned in 0.00 seconds [CSV Export](#)

FOR PLAYER:

```

INSERT INTO PLAYER VALUES (PLAYER_ID_SEQ.NEXTVAL, 'MASHRAFI MORTAZA', '05-10-1983',
'01-09-2019', 'BOWLER', '0451856165', '0145865412', '018529652', '80000');

INSERT INTO PLAYER VALUES (PLAYER_ID_SEQ.NEXTVAL, 'LITON DAS', '13-10-1994', '03-08-
2019', 'AL-ROUNDER','08645189', '05648966', '06542986', '65000');

INSERT INTO PLAYER VALUES (PLAYER_ID_SEQ.NEXTVAL, 'MUSTAFIZUR RAHMAN','06-09-1995',
'06-12-2019', 'BOWLER', '0546298652', '01896528', '058642962', '70000');

INSERT INTO PLAYER VALUES (PLAYER_ID_SEQ.NEXTVAL, 'TAMIM IQBAL', '20-03-1989','03-02-
2019', 'BATSMAN', '0188965926', '05412965', '08456296', '80000');

INSERT INTO PLAYER VALUES (PLAYER_ID_SEQ.NEXTVAL, 'MAHMUDULLAH', '04-02-1986', '15-
02-2019', 'AL-ROUNDER','086451256', '012845626', '01496526', '80000');

SELECT * FROM PLAYER;

```

Results	Explain	Describe	Saved SQL	History				
PLAYER_ID	PLAYER NAME	DATE OF BIRTH	REGISTRATION DATE	SKILLS	PLAYER_PHONE NUMBER 1	PLAYER_PHONE NUMBER 2	PLAYER_PHONE NUMBER 3	PLAYER SALARY
1001	MASHRAFI MORTAZA	05-10-1983	01-09-2019	BOWLER	0451856165	045863412	0129652	80000
1002	LITON DAS	13-10-1994	03-08-2019	AL-ROUNDER	08645189	05648966	06542986	65000
1003	MUSTAFIZUR RAHMAN	06-09-1995	06-12-2019	BOWLER	0546298652	01896528	058642962	70000
1004	TAMIM IQBAL	20-03-1989	03-02-2019	BATSMAN	0188965926	05412965	08456296	80000
1005	MAHMUDULLAH	04-02-1986	15-02-2019	AL-ROUNDER	086451256	012845626	01496526	80000

5 rows returned in 0.00 seconds [CSV Export](#)

FOR MEDICAL STUFF:

```
INSERT INTO MEDICAL_STUFF VALUES (MEDICAL_S_ID_SEQ.NEXTVAL, 'DR. ABIR', '084758415',  
'0586542085', '05418566', '30000');
```

```
INSERT INTO MEDICAL_STUFF VALUES (MEDICAL_S_ID_SEQ.NEXTVAL, 'DR. RAFIQ', '06584289',  
'08654156', '089541896', '35000');
```

```
INSERT INTO MEDICAL_STUFF VALUES (MEDICAL_S_ID_SEQ.NEXTVAL, 'DR. RAHMAN',  
'0896541', '018542916', '08415646', '30000');
```

```
INSERT INTO MEDICAL_STUFF VALUES (MEDICAL_S_ID_SEQ.NEXTVAL, 'DR. IQBAL',  
'0564512856', '0486596', '094615665', '30000');
```

```
INSERT INTO MEDICAL_STUFF VALUES (MEDICAL_S_ID_SEQ.NEXTVAL, 'DR. AMIN', '09529652',  
'08496542', '08418965', '25000');
```

```
SELECT * FROM MEDICAL_STUFF ;
```

Results	Explore	Describe	Save SQL	History	
MEDICAL_S_ID	MEDICAL_S_NAME	MEDICAL_S_PHONE_NUMBER_1	MEDICAL_S_PHONE_NUMBER_2	MEDICAL_S_PHONE_NUMBER_3	MEDICAL_S_SALARY
1001	DR. ABIR	0576870	08042806	0470046	30000
1002	DR. RAFIQ	0663206	0804118	0964286	35000
1003	DR. RAHMAN	090141	18542916	0410046	30000
1004	DR. IQBAL	05412856	0486596	0400000	30000
1005	DR. AMIN	0029042	0400042	0410046	25000

3 rows returned in 0.00 seconds

FOR STADIUM:

```
INSERT INTO STADIUM VALUES (STADIUM_ID_SEQ.NEXTVAL, 'MIRPUR NATIONAL STADIUM');
```

```
INSERT INTO STADIUM VALUES (STADIUM_ID_SEQ.NEXTVAL, 'SAHID A.H.M. STADIUM');
```

```
INSERT INTO STADIUM VALUES (STADIUM_ID_SEQ.NEXTVAL, 'RANGPUR STADIUM');
```

```
INSERT INTO STADIUM VALUES (STADIUM_ID_SEQ.NEXTVAL, 'BARISAL SOINIK STADIUM');
```

```
INSERT INTO STADIUM VALUES (STADIUM_ID_SEQ.NEXTVAL, 'SHADID ABDUR RAB STADIUM');
```

```
SELECT * FROM STADIUM;
```

Results Explain Describe Saved SQL History

STADIUM_ID	STADIUM_NAME
1001	MIRPUR NATIONAL STADIUM
1051	SAHID A.H.M. STADIUM
1101	RANGPUR STADIUM
1151	BARISAL SOINIK STADIUM
1201	SHADID ABDUR RAB STADIUM

5 rows returned in 0.00 seconds

[CSV Export](#)

FOR STADIUM ADDRESS:

```
INSERT INTO STADIUM_ADDRESS VALUES (S_ID_SEQ.NEXTVAL, 'DHAKA', 'BANGLADESH');  
INSERT INTO STADIUM_ADDRESS VALUES (S_ID_SEQ.NEXTVAL, 'RAJSHAHI', 'BANGLADESH');  
INSERT INTO STADIUM_ADDRESS VALUES (S_ID_SEQ.NEXTVAL, 'RANGPUR', 'BANGLADESH');  
INSERT INTO STADIUM_ADDRESS VALUES (S_ID_SEQ.NEXTVAL, 'BARISAL', 'BANGLADESH');  
INSERT INTO STADIUM_ADDRESS VALUES (S_ID_SEQ.NEXTVAL, 'CUMILLA', 'BANGLADESH');  
SELECT * FROM STADIUM_ADDRESS;
```

Results Explain Describe Saved SQL History

S_ID	STADIUM_CITY	STADIUM_COUNTRY
1001	DHAKA	BANGLADESH
1051	RAJSHAHI	BANGLADESH
1101	RANGPUR	BANGLADESH
1151	BARISAL	BANGLADESH
1201	CUMILLA	BANGLADESH

5 rows returned in 0.00 seconds

[CSV Export](#)

FOR GROUND STUFF:

```
INSERT INTO GROUND_STUFF VALUES (GROUND_S_ID_SEQ.NEXTVAL, 'ABIR', '08956165',  
'07854145', '084155', '15000', '1151', '1201');  
  
INSERT INTO GROUND_STUFF VALUES (GROUND_S_ID_SEQ.NEXTVAL, 'RAFIQ', '05841866',  
'06526526', '069854156', '15000', '1201', '1251');
```

```
INSERT INTO GROUND_STUFF VALUES (GROUND_S_ID_SEQ.NEXTVAL, 'RAHMAN', '023528564',
'03542152', '0318562', '15000', '1101', '1051');
```

```
INSERT INTO GROUND_STUFF VALUES (GROUND_S_ID_SEQ.NEXTVAL, 'IQBAL', '0874596',
'08976596', '0549641', '15000', '1101', '1151');
```

```
INSERT INTO GROUND_STUFF VALUES (GROUND_S_ID_SEQ.NEXTVAL, 'AMIN', '089742963',
'07526320', '0496542065', '15000', '1151', '1201');
```

```
SELECT * FROM GROUND_STUFF ;
```

GROUND_S_ID	GROUND_S_NAME	GROUND_S_PHONE_NUMBER_1	GROUND_S_PHONE_NUMBER_2	GROUND_S_PHONE_NUMBER_3	GROUND_S_SALARY	S_ID	STADIUM_ID
1001	ASBI	0958783	1054145	84135	15000	1101	1201
1051	BAFIG	0641896	6629526	88954196	15000	1201	1251
1151	RAHMAN	23528564	3642152	318562	15000	1101	1051
1401	IQBAL	874596	8879596	549641	15000	1101	1151
1451	AMIN	89742963	7526320	496542065	15000	1151	1201

5 rows returned in 0.00 seconds

FOR COACH:

```
INSERT INTO COACH VALUES (COACH_ID_SEQ.NEXTVAL, 'KHALED MAHMUD', 'AL_ROUNDNER',
'khaled@gmail.com', '0785412', '05419655', '0185496', '50000');
```

```
INSERT INTO COACH VALUES (COACH_ID_SEQ.NEXTVAL, 'OWISH SAHA', 'BATTING',
'owish@gmail.com', '08562966', '054196526', '065296352', '50000');
```

```
INSERT INTO COACH VALUES (COACH_ID_SEQ.NEXTVAL, 'MARK O DONNELL', 'BATTING',
'mark@gmail.com', '8418541', '9749652', '0564521096', '60000');
```

```
INSERT INTO COACH VALUES (COACH_ID_SEQ.NEXTVAL, 'OTTIS GIBSON', 'AL_ROUNDNER',
'ottis@gmail.com', '024965269', '03251854', '085741896', '50000');
```

```
INSERT INTO COACH VALUES (COACH_ID_SEQ.NEXTVAL, 'DAV WHATMORE', 'AL_ROUNDNER',
'dav@gmail.com', '08541895632', '01856296', '04896526', '60000');
```

```
SELECT * FROM COACH;
```

COACH_ID	COACH_NAME	TYPE	EMAIL	COACH_PHONE_NUMBER_1	COACH_PHONE_NUMBER_2	COACH_PHONE_NUMBER_3	COACH_SALARY
1001	KHALED MAHMUD	AL_ROUNDNER	khaled@gmail.com	785412	5419655	185496	50000
1051	OWISH SAHA	BATTING	owish@gmail.com	8542966	54196526	65296352	50000
1101	MARK O DONNELL	BATTING	mark@gmail.com	8418541	9749652	564521096	60000
1151	OTTIS GIBSON	AL_ROUNDNER	ottis@gmail.com	24965269	3251854	85741896	50000
1201	DAV WHATMORE	AL_ROUNDNER	dav@gmail.com	8541895632	1856296	4896526	60000

5 rows returned in 0.00 seconds

FOR FOREIGN KEY OWNER PLAYER:

```
INSERT INTO FOREIGN_KEY_OWNER_PLAYER VALUES ('1051', '1001');
```



```

INSERT INTO FOREIGN_KEY_OWNER_PLAYER VALUES ('1101', '1051');
INSERT INTO FOREIGN_KEY_OWNER_PLAYER VALUES ('1251', '1101');
INSERT INTO FOREIGN_KEY_OWNER_PLAYER VALUES ('1301', '1151');
INSERT INTO FOREIGN_KEY_OWNER_PLAYER VALUES ('1351', '1251');
SELECT * FROM FOREIGN_KEY_OWNER_PLAYER;

```

Results	Explain	Describe	Saved SQL	History
OWNER_ID		PLAYER_ID		
1051		1001		
1101		1051		
1251		1101		
1301		1151		
1351		1251		

5 rows returned in 0.00 seconds [CSV Export](#)

FOR FOREIGN KEY PLAYER MEDICAL STUFF:

```

INSERT INTO FOREIGN_KEY_PLAYER_MS VALUES ('1001', '1001');
INSERT INTO FOREIGN_KEY_PLAYER_MS VALUES ('1051', '1051');
INSERT INTO FOREIGN_KEY_PLAYER_MS VALUES ('1101', '1101');
INSERT INTO FOREIGN_KEY_PLAYER_MS VALUES ('1151', '1151');
INSERT INTO FOREIGN_KEY_PLAYER_MS VALUES ('1251', '1201');
SELECT * FROM FOREIGN_KEY_PLAYER_MS;

```

Results Explain Describe Saved SQL History

PLAYER_ID	MEDICAL_S_ID
1001	1001
1051	1051
1101	1101
1151	1151
1251	1201

5 rows returned in 0.02 seconds

[CSV Export](#)

FOR FOREIGN KEY OWNER STADIUM:

```
INSERT INTO FOREIGN_KEY_OWNER_STADIUM VALUES ('1051', '1001');
```

```
INSERT INTO FOREIGN_KEY_OWNER_STADIUM VALUES ('1101', '1051');
```

```
INSERT INTO FOREIGN_KEY_OWNER_STADIUM VALUES ('1251', '1101');
```

```
INSERT INTO FOREIGN_KEY_OWNER_STADIUM VALUES ('1301', '1151');
```

```
INSERT INTO FOREIGN_KEY_OWNER_STADIUM VALUES ('1351', '1201');
```

```
SELECT * FROM FOREIGN_KEY_OWNER_STADIUM;
```

Results Explain Describe Saved SQL History

OWNER_ID	STADIUM_ID
1051	1001
1101	1051
1251	1101
1301	1151
1351	1201

5 rows returned in 0.00 seconds

[CSV Export](#)

FOR FOREIGN KEY STADIUM GROUND STUFF:

```
INSERT INTO FOREIGN_KEY_STADIUM_GS VALUES ('1001', '1001');
```

```
INSERT INTO FOREIGN_KEY_STADIUM_GS VALUES ('1051', '1051');
```

```
INSERT INTO FOREIGN_KEY_STADIUM_GS VALUES ('1101', '1351');
```

```
INSERT INTO FOREIGN_KEY_STADIUM_GS VALUES ('1151', '1401');
```

```
INSERT INTO FOREIGN_KEY_STADIUM_GS VALUES ('1201', '1451');
```

```
SELECT * FROM FOREIGN_KEY_OWNER_STADIUM;
```

Results Explain Describe Saved SQL History

OWNER_ID	STADIUM_ID
1051	1001
1101	1051
1251	1101
1301	1151
1351	1201

5 rows returned in 0.00 seconds

[CSV Export](#)

FOR FOREIGN KEY MANAGER COACH:

```
INSERT INTO FOREIGN_KEY_MANAGER_COACH VALUES ('1001', '1001');
```

```
INSERT INTO FOREIGN_KEY_MANAGER_COACH VALUES ('1051', '1051');
```

```
INSERT INTO FOREIGN_KEY_MANAGER_COACH VALUES ('1101', '1101');
```

```
INSERT INTO FOREIGN_KEY_MANAGER_COACH VALUES ('1151', '1151');
```

```
INSERT INTO FOREIGN_KEY_MANAGER_COACH VALUES ('1201', '1201');
```

```
SELECT * FROM FOREIGN KEY MANAGER COACH;
```

Results

Explain

Describe

Saved SQL

History

MANAGER_ID	COACH_ID
1001	1001
1051	1051
1101	1101
1151	1151
1201	1201

5 rows returned in 0.00 seconds

CSV Export

QUERY WRITING:

01. Display the player names who earn more than LITON DAS.

```
SELECT PLAYER_NAME FROM PLAYER WHERE PLAYER_SALARY > (SELECT PLAYER_SALARY FROM  
PLAYER WHERE PLAYER_NAME='LITON DAS');
```

Results	Explain	Describe	Saved SQL	History
PLAYER_NAME				
MASHRAFI MORTAZA				
MUSTAFIZUR RAHMAN				
TAMIM IQBAL				
MAHMUDULLAH				
4 rows returned in 0.00 seconds				
CSV Export				

02. Display the manager names who joined after FARHAN.

```
SELECT * FROM MANAGER WHERE JOIN_DATE < (SELECT JOIN_DATE FROM MANAGER WHERE  
MANAGER_NAME = 'FARHAN');
```

Results	Explain	Describe	Saved SQL	History		
MANAGER_ID	MANAGER_NAME	JOIN DATE	MANAGER_PHONE_NUMBER_1	MANAGER_PHONE_NUMBER_2	MANAGER_PHONE_NUMBER_3	MANAGER_SALARY
1101	SADI	01-06-2020	80054296	541096526	1875496	8000
1151	MUSTAFA	03-02-2018	8520952	56290635	88645296	45000
1201	NAFI	09-08-2021	85481265	56412566	56429652	50000
1251	NAFI	09-08-2021	85481265	56412566	56429652	50000
4 rows returned in 0.00 seconds						
CSV Export						

03. Display the name of owners who is from BANGLADESH.

```
SELECT OWNER.OWNER_NAME, STADIUM_ADDRESS.STADIUM_COUNTRY FROM OWNER,  
STADIUM_ADDRESS WHERE OWNER.S_ID = STADIUM_ADDRESS.S_ID AND  
STADIUM_ADDRESS.STADIUM_COUNTRY = 'BANGLADESH';
```

Results Explain Describe Saved SQL History

OWNER_NAME	STADIUM_COUNTRY
BEXIMCO GROUP	BANGLADESH
BANGLADESH CRICKET BOARD	BANGLADESH
BANGLADESH CRICKET BOARD	BANGLADESH
MUSTAFA RAFIQUUL ISLAM	BANGLADESH
NAFISA KAMAL	BANGLADESH

5 rows returned in 0.00 seconds CSV Export

04. Display the home kit info of all clubs.

```
SELECT KIT.HOME_KIT_INFO, CLUB.CLUB_NAME FROM KIT, CLUB WHERE KIT.KIT_ID = CLUB.CLUB_ID;
```

Results	Explain	Describe	Saved SQL	History
HOME_KIT_INFO CLUB_NAME				
YELLOW	RANGPUR RANGERS			
BLUE	CUMILLA WARRIORS			
BLACK	BARISHAL BULLS			

3 rows returned in 0.01 seconds [CSV Export](#)

05. Find out the skills of a player with maximum no of average salary.

```
SELECT SKILLS FROM PLAYER WHERE PLAYER_ID=(SELECT PLAYER_ID FROM PLAYER GROUP BY PLAYER_ID HAVING AVG(PLAYER_SALARY) = (SELECT MAX(AVG(PLAYER_SALARY))FROM PLAYER GROUP BY PLAYER_ID));
```

Results	Explain	Describe	Saved SQL	History
SKILLS				
AL-ROUNDER				

1 rows returned in 0.01 seconds [CSV Export](#)

06. Find out the coach which salary is 40000 to 50000.

```
SELECT * FROM COACH WHERE COACH_SALARY BETWEEN 40000 AND 50000;
```

Results Explain Describe Saved SQL History

COACH_ID	COACH_NAME	TYPE	EMAIL	COACH_PHONE_NUMBER_1	COACH_PHONE_NUMBER_2	COACH_PHONE_NUMBER_3	COACH_SALARY
1001	KHALED MAHMUD	AL-ROUNDER	khaled@gmail.com	785412	5419555	105496	\$6000
1051	OWISH SAHA	BATTING	owish@gmail.com	8562966	54195526	45294352	\$6000
1151	OTTIS GIBSON	AL-ROUNDER	otth@gmail.com	24565269	3251854	85741894	\$6000

3 rows returned in 0.00 seconds

CSV Export

07. Find the player who got second height salary

```
SELECT MAX(PLAYER_SALARY) FROM PLAYER WHERE PLAYER_SALARY < (SELECT MAX(PLAYER_SALARY) FROM PLAYER);
```

Results	Explain	Describe	Saved SQL	History
<div>MAX(PLOYER_SALARY)</div> <div>80000</div>				
1 rows returned in 0.00 seconds CSV Export				

08. Find first and last record of club.

```
SELECT * FROM CLUB WHERE CLUB_ID = (SELECT MIN(CLUB_ID) FROM CLUB) OR CLUB_ID =
(SELECT MAX(CLUB_ID) FROM CLUB);
```

Results Explain Describe Saved SQL History

CLUB_ID	CLUB_NAME	DATE_OF_ESTABLISHED	CLUB_PHONE_NUMBER_1	CLUB_PHONE_NUMBER_2	CLUB_PHONE_NUMBER_3
1001	DHAKA PLATOON	2015	94652641	58412656	179654166
1201	BARISHAL BULLS	2012	87486642	9886464	1796554486

2 rows returned in 0.00 seconds

CSV Export

VIEW:

01. Create a view called Club_View based on Club_Name and Club_ID from the Club Table

```
CREATE OR REPLACE VIEW CLUB_VIEW AS SELECT CLUB_NAME, CLUB_ID FROM CLUB;
```

```
SELECT * FROM CLUB_VIEW;
```

User: SCOTT

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE VIEW CLUB_VIEW AS SELECT CLUB_NAME, CLUB_ID FROM CLUB;
SELECT * FROM CLUB_VIEW;
```

Results Explain Describe Saved SQL History

View created.

0.13 seconds

Results Explain Describe Saved SQL History

CLUB_NAME	CLUB_ID
DHAKA PLATOON	1001
RAJSHAHI ROYELS	1051
RANGPUR RANGERS	1101
CUMILLA WARRIORS	1151
BARISHAL BULLS	1201

5 rows returned in 0.00 seconds

[CSV Export](#)

02. Create a view to show the name and type of coach id 1201.

```
CREATE OR REPLACE VIEW COACH_VIEW AS SELECT COACH_NAME, TYPE FROM COACH WHERE
COACH_ID=1201;
```

```
SELECT * FROM COACH_VIEW;
```

ORACLE Database Express Edition

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE VIEW COACH_VIEW AS SELECT COACH_NAME, TYPE FROM COACH WHERE COACH_ID=1201;  
SELECT * FROM COACH_VIEW;
```

Results Explain Describe Saved SQL History

View created.

0.11 seconds

Results Explain Describe Saved SQL History

COACH_NAME	TYPE
DAV WHATMORE	AL_ROUNDNER

1 rows returned in 0.00 seconds

[CSV Export](#)

03. Create a view to show the name, id and salary of player who got salary more than 50000

```
CREATE OR REPLACE VIEW PLAYER_VIEW AS SELECT PLAYER_NAME, PLAYER_ID,  
PLAYER_SALARY FROM PLAYER WHERE PLAYER_SALARY>50000;
```

```
SELECT * FROM PLAYER_VIEW;
```


User: SCOTT

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE VIEW PLAYER_VIEW
AS
SELECT PLAYER_NAME, PLAYER_ID, PLAYER_SALARY FROM PLAYER WHERE PLAYER_SALARY>50000;

SELECT * FROM PLAYER_VIEW;
```

Results Explain Describe Saved SQL History

View created.

0.00 seconds

Results Explain Describe Saved SQL History

PLAYER_NAME	PLAYER_ID	PLAYER_SALARY
MASHRAFI MORTAZA	1001	80000
LITON DAS	1051	65000
MUSTAFIZUR RAHMAN	1101	70000
TAMIM IQBAL	1151	80000
MAHMUDULLAH	1251	80000

5 rows returned in 0.00 seconds [CSV Export](#)

04. Display the CLUB all information along with their respective EMAIL.

```
CREATE OR REPLACE VIEW OWNER_CLUB_VIEW AS

SELECT C.*,O.EMAIL

FROM CLUB C,OWNER O WHERE C.CLUB_ID=O.CLUB_ID;

SELECT * FROM OWNER_CLUB_VIEW;
```

User: SCOTT

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE VIEW OWNER_CLUB_VIEW AS
SELECT C.*,O.EMAIL
FROM CLUB C,OWNER O WHERE C.CLUB_ID=O.CLUB_ID;
```

```
SELECT * FROM OWNER_CLUB_VIEW;
```

Results Explain Describe Saved SQL History

View created.

0.49 seconds

Results Explain Describe Saved SQL History

CLUB_ID	CLUB_NAME	DATE_OF_ESTABLISHED	CLUB_PHONE_NUMBER_1	CLUB_PHONE_NUMBER_2	CLUB_PHONE_NUMBER_3	EMAIL
1001	DHAKA PLATOON	2015	94652641	58412658	175654166	beximca@gmail.com
1051	RAJSHAHI ROYELS	2012	9545463	1785621265	1941535126	bcb@gmail.com
1201	BARISHAL BULLS	2012	87486542	906646975	1796554406	bcb@gmail.com
1101	RANGPUR RANGERS	2012	54308523	241574521	1815461265	naftaa@gmail.com

4 rows returned in 0.00 seconds

CSV Export

PROCEDURE AND FUNCTION:

PROCEDURE:

01. Write a procedure that will show number of player.

CREATE OR REPLACE PROCEDURE PLAYERSHOW

```

AS
P NUMBER;
BEGIN
SELECT COUNT(*) INTO P FROM PLAYER;
DBMS_OUTPUT.PUT_LINE('TOTAL PLAYER IS: ' || P);
END;

```

ORACLE Database Express Edition

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▼

```

CREATE OR REPLACE PROCEDURE PLAYERSHOW
AS
P NUMBER;
BEGIN
SELECT COUNT(*) INTO P FROM PLAYER;
DBMS_OUTPUT.PUT_LINE('TOTAL PLAYER IS: ' || P);
END;

```

Results Explain Describe Saved SQL History

Procedure created.

0.00 seconds

02. Write a procedure that will show most senior player

```

CREATE OR REPLACE PROCEDURE SENIOR_PLAYER AS

```

```

P VARCHAR(50);

BEGIN

SELECT PLAYER_NAME INTO P FROM PLAYER WHERE REGISTRATION_DATE=(SELECT
MIN(REGISTRATION_DATE)FROM PLAYER);

DBMS_OUTPUT.PUT_LINE('MOST SENIOR PLAYER IS: '||P);

END;

```



03. Write a procedure that will take as parameter COACH_ID and show coach email.

```

CREATE OR REPLACE PROCEDURE EMAIL_COACH(CID IN NUMBER) AS
E VARCHAR(50);

BEGIN

SELECT EMAIL INTO E FROM COACH WHERE COACH_ID=CID;

DBMS_OUTPUT.PUT_LINE(CID||'|' COACH EMAIL IS: '||E);

END;

```

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE PROCEDURE EMAIL_COACH(CID IN NUMBER) AS  
E VARCHAR(50);  
BEGIN  
SELECT EMAIL INTO E FROM COACH WHERE COACH_ID=CID;  
DBMS_OUTPUT.PUT_LINE(CID||' COACH EMAIL IS: '||E);  
END;
```

Results Explain Describe Saved SQL History

Procedure created.

0.00 seconds

FUNCTION:

04. Create a function that returns the total number of KITS.

CREATE OR REPLACE FUNCTION TOTALKIT

RETURN NUMBER IS

TOTAL NUMBER := 0;

BEGIN

SELECT count(*) INTO TOTAL FROM KIT;

RETURN TOTAL;

```
END;
```

```
DECLARE
```

```
B NUMBER;
```

```
BEGIN
```

```
B := TOTALKIT();
```

```
DBMS_OUTPUT.PUT_LINE('TOTAL NUMBER OF KIT: ' || B);
```

```
END;
```

ORACLE® Database Express Edition

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display ▼

```
CREATE OR REPLACE FUNCTION TOTALKIT
RETURN NUMBER IS
TOTAL NUMBER := 0;
BEGIN
SELECT count(*) INTO TOTAL FROM KIT;
RETURN TOTAL;
END;

DECLARE
B NUMBER;
BEGIN
B := TOTALKIT();
DBMS_OUTPUT.PUT_LINE('TOTAL NUMBER OF KIT: ' || B);
END;
```

Results Explain Describe Saved SQL History

Function created.

0.00 seconds

Results	Explain	Describe	Saved SQL	History
---------	---------	----------	-----------	---------

TOTAL NUMBER OF KIT: 5

Statement processed.

0.01 seconds

05. Create a function that returns the total number of GROUND STUFF in club.

```
CREATE OR REPLACE FUNCTION GROUND_STUFF_FUN
```

```
RETURN NUMBER IS
```

```
TOTAL NUMBER:= 0;
```

```
BEGIN
```

```
SELECT COUNT(*) INTO TOTAL FROM GROUND_STUFF;
```

```
RETURN TOTAL;
```

```
END;
```

```
DECLARE
```

```
G NUMBER;
```

```
BEGIN
```

```
G := GROUND_STUFF_FUN();
```

```
DBMS_OUTPUT.PUT_LINE('TOTAL NUMBER OF GROUND STUFF: ' || G);
```

```
END;
```

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE FUNCTION GROUND_STUFF_FUN
RETURN NUMBER IS
TOTAL NUMBER:= 0;
BEGIN
SELECT COUNT(*) INTO TOTAL FROM GROUND_STUFF;
RETURN TOTAL;
END;

DECLARE
G NUMBER;
BEGIN
G := GROUND_STUFF_FUN();
DBMS_OUTPUT.PUT_LINE('TOTAL NUMBER OF GROUND STUFF: ' || G);
END;
```

Results Explain Describe Saved SQL History

Function created.

0.22 seconds

Results Explain Describe Saved SQL History

TOTAL NUMBER OF GROUND STUFF: 5

Statement processed.

0.02 seconds

06. Create a function that returns the total number of stadium for club.

CREATE OR REPLACE FUNCTION STADIUM_TOTAL

RETURN NUMBER IS

TOTAL NUMBER:= 0;


```
BEGIN  
SELECT COUNT(*) INTO TOTAL FROM STADIUM;  
RETURN TOTAL;  
END;  
  
DECLARE  
S NUMBER;  
BEGIN  
S := STADIUM_TOTAL();  
DBMS_OUTPUT.PUT_LINE('THE TOTAL NUMBER OF STADIUM IS: ' || S);  
END;
```

User: SCOTT

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE FUNCTION STADIUM_TOTAL
RETURN NUMBER IS
TOTAL NUMBER:= 0;
BEGIN
SELECT COUNT(*) INTO TOTAL FROM STADIUM;
RETURN TOTAL;
END;

DECLARE
S NUMBER;
BEGIN
S := STADIUM_TOTAL();
DBMS_OUTPUT.PUT_LINE('THE TOTAL NUMBER OF STADIUM IS: ' || S);
END;
```

Results Explain Describe Saved SQL History

Function created.

0.06 seconds

Results Explain Describe Saved SQL History

THE TOTAL NUMBER OF STADIUM IS: 5

Statement processed.

0.00 seconds

TRIGGER:

01. Create a trigger if insert in manager will print added.

```
CREATE OR REPLACE TRIGGER MANAGER_T  
AFTER INSERT ON MANAGER  
FOR EACH ROW  
BEGIN  
DBMS_OUTPUT.PUT_LINE('MANAGER IS ADDED');  
END;
```

```
INSERT INTO MANAGER VALUES(1301, 'RAFI', '12-1-2020', 0541855418, 08741895346,  
057418954696, 60000);
```

ORACLE Database Express Edition

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display ▼

```
CREATE OR REPLACE TRIGGER MANAGER_T  
AFTER INSERT ON MANAGER  
FOR EACH ROW  
BEGIN  
DBMS_OUTPUT.PUT_LINE('MANAGER IS ADDED');  
END;
```

Results Explain Describe Saved SQL History

Trigger created.

0.03 seconds

Results	Explain	Describe	Saved SQL	History
---------	---------	----------	-----------	---------

MANAGER IS ADDED

1 row(s) inserted.

0.01 seconds

02. Create a trigger if update in manager will print updated.

```
CREATE OR REPLACE TRIGGER MANAGER_T
```

```
AFTER UPDATE ON MANAGER
```

```
FOR EACH ROW
```

```
BEGIN
```

```
DBMS_OUTPUT.PUT_LINE('MANAGER INFORMATION UPDATEED SUCCESSFULLY');
```

```
END;
```

```
UPDATE MANAGER SET MANAGER_SALARY=55000 WHERE MANAGER_ID=1001 ;
```

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▾

```
CREATE OR REPLACE TRIGGER MANAGER_T
AFTER UPDATE ON MANAGER
FOR EACH ROW
BEGIN
  DBMS_OUTPUT.PUT_LINE('MANAGER INFORMATION UPDATEED SUCCESSFULLY');
END;
```

Results Explain Describe Saved SQL History

Trigger created.

0.15 seconds

Results Explain Describe Saved SQL History

MANAGER INFORMATION UPDATEED SUCCESSFULLY

1 row(s) updated.

1.11 seconds

03. Create a trigger if insert in STADIUM will print added.

```
CREATE OR REPLACE TRIGGER STADIUM_TRIGGER
AFTER INSERT ON STADIUM FOR EACH ROW
BEGIN
```

```
DBMS_OUTPUT.PUT_LINE('NEW STADIUM ADDED');  
END;
```

```
INSERT INTO STADIUM VALUES(1251, 'BARIDHARA STADIUM');
```

ORACLE Database Express Edition

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE TRIGGER STADIUM_TRIGGER  
AFTER INSERT ON STADIUM FOR EACH ROW  
BEGIN  
DBMS_OUTPUT.PUT_LINE('NEW STADIUM ADDED');  
END;
```



```
INSERT INTO STADIUM VALUES(1251, 'BARIDHARA STADIUM');
```

Results Explain Describe Saved SQL History

Trigger created.

0.28 seconds

Results Explain Describe Saved SQL History


```
NEW STADIUM ADDED
```



```
1 row(s) inserted.
```


0.15 seconds

04. Write triggers that checks that COACH_SALARY does not become negative.

```
CREATE OR REPLACE TRIGGER COACH_TRIGGER
BEFORE UPDATE OR INSERT ON COACH FOR EACH ROW
BEGIN
IF :NEW.COACH_SALARY <0 THEN
raise_application_error(-20202,'SALARY IS NOT NEGATIVE');
END IF;
END;
```

ORACLE Database Express Edition

User: SCOTT

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▼

```
CREATE OR REPLACE TRIGGER COACH_TRIGGER
BEFORE UPDATE OR INSERT ON COACH FOR EACH ROW
BEGIN
IF :NEW.COACH_SALARY <0 THEN
raise_application_error(-20202,'SALARY IS NOT NEGATIVE');
END IF;
END;
```

Results Explain Describe Saved SQL History

Trigger created.

0.00 seconds

CONCLUSION :

We have shown all the queries to create the tables in 'oracle 10g'. We had shown the queries to insert the values and took their screen-shots. The Admin panel is created and the interface is properly communicated with the oracle database. We have successfully done the procedure and function with the help of plsql. In this system, the trigger is also implemented and the package is applied.

IN THE FUTURE :

This job of ours can help a Cricket club in their data storing systems. Also We Create Users For Easy Our Work. And like the club, we want to make the admin interface for every table so that it can be easy to add, update, and delete data.