



IE2042 - Database Management Systems for Security


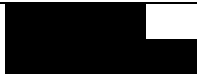


Assignment 01: 2020 Regular Intake

Title: *Students Record Keeping System*

Group Members:

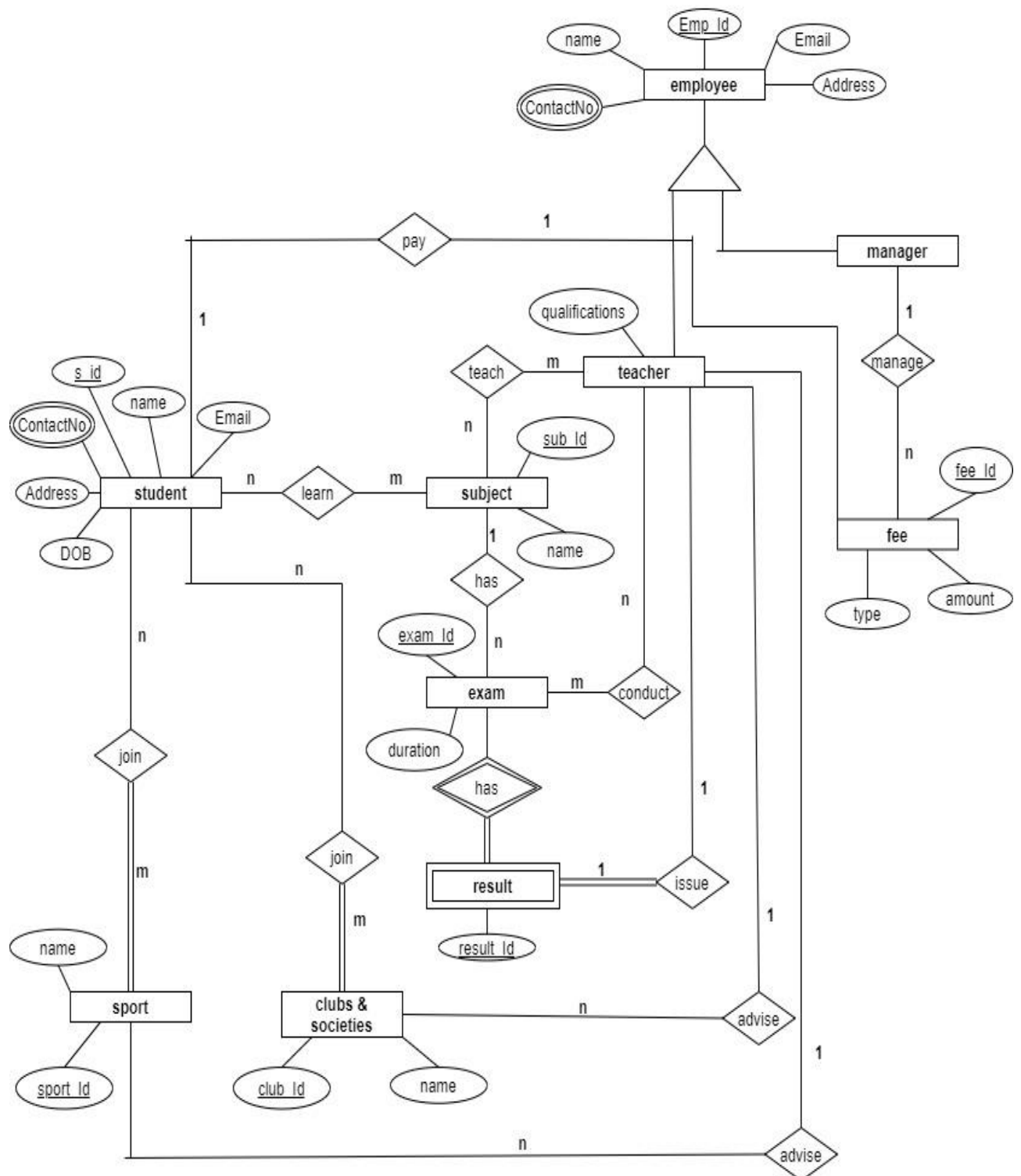
IT Number	Name
	Senarathna K. M. G. S. B

Contribution for the Project

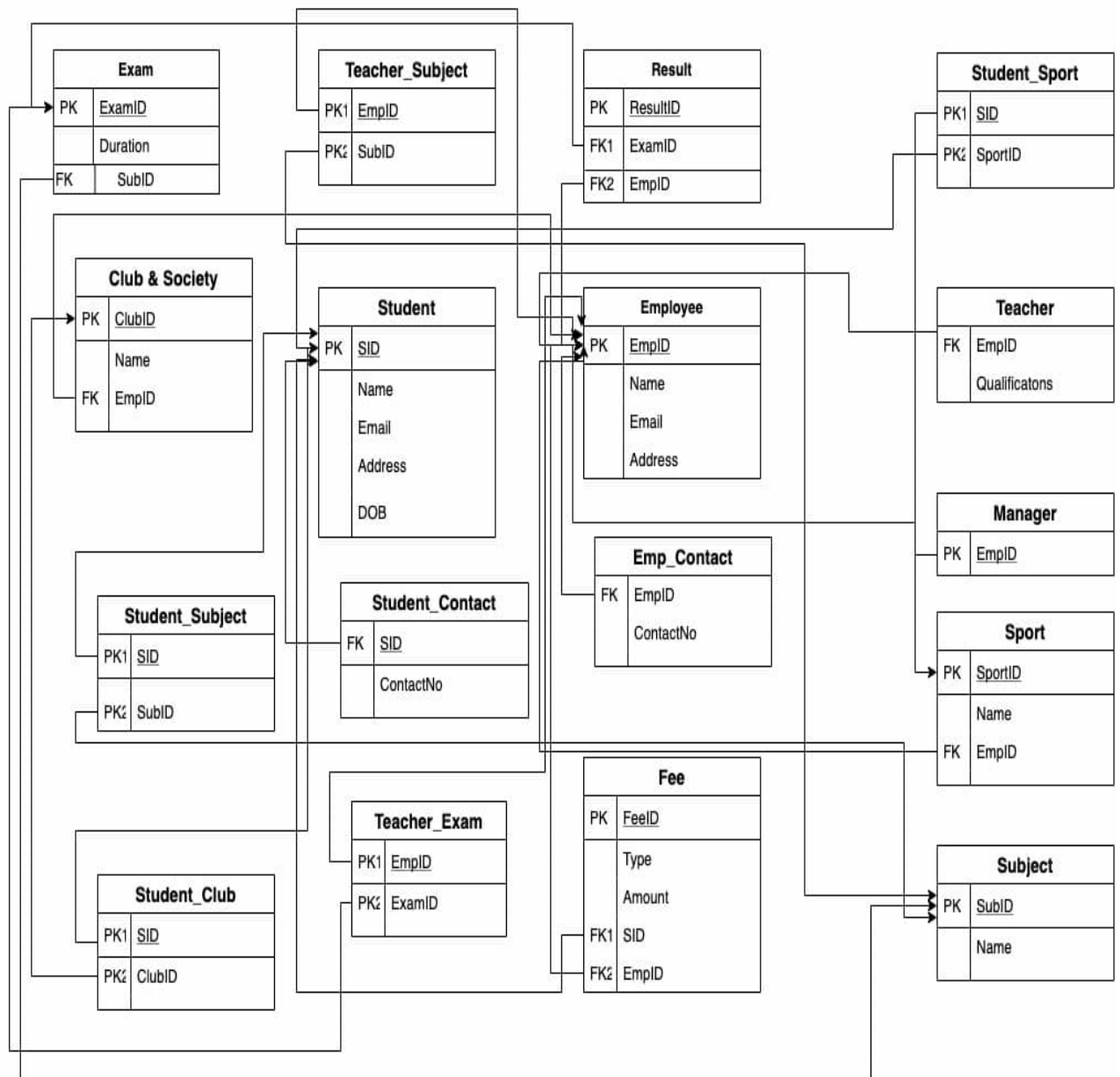
IT Number	Contribution
 Senarathna K.M.G.S. B	<ul style="list-style-type: none">• Implementing database security features (Access control privileges, Creation of views, Recovery mechanism)
	<ul style="list-style-type: none">• Designing relational model type 1• Designing relational model type 2• Drafting report of the project
	<ul style="list-style-type: none">• Creation of Database (Tables, inserting sample data, Applying table relations)
	<ul style="list-style-type: none">• Designing ER diagram• Drafting report of the project

ER Diagram

ER diagram -Students record keeping system



Relational Model (Type 1)



Relational Model (Type 2)

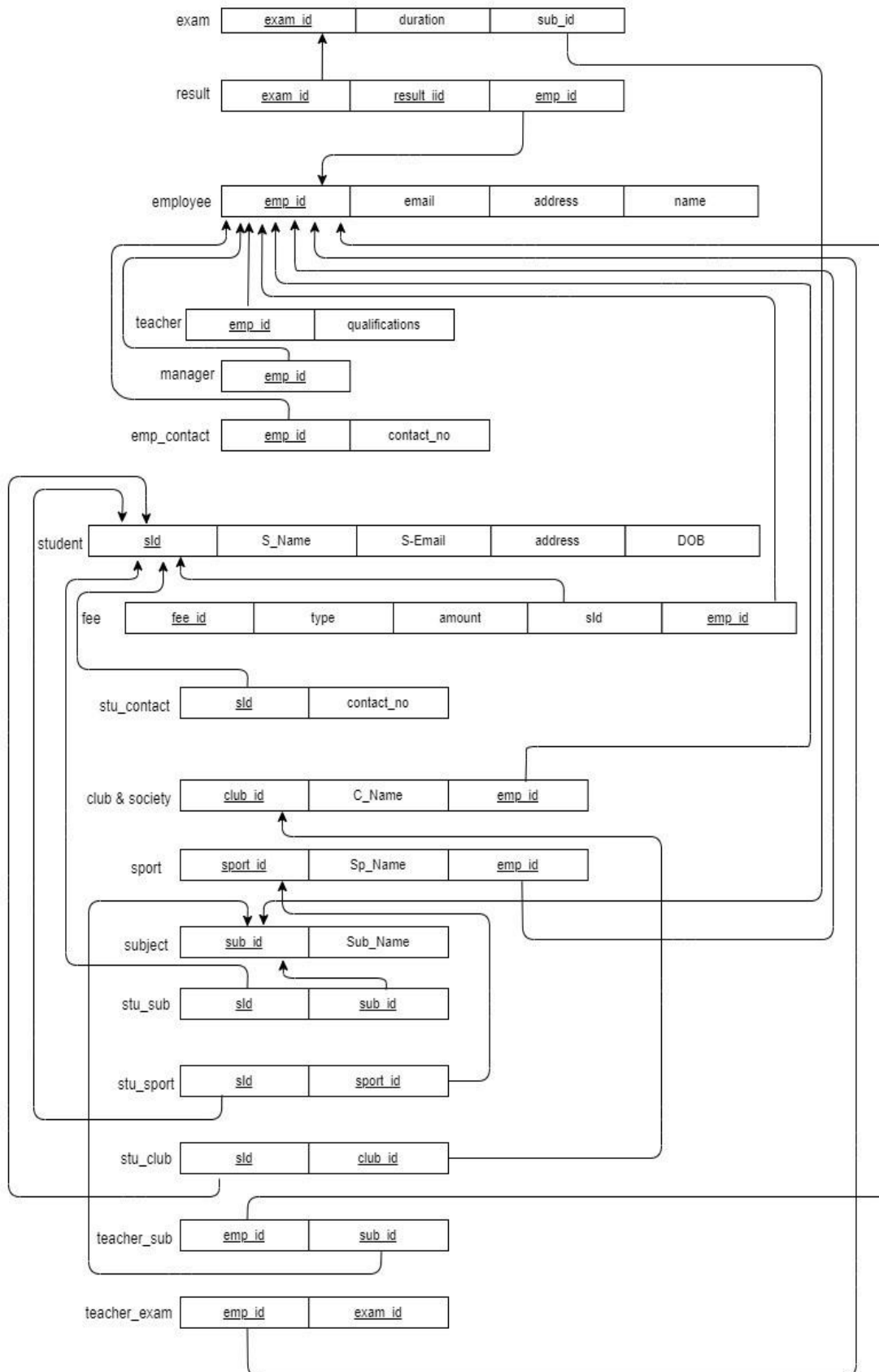


Table creation queries

- Creation query of **Employee** table.

```
--Creating Employee table
CREATE TABLE Employee
(
    Emp_ID CHAR(20) PRIMARY KEY,
    Name VARCHAR(100),
    Email CHAR(200),
    Post_Code CHAR(20),
    Lane VARCHAR(200),
    Town VARCHAR(200)
);
```

- Creation query of **Result** table.

```
--creating Result table
CREATE TABLE Result
(
    Exam_ID CHAR(20), --FK
    Result_ID CHAR(20) PRIMARY KEY,
    Emp_ID CHAR(20), --FK
    CONSTRAINT fk_Exam_Result FOREIGN KEY(Exam_ID) REFERENCES Exam(Exam_ID) ON DELETE CASCADE,
    CONSTRAINT fk_Employee_Result FOREIGN KEY(Emp_ID) REFERENCES Employee(Emp_ID) ON DELETE CASCADE
);
```

- Creation query of **Teacher_Sub** table.

```
--Creating Teacher_Sub table
CREATE TABLE Teacher_Sub
(
    Emp_ID CHAR(20), --FK
    Sub_ID CHAR(20), --FK
    CONSTRAINT fk_Employee_Teacher_Sub FOREIGN KEY(Emp_ID) REFERENCES Employee(Emp_ID) ON DELETE CASCADE,
    CONSTRAINT fk_Subject_Teacher_Sub FOREIGN KEY(Sub_ID) REFERENCES Subject(Sub_ID) ON DELETE CASCADE
);
```

Tables

- Student table

❖ S_ID	❖ S_NAME	❖ S_EMAIL	❖ S_LANE	❖ S_TOWN	❖ DOB
1 IT1911	Sachintha Senarathna	sac@gmail.com	Kandy rd	Kadawatha	10.04.98
2 IT1912	Rashmika Ekanayake	rash@gmail.com	Kuru rd	Wattala	03.12.98
3 IT1913	Thanuri Pamalka	thanu@gmail.com	Sama rd	Malabe	10.01.99
4 IT1914	Dilki Hearth	dilki@gmail.com	Weliwita rd	Kiribathgoda	10.04.98
5 IT1915	Ranul Deelaka	ranu@gmail.com	Lakshman rd	Kaduwela	21.04.98
6 IT1916	Chiranth Rukshan	chira@gmail.com	Police rd	Nugegoda	31.12.98
7 IT1917	Yasas Deelaka	yasa@gmail.com	Gemunu rd	Rajagiriya	21.05.98
8 IT1918	Yasith Dissanayake	yasi@gmail.com	Sudarshana rd	Weliwita	02.06.98
9 IT1919	Vibatha Devasinghe	vib@gmail.com	Kandy rd	Kiribathgoda	26.08.98
10 IT1920	Salila Hearth	sali@gmail.com	Dave rd	Biyagama	15.09.98

- Result table

	❖ EXAM_ID	❖ RESULT_ID	❖ EMP_ID
1	AB001	RE1	E001
2	AC002	RE2	E002
3	AD003	RE3	E003
4	AG006	RE6	E006
5	AH007	RE7	E007
6	AI008	RE8	E008
7	AJ009	RE9	E009
8	AK001	RE10	E010

- Exam table

	❖ EXAM_ID	❖ DURATION	❖ SUB_ID
1	AB001		2 IT1010
2	AC002		1 BM1020
3	AD003		3 EN1030
4	AE005		1 BM1040
5	AF002		2 IT1050
6	AG006		2 IT1010
7	AH007		1 BM1020
8	AI008		3 EN1030
9	AJ009		1 BM1040
10	AK001		2 IT1050

- Teacher table

	EMP_ID	QUALIFICATION
1	E001	GRADE A
2	E002	GRADE A
3	E003	GRADE B
4	E004	GRADE C
5	E005	GRADE B
6	E006	GRADE A
7	E007	GRADE A
8	E008	GRADE B
9	E009	GRADE B
10	E010	GRADE C

- Manager table

	EMP_ID
1	E001
2	E002
3	E003
4	E004
5	E005
6	E006
7	E007
8	E008
9	E009
10	E010

- Emp_Contact table

	EMP_ID	CONTACT_NO
1	E001	774517345
2	E002	715678321
3	E003	764567856
4	E004	719806784
5	E005	761236789
6	E006	715639629
7	E007	712349870
8	E008	765567894
9	E009	778512585
10	E010	764599323

- Employee table

	EMP_ID	NAME	EMAIL
1	E001	Sathsara Bandara	sath@gmail.com
2	E002	Devinda Senadeera	devi@gmail.com
3	E003	Nilu Adikari	nil@gmail.com
4	E004	Chathu Hearth	chathu@gmail.com
5	E005	Ishani Rathnayake	ish@gmail.com
6	E006	Sathsarani Ekanayake	sathi@gmail.com
7	E007	Ashara Karunathilake	ash@gmail.com
8	E008	Thanuja Senadhipathi	tha@gmail.com
9	E009	Danuri Alwis	dan@gmail.com
10	E010	Pawani Omesha	paw@gmail.com

- Fee table

	FEE_ID	TYPE	AMOUNT	S_ID	EMP_ID
1	F50	C	5500	IT1911	E001
2	F51	B	6000	IT1912	E002
3	F52	E	6500	IT1913	E003
4	F53	F	7000	IT1914	E004
5	F54	G	8000	IT1915	E005
6	F55	H	8500	IT1916	E006
7	F56	I	9000	IT1917	E007
8	F57	J	9500	IT1918	E008
9	F58	K	4000	IT1919	E009
10	F59	L	4500	IT1920	E010

- Stu_contact table

	⚡ S_ID	⚡ CONTACT_NO
1	IT1911	711250028
2	IT1912	764558970
3	IT1913	773421634
4	IT1914	765794512
5	IT1915	776869841
6	IT1916	762662899
7	IT1917	768907882
8	IT1918	779906784
9	IT1919	767899092
10	IT1920	711224364

- Club_Society table

	⚡ CLUB_ID	⚡ C_NAME	⚡ EMP_ID
1	G10	GC	E001
2	H20	HC	E002
3	I30	IC	E003
4	J40	JS	E004
5	K50	KC	E005
6	L60	LC	E006
7	M70	MS	E007
8	N80	NS	E008
9	O90	OC	E009
10	P55	PS	E010

- Sport table

	⚡ SPORT_ID	⚡ SP_NAME	⚡ EMP_ID
1	C1	Cricket	E001
2	R2	Rugby	E002
3	H3	Hockey	E003
4	TT4	Table Tennis	E004
5	T5	Tennis	E005
6	S6	Swimming	E006
7	C7	Carrom	E007
8	C8	Cycling	E008
9	A9	Archery	E009
10	C10	Chess	E010

- Subject table

	⚡ SUB_ID	⚡ SUB_NAME
1	IT1010	Information Technology 1
2	BM1020	Business Management 1
3	EN1030	English
4	BM1040	Business Management 2
5	IT1050	Information Technology 2

- Stu_Sub table

	⚡ S_ID	⚡ SUB_ID
1	IT1911	IT1010
2	IT1912	IT1050
3	IT1913	BM1020
4	IT1914	EN1030
5	IT1915	IT1010
6	IT1916	BM1040
7	IT1917	IT1050
8	IT1918	EN1030
9	IT1919	BM1020
10	IT1920	BM1040

- Stu_Sport table

	⚡ S_ID	⚡ SPORT_ID
1	IT1911	T5
2	IT1912	S6
3	IT1913	C7
4	IT1914	C8
5	IT1915	A9
6	IT1916	C10
7	IT1917	C1
8	IT1918	R2
9	IT1919	H3
10	IT1920	TT4

- Stu_Club table

	⚡ S_ID	⚡ CLUB_ID
1	IT1911	P55
2	IT1912	O90
3	IT1913	N80
4	IT1914	M70
5	IT1915	L60
6	IT1916	K50
7	IT1917	J40
8	IT1918	I30
9	IT1919	H20
10	IT1920	G10

- Teacher_Sub table

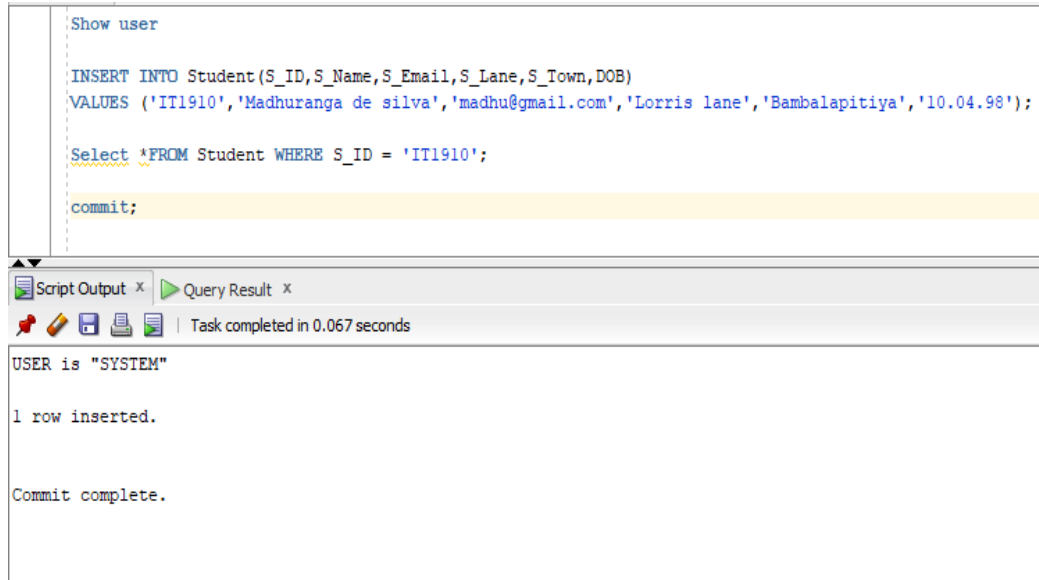
	⚡ EMP_ID	⚡ SUB_ID
1	E001	IT1010
2	E002	BM1020
3	E003	EN1030
4	E004	BM1040
5	E005	IT1050
6	E006	IT1050
7	E007	BM1040
8	E008	EN1030
9	E009	BM1020
10	E010	IT1010

- Teacher_Exam table

	⚡ EMP_ID	⚡ EXAM_ID
1	E001	AB001
2	E002	AC002
3	E003	AD003
4	E006	AG006
5	E007	AH007
6	E008	AI008
7	E009	AJ009
8	E010	AK001

Transactions in the database

1. Inserting details of a new student to the database



```
Show user

INSERT INTO Student(S_ID,S_Name,S_Email,S_Lane,S_Town,DOB)
VALUES ('IT1910','Madhuranga de silva','madhu@gmail.com','Lorris lane','Bambalapitiya','10.04.98');

Select *FROM Student WHERE S_ID = 'IT1910';

commit;
```

Script Output x Query Result x

Task completed in 0.067 seconds

USER is "SYSTEM"

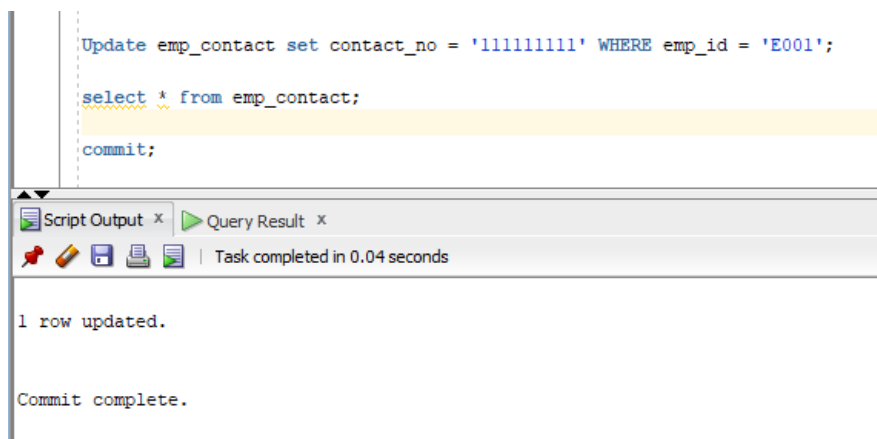
1 row inserted.

Commit complete.

For inserting student details, a user needs administrator privileges. SYSTEM is a user which has been granted all privileges in this database. Steps in this transaction are,

- Inserting new student's details into the table.
- Confirmation of newly added data.
- Commit transaction

2. Updating employee data in Employee table



```
Update emp_contact set contact_no = '111111111' WHERE emp_id = 'E001';

select * from emp_contact;

commit;
```

Script Output x Query Result x

Task completed in 0.04 seconds

1 row updated.

Commit complete.

Steps in this transaction are,

- Checking existing employee data in the table.

EMP_ID	CONTACT_NO
1 E001	774517345
2 E002	715678321
3 E003	764567856
4 E004	719806784
5 E005	761236789
6 E006	715639629
7 E007	712349870
8 E008	765567894
9 E009	778512585
10 E010	764599323

- Update new contact of employee, E001.
- Confirmation of updated contact.

```
select * from emp_contact;
```

EMP_ID	CONTACT_NO
1 E001	111111111
2 E002	715678321
3 E003	764567856
4 E004	719806784
5 E005	761236789
6 E006	715639629
7 E007	712349870
8 E008	765567894
9 E009	778512585
10 E010	764599323

- Commit transaction.

3. Adding a new exam by a Teacher

```
Show user;

Select * from SYSTEM.exam;
SELECT * FROM SYSTEM.subject;

INSERT INTO SYSTEM.exam(Exam_ID,Duration,Sub_ID) VALUES ('Az100',4,'IT1010');

Select * FROM SYSTEM.exam;

commit;
```

Script Output x Query Result x

Task completed in 0.04 seconds

USER is "THANURI"

1 row inserted.

Commit complete.

In this transaction logged on user Thanuri, is a user with privileges of the role Teacher. Because of that the user can insert new exam to the Exam table.

Steps of this transaction are,

- Inserting new exam details to the table.
- Confirming newly added exam details.

```
Select * FROM SYSTEM.exam;
```

EXAM_ID	DURATION	SUB_ID
1 AB001	2	IT1010
2 AC002	1	BM1020
3 AD003	3	EN1030
4 AE005	1	BM1040
5 AF002	2	IT1050
6 AG006	2	IT1010
7 AH007	1	BM1020
8 AI008	3	EN1030
9 AJ009	1	BM1040
10 AK001	2	IT1050
11 AL100	4	IT1010
12 Az100	4	IT1010

- Commit transaction.

Access control privileges

Initially, main actors in the database was identified as Teacher, Student and manager. According to identified actors, roles were created with specific access privileges.

Creation of roles

```
ALTER SESSION SET "_ORACLE_SCRIPT"=TRUE;

CREATE ROLE Teacher;
CREATE ROLE STUDENT;
CREATE ROLE Manager;
```

Creation of user accounts and granting roles to each created account

```
--Creation of users

CREATE USER sachintha IDENTIFIED BY Sac123;

CREATE USER Thanuri IDENTIFIED BY Thanu99;

CREATE USER Rashmika IDENTIFIED BY Mickey98;

CREATE USER Dilki_N IDENTIFIED BY Dilki123;

--Assigning roles to users

GRANT Teacher TO Thanuri;
GRANT STUDENT TO sachintha;
GRANT STUDENT TO Rashmika;
GRANT Manager TO Dilki_N;
```


Granting access privileges to each created role

```
--Granting privileges to role: Manager
GRANT Create session to Manager;
GRANT ALL Privileges TO Manager;

--Granting privileges to role: Teacher
GRANT Create session to Teacher;
GRANT SELECT,INSERT,DELETE ON Exam TO Teacher;
GRANT SELECT,INSERT,DELETE ON Result TO Teacher;
GRANT SELECT,INSERT,DELETE ON Subject TO Teacher;

GRANT SELECT,INSERT,UPDATE ON Club_Society TO Teacher;
GRANT SELECT,INSERT,UPDATE ON Sport TO Teacher;
GRANT SELECT,INSERT,UPDATE ON Stu_Sport TO Teacher;
GRANT SELECT,INSERT,UPDATE ON Stu_Club TO Teacher;

GRANT SELECT ON Stu_Contact TO Teacher;
GRANT SELECT ON Teacher_Sub TO Teacher;
GRANT SELECT ON Teacher_Exam TO Teacher;

--Granting privileges to role: STUDENT
GRANT Create session to STUDENT;
GRANT SELECT ON Result TO STUDENT;
GRANT SELECT ON emp_contact TO STUDENT;
GRANT SELECT ON subject TO STUDENT;
GRANT SELECT ON stu_sub TO STUDENT;
GRANT SELECT ON Sport TO STUDENT;
GRANT SELECT ON club_society TO STUDENT;
```

In order to maintain security of the database, special privilege accesses were used. When granting privileges,

- All privileges were granted to manager since manager is considered as an administrator
- View, updating and delete access were granted to role Teacher for tables Exam, Result and Subject. Only View and update permissions were granted to Teacher on tables Club_Society, Sport, Stu_Sport and Stu_Club because sports and clubs are supervised through the role Teacher.
- Only View permissions for Result, Emp_Contact, Subject, Stu_Sub, Sport and Club_Society tables were given to Students.

Attacks that can happen on the database and countermeasures

(Assuming database relates to a web application)

There are several vulnerabilities can be occurred in a database that is related to a web application. These vulnerabilities can lead to attacks on that database.

Some these attacks that can be targeting a database are,

- Brute force password attacks (attacks which are capable of cracking weak passwords)
- SQL injection attacks
- Privilege abuse
- Theft of backup unencrypted backup tapes.

Countermeasures that can be taken

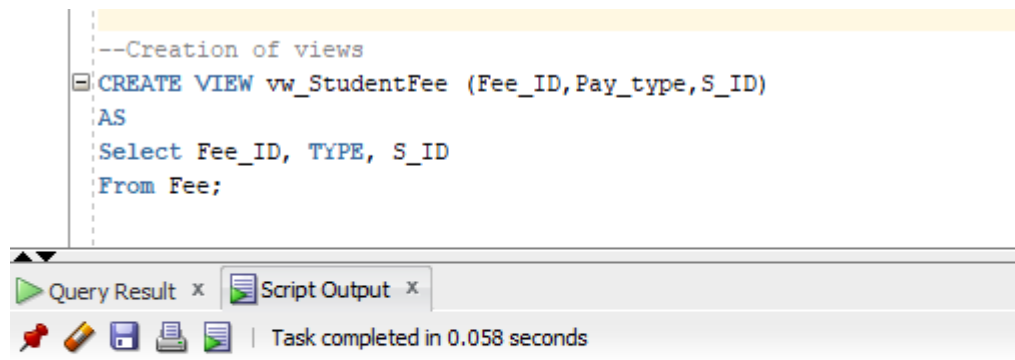
There are few countermeasures that can be implemented in order to overcome these attacks targeting databases.

- Avoiding usage of dynamic queries within web applications and validating user input data is considered as a preventing method of SQL injection attacks.
- Brute force password attacks can be prevented by using strong user authentications credentials inside the database. Using strong password hashing algorithm to store passwords is also can be considered as a countermeasure.
- Usage of access control privileges specified for each user can reduce privilege escalation attacks on the database.
- Encrypting backup data can be used as a countermeasure in order to overcome theft of backup tapes.

Creation of Views

Creation of different views is also used as a security measure in order to encapsulate and to display only selected data through a virtual table.

- A view named **vw_StudentFee** was created in the database to display only the Fee_ID and Payment type belongs to a student.



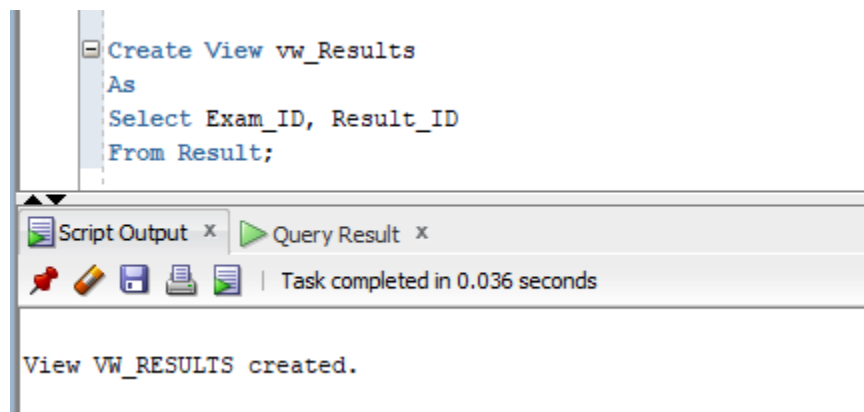
The screenshot shows a SQL script editor with the following text:

```
--Creation of views  
CREATE VIEW vw_StudentFee (Fee_ID,Pay_type,S_ID)  
AS  
Select Fee_ID, TYPE, S_ID  
From Fee;
```

Below the editor, the 'Script Output' tab is active, displaying the message: 'Task completed in 0.058 seconds'.

View VW_STUDENTFEE created.

- A view named vw_Results was created to display results.



The screenshot shows a SQL script editor with the following text:

```
Create View vw_Results  
As  
Select Exam_ID, Result_ID  
From Result;
```

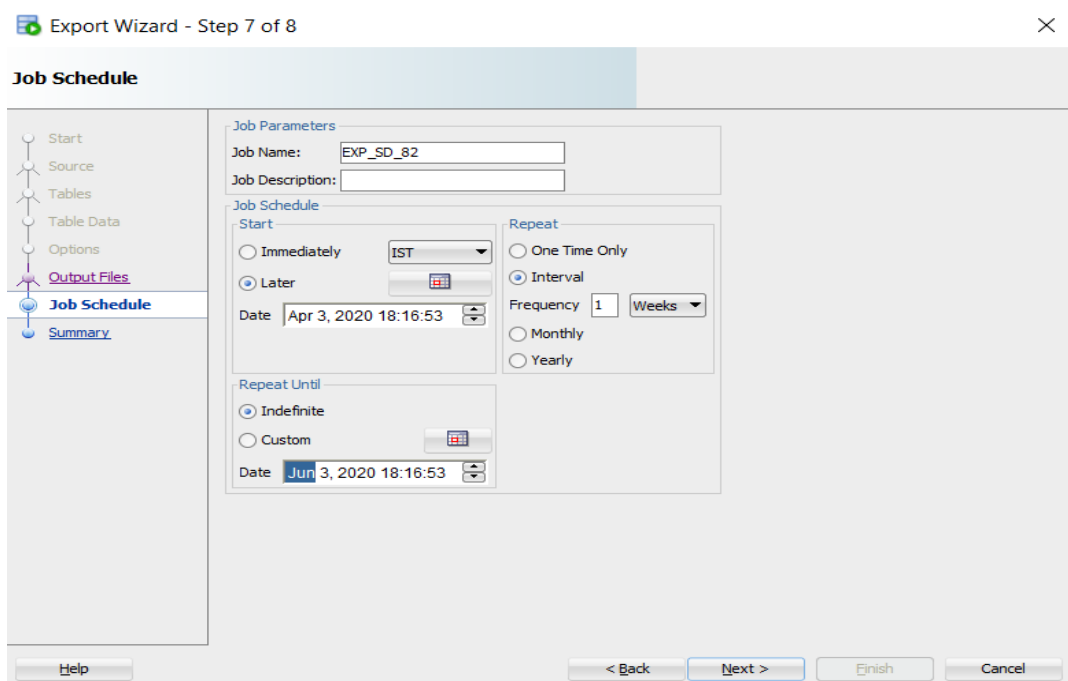
Below the editor, the 'Script Output' tab is active, displaying the message: 'Task completed in 0.036 seconds'.

View VW_RESULTS created.

Database Recovery Mechanisms

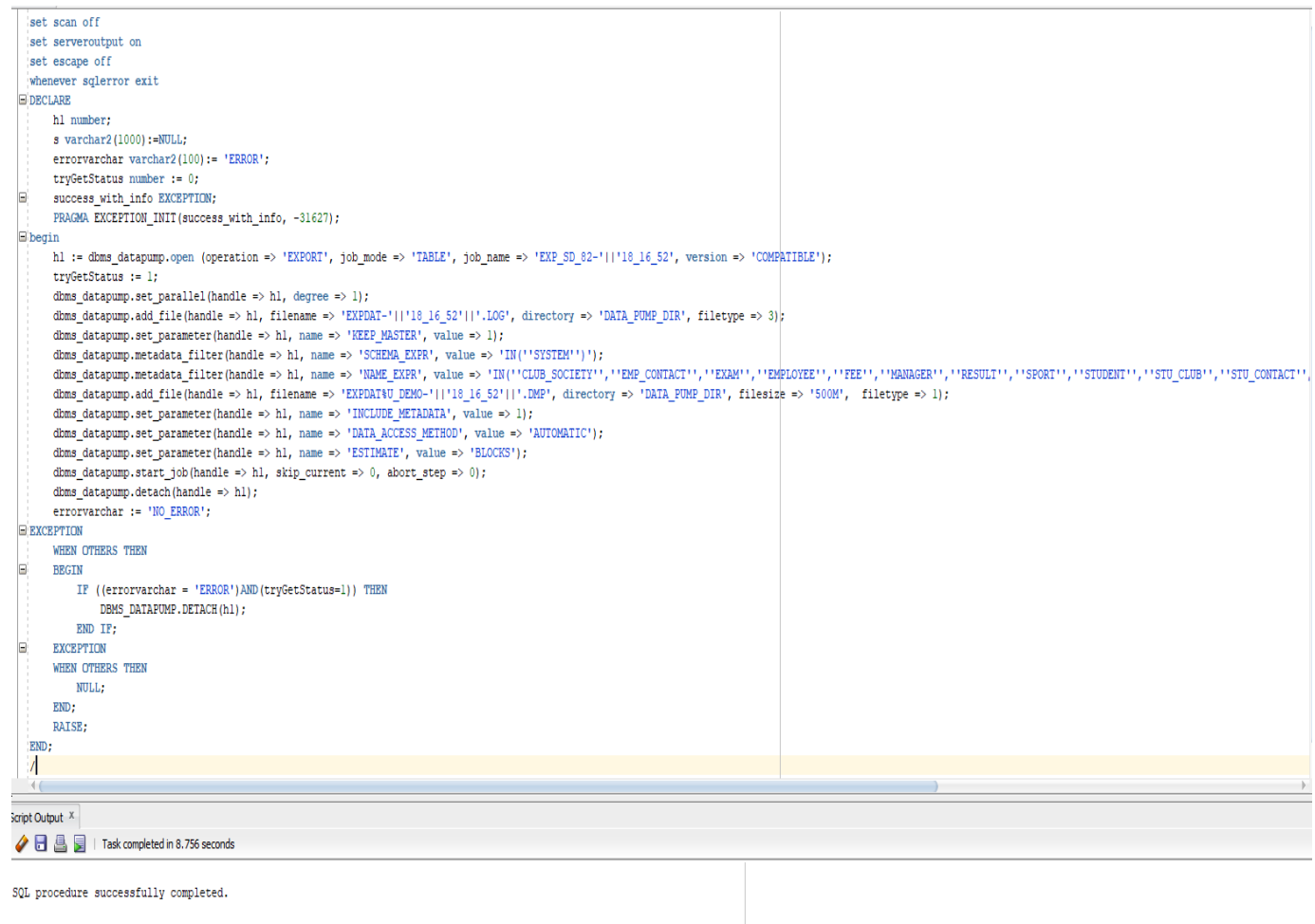
Database recovery mechanism was created for the database to export tables in an interval of one week. Since Student record keeping system is not a critical database, it was assumed that a weekly backup is the most suitable recovery mechanism.

Recovery mechanism was created using Export wizard built in oracle SQL developer.



PL/SQL scripts for this procedure is as included in the next page.

PL/SQL Scripts for recovery mechanism



```
set scan off
set serveroutput on
set escape off
whenever sqlerror exit

DECLARE
    h1 number;
    s varchar2(1000):=NULL;
    errorvarchar varchar2(100):= 'ERROR';
    tryGetStatus number := 0;
    success_with_info EXCEPTION;
    PRAGMA EXCEPTION_INIT(success_with_info, -31627);

BEGIN
    h1 := dms_datapump.open (operation => 'EXPORT', job_mode => 'TABLE', job_name => 'EXP_SD_82-||'18_16_52', version => 'COMPATIBLE');
    tryGetStatus := 1;
    dms_datapump.set_parallel(handle => h1, degree => 1);
    dms_datapump.add_file(handle => h1, filename => 'EXPDAT-||'18_16_52||'.LOG', directory => 'DATA_PUMP_DIR', filetype => 3);
    dms_datapump.set_parameter(handle => h1, name => 'KEEP_MASTER', value => 1);
    dms_datapump.metadata_filter(handle => h1, name => 'SCHEMA_EXPR', value => 'IN(''SYSTEM'')');
    dms_datapump.metadata_filter(handle => h1, name => 'NAME_EXPR', value => 'IN(''CLUB_SOCIETY'', ''EMP_CONTACT'', ''EXAM'', ''EMPLOYEE'', ''FEE'', ''MANAGER'', ''RESULT'', ''SPORT'', ''STUDENT'', ''STU_CLUB'', ''STU_CONTACT'', ''EXPDATU_DEMO-||'18_16_52||'.DMP', directory => 'DATA_PUMP_DIR', filesize => '500M', filetype => 1);
    dms_datapump.set_parameter(handle => h1, name => 'INCLUDE_METADATA', value => 1);
    dms_datapump.set_parameter(handle => h1, name => 'DATA_ACCESS_METHOD', value => 'AUTOMATIC');
    dms_datapump.set_parameter(handle => h1, name => 'ESTIMATE', value => 'BLOCKS');
    dms_datapump.start_job(handle => h1, skip_current => 0, abort_step => 0);
    dms_datapump.detach(handle => h1);
    errorvarchar := 'NO_ERROR';

EXCEPTION
    WHEN OTHERS THEN
        BEGIN
            IF ((errorvarchar = 'ERROR') AND (tryGetStatus=1)) THEN
                DMS_DATAPUMP.DETACH(h1);
            END IF;
        EXCEPTION
            WHEN OTHERS THEN
                NULL;
        END;
        RAISE;
END;
/
```

Script Output x

Task completed in 8.756 seconds

SQL procedure successfully completed.

set scan off

set serveroutput on

set escape off

whenever sqlerror exit

DECLARE

h1 number;

s varchar2(1000):=NULL;

errorvarchar varchar2(100):= 'ERROR';

tryGetStatus number := 0;

success_with_info EXCEPTION;

PRAGMA EXCEPTION_INIT(success_with_info, -31627);

```

begin

    h1 := dbms_datapump.open (operation => 'EXPORT', job_mode => 'TABLE', job_name =>
'EXP_SD_82-'||'18_16_52', version => 'COMPATIBLE');

    tryGetStatus := 1;

    dbms_datapump.set_parallel(handle => h1, degree => 1);

    dbms_datapump.add_file(handle => h1, filename => 'EXPDAT-'||'18_16_52'||'.LOG', directory =>
'DATA_PUMP_DIR', filetype => 3);

    dbms_datapump.set_parameter(handle => h1, name => 'KEEP_MASTER', value => 1);

    dbms_datapump.metadata_filter(handle => h1, name => 'SCHEMA_EXPR', value =>
'IN(''SYSTEM'')');

    dbms_datapump.metadata_filter(handle => h1, name => 'NAME_EXPR', value =>
'IN(''CLUB_SOCIETY'', ''EMP_CONTACT'', ''EXAM'', ''EMPLOYEE'', ''FEE'', ''MANAGER'', ''RESU
LT'', ''SPORT'', ''STUDENT'', ''STU_CLUB'', ''STU_CONTACT'', ''STU_SPORT'', ''STU_SUB'', ''SUBJE
CT'', ''TEACHER'', ''TEACHER_EXAM'', ''TEACHER_SUB'')');

    dbms_datapump.add_file(handle => h1, filename => 'EXPDAT%U_DEMO-'||'18_16_52'||'.DMP',
directory => 'DATA_PUMP_DIR', filesize => '500M', filetype => 1);

    dbms_datapump.set_parameter(handle => h1, name => 'INCLUDE_METADATA', value => 1);

    dbms_datapump.set_parameter(handle => h1, name => 'DATA_ACCESS_METHOD', value =>
'AUTOMATIC');

    dbms_datapump.set_parameter(handle => h1, name => 'ESTIMATE', value => 'BLOCKS');

    dbms_datapump.start_job(handle => h1, skip_current => 0, abort_step => 0);

    dbms_datapump.detach(handle => h1);

    errorvarchar := 'NO_ERROR';

EXCEPTION

    WHEN OTHERS THEN

        BEGIN

            IF ((errorvarchar = 'ERROR')AND(tryGetStatus=1)) THEN

                DBMS_DATAPUMP.DETACH(h1);

            END IF;

        EXCEPTION

            WHEN OTHERS THEN

                NULL.

            END.

            RAISE;

        END;

/

```