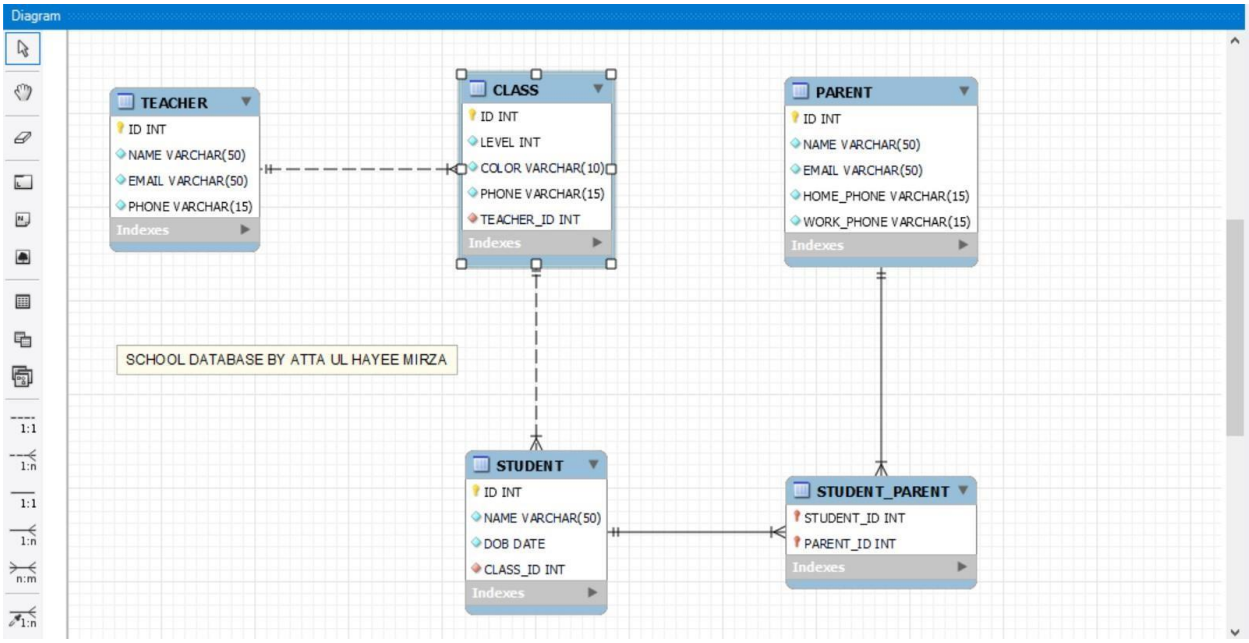
IT 214 DL7

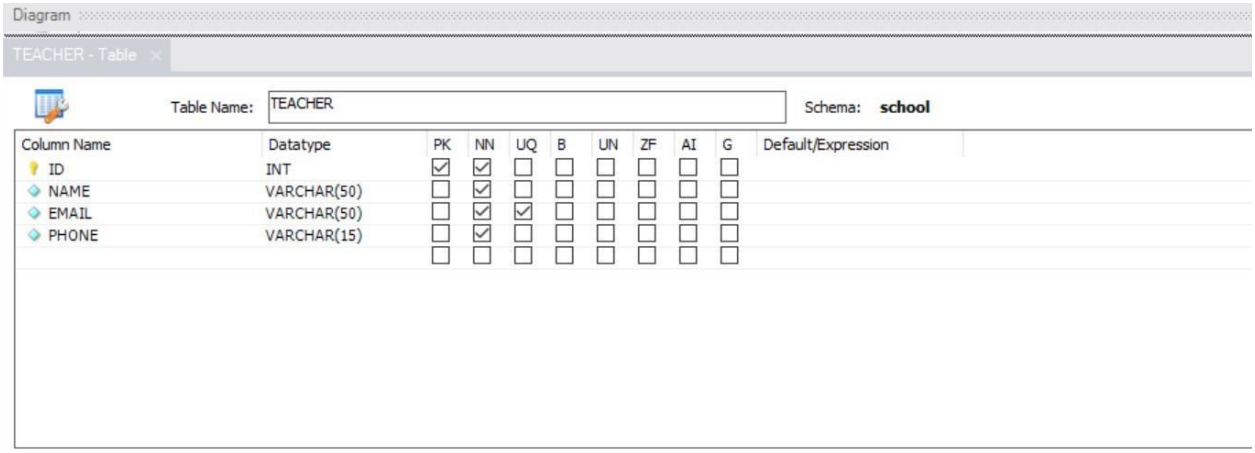
11/20/2023

P3 Project Assignment

Task 1 ERD report

The Entity-Relationship Diagram (ERD) represents a school database with tables for teachers, classes, students, parents, and a join table for student-parent relationships. Relationships are established through foreign key constraints, providing a comprehensive structure for managing school-related data.





Diagram

CLASS - Table

Table Name: CLASS Schema: **school**

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
ID	INT	☑	☑	☐	☐	☐	☐	☐	☐	
LEVEL	INT	☐	☑	☐	☐	☐	☐	☐	☐	
COLOR	VARCHAR(10)	☐	☑	☐	☐	☐	☐	☐	☐	
PHONE	VARCHAR(15)	☐	☑	☐	☐	☐	☐	☐	☐	
TEACHER_ID	INT	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Diagram

STUDENT - Table

Table Name: STUDENT Schema: **school**

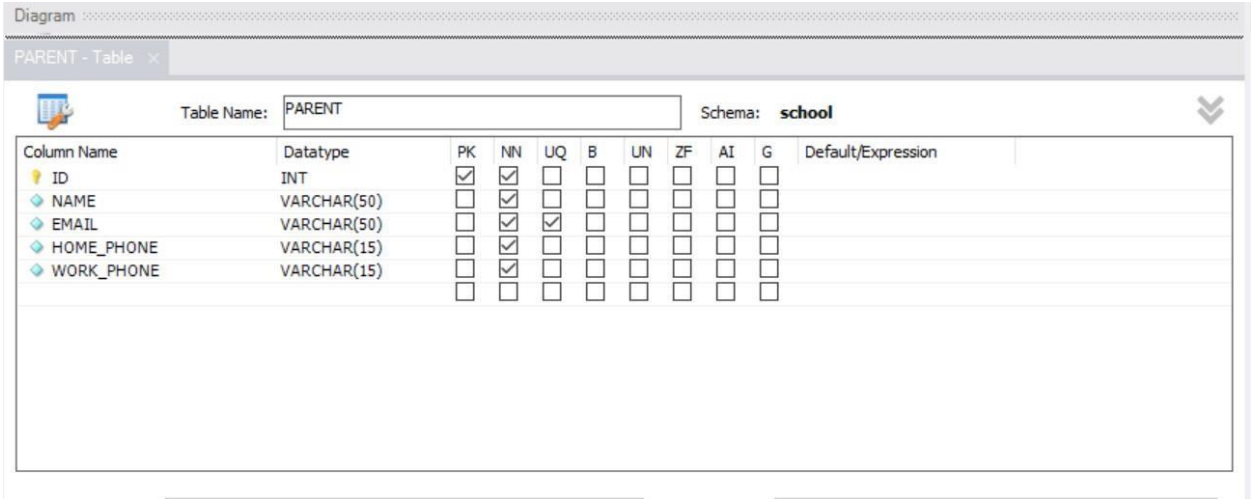
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
ID	INT	☑	☑	☐	☐	☐	☐	☐	☐	
NAME	VARCHAR(50)	☐	☑	☐	☐	☐	☐	☐	☐	
DOB	DATE	☐	☑	☐	☐	☐	☐	☐	☐	
CLASS_ID	INT	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

Diagram

STUDENT_PARENT - Table

Table Name: STUDENT_PARENT Schema: **school**

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
STUDENT_ID	INT	☑	☑	☐	☐	☐	☐	☐	☐	
PARENT_ID	INT	☑	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	



Task 2 - Create database Report

A school database named "school" has been created, encompassing tables for teachers, classes, students, parents, and student-parent relationships. The database schema establishes relationships between entities using primary and foreign key constraints, ensuring a structured foundation for storing and managing school-related information.

```
-- Task 2 - Create database BY AM
DROP DATABASE IF EXISTS SCHOOL;

CREATE DATABASE SCHOOL;

USE SCHOOL;

SHOW DATABASES;
```

Task 3 - Create tables Report

Tables have been successfully created for teachers, classes, students, parents, and student-parent relationships in the "school" database. These tables define the essential attributes and relationships necessary for organizing comprehensive school-related data.

```sql

-- Task 3 - Create tables BY AM
SHOW TABLES;

CREATE TABLE TEACHER (
  ID INT PRIMARY KEY,
  NAME VARCHAR(50) NOT NULL,
  EMAIL VARCHAR(50) UNIQUE NOT NULL,
  PHONE VARCHAR(15) NOT NULL
);

CREATE TABLE CLASS (
  ID INT PRIMARY KEY,
  LEVEL INT NOT NULL,
  COLOR VARCHAR(10) NOT NULL,
  PHONE VARCHAR(15) NOT NULL,
  TEACHER_ID INT NOT NULL,
  FOREIGN KEY (TEACHER_ID) REFERENCES TEACHER(ID)
);
```

```sql

CREATE TABLE STUDENT (
  ID INT PRIMARY KEY,
  NAME VARCHAR(50) NOT NULL,
  DOB DATE NOT NULL,
  CLASS_ID INT NOT NULL,
  FOREIGN KEY (CLASS_ID) REFERENCES CLASS(ID)
);

CREATE TABLE PARENT (
  ID INT PRIMARY KEY,
  NAME VARCHAR(50) NOT NULL,
  EMAIL VARCHAR(50) UNIQUE NOT NULL,
  HOME_PHONE VARCHAR(15) NOT NULL,
  WORK_PHONE VARCHAR(15) NOT NULL
);

```

```sql

CREATE TABLE STUDENT_PARENT (
  STUDENT_ID INT NOT NULL,
  PARENT_ID INT NOT NULL,
  PRIMARY KEY (STUDENT_ID, PARENT_ID),
  FOREIGN KEY (STUDENT_ID) REFERENCES STUDENT(ID),
  FOREIGN KEY (PARENT_ID) REFERENCES PARENT(ID)
);

SHOW TABLES;

-- Show the structure of the created tables
DESCRIBE TEACHER;
DESCRIBE CLASS;
DESCRIBE STUDENT;
DESCRIBE PARENT;
DESCRIBE STUDENT_PARENT;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Tables_in_school

Result 90 | Result 91 | Result 92 | Result 93 | Result 94 | Result 95 | Result 96 | Result 97

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Tables_in_school
class
parent
student
student_parent
teacher

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID	int	NO	PRI	NULL	
NAME	varchar(50)	NO		NULL	
EMAIL	varchar(50)	NO	UNI	NULL	
PHONE	varchar(15)	NO		NULL	

Result 90 | Result 91 | Result 92 | Result 93 | Result 94 | Result 95 | Result 96 | Result 97

Output

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID	int	NO	PRI	NULL	
LEVEL	int	NO		NULL	
COLOR	varchar(10)	NO		NULL	
PHONE	varchar(15)	NO		NULL	
TEACHER_ID	int	NO	MUL	NULL	

Result 90 | Result 91 | Result 92 | Result 93 | Result 94 | Result 95 | Result 96 | Result 97

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID	int	NO	PRI	NULL	
NAME	varchar(50)	NO		NULL	
DOB	date	NO		NULL	
CLASS_ID	int	NO	MUL	NULL	

Result 90 | Result 91 | Result 92 | Result 93 | Result 94 | Result 95 | Result 96 | Result 97

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
ID	int	NO	PRI	NULL	
NAME	varchar(50)	NO		NULL	
EMAIL	varchar(50)	NO	UNI	NULL	
HOME_PHONE	varchar(15)	NO		NULL	
WORK_PHONE	varchar(15)	NO		NULL	

Result 90 | Result 91 | Result 92 | Result 93 | Result 94 | Result 95 | Result 96 | Result 97

Output

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
STUDENT_ID	int	NO	PRI	NULL	
PARENT_ID	int	NO	PRI	NULL	

Result 90 | Result 91 | Result 92 | Result 93 | Result 94 | Result 95 | Result 96 | Result 97

Task 4 - Add data

Data has been inserted into tables for teachers, classes, students, parents, and student-parent relationships in the "school" database. These entries populate the database with information about educators, classes, students, and their respective parents, establishing a foundation for effective data retrieval and analysis.

```sql
-- Task 4 - Add data BY AM
SELECT * FROM TEACHER;
SELECT * FROM CLASS;
SELECT * FROM STUDENT;
SELECT * FROM PARENT;
SELECT * FROM STUDENT_PARENT;

INSERT INTO TEACHER (ID, NAME, EMAIL, PHONE) VALUES
(1, 'Alice Smith', 'alice.smith@school.edu', '555-1111'),
(2, 'Bob Jones', 'bob.jones@school.edu', '555-2222'),
(3, 'Charlie Brown', 'charlie.brown@school.edu', '555-3333'),
(4, 'David Lee', 'david.lee@school.edu', '555-4444'),
(5, 'Emma Watson', 'emma.watson@school.edu', '555-5555');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	NAME	EMAIL	PHONE
1	Alice Smith	alice.smith@school.edu	555-1111
2	Bob Jones	bob.jones@school.edu	555-2222
3	Charlie Brown	charlie.brown@school.edu	555-3333
4	David Lee	david.lee@school.edu	555-4444
5	Emma Watson	emma.watson@school.edu	555-5555
NULL	NULL	NULL	NULL

```sql
INSERT INTO CLASS (ID, LEVEL, COLOR, PHONE, TEACHER_ID) VALUES
(1, 1, 'Blue', '555-1001', 1),
(2, 1, 'Green', '555-1002', 2),
(3, 2, 'Red', '555-2001', 3),
(4, 2, 'Yellow', '555-2002', 4),
(5, 3, 'Purple', '555-3001', 5);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	LEVEL	COLOR	PHONE	TEACHER_ID
1	1	Blue	555-1001	1
2	1	Green	555-1002	2
3	2	Red	555-2001	3
4	2	Yellow	555-2002	4
5	3	Purple	555-3001	5
NULL	NULL	NULL	NULL	NULL

```sql
INSERT INTO STUDENT (ID, NAME, DOB, CLASS_ID) VALUES
(1, 'Anna Lee', '2020-01-01', 1),
(2, 'Ben Smith', '2020-02-02', 1),
(3, 'Cathy Jones', '2020-03-03', 2),
(4, 'Dan Brown', '2020-04-04', 2),
(5, 'Ella Green', '2020-05-05', 3),
(6, 'Frank White', '2020-06-06', 3),
(7, 'Grace Kim', '2020-07-07', 4),
(8, 'Harry Potter', '2020-08-08', 4),
(9, 'Ivy Chen', '2020-09-09', 5),
(10, 'Jack Black', '2020-10-10', 5);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	NAME	DOB	CLASS_ID
1	Anna Lee	2020-01-01	1
2	Ben Smith	2020-02-02	1
3	Cathy Jones	2020-03-03	2
4	Dan Brown	2020-04-04	2
5	Ella Green	2020-05-05	3
6	Frank White	2020-06-06	3
7	Grace Kim	2020-07-07	4
8	Harry Potter	2020-08-08	4
9	Ivy Chen	2020-09-09	5
10	Jack Black	2020-10-10	5
NULL	NULL	NULL	NULL

```sql
INSERT INTO PARENT (ID, NAME, EMAIL, HOME_PHONE, WORK_PHONE) VALUES
(1, 'Adam Lee', 'adam.lee@gmail.com', '555-6001', '555-7001'),
(2, 'Beth Lee', 'beth.lee@gmail.com', '555-6001', '555-7002'),
(3, 'Carl Smith', 'carl.smith@gmail.com', '555-6002', '555-7003'),
(4, 'Dora Smith', 'dora.smith@gmail.com', '555-6002', '555-7004'),
(5, 'Eric Jones', 'eric.jones@gmail.com', '555-6003', '555-7005'),
(6, 'Fiona Jones', 'fiona.jones@gmail.com', '555-6003', '555-7006'),
(7, 'Gary Brown', 'gary.brown@gmail.com', '555-6004', '555-7007'),
(8, 'Helen Brown', 'helen.brown@gmail.com', '555-6004', '555-7008'),
(9, 'Ian Green', 'ian.green@gmail.com', '555-6005', '555-7009'),
(10, 'Jenny Green', 'jenny.green@gmail.com', '555-6005', '555-7010');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ID	NAME	EMAIL	HOME_PHONE	WORK_PHONE
1	Adam Lee	adam.lee@gmail.com	555-6001	555-7001
2	Beth Lee	beth.lee@gmail.com	555-6001	555-7002
3	Carl Smith	carl.smith@gmail.com	555-6002	555-7003
4	Dora Smith	dora.smith@gmail.com	555-6002	555-7004
5	Eric Jones	eric.jones@gmail.com	555-6003	555-7005
6	Fiona Jones	fiona.jones@gmail.com	555-6003	555-7006
7	Gary Brown	gary.brown@gmail.com	555-6004	555-7007
8	Helen Brown	helen.brown@gmail.com	555-6004	555-7008
9	Ian Green	ian.green@gmail.com	555-6005	555-7009
10	Jenny Green	jenny.green@gmail.com	555-6005	555-7010
NULL	NULL	NULL	NULL	NULL

```sql
INSERT INTO STUDENT_PARENT (STUDENT_ID, PARENT_ID) VALUES
(1, 1), (1, 2), (2, 3), (2, 4), (3, 5), (3, 6), (4, 7), (4, 8), (5, 9), (5, 10),
(6, 1), (6, 2), (7, 3), (7, 4), (8, 5), (8, 6), (9, 7), (9, 8), (10, 9), (10, 10);

-- Show all the data in your table
SELECT * FROM TEACHER;
SELECT * FROM CLASS;
SELECT * FROM STUDENT;
SELECT * FROM PARENT;
SELECT * FROM STUDENT_PARENT;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

STUDENT_ID	PARENT_ID
1	1
6	1
1	2
6	2
2	3
7	3
2	4
7	4
3	5
8	5
3	6
8	6
4	7
9	7
4	8
9	8
5	9
10	9
5	10
10	10
NULL	NULL

TEACHER 22 | CLASS 23 | STUDENT 24 | PARENT 25 | STUDENT_PARENT 26 | Apply

Task 5 - Query Data Report:

Structured queries provide insights into the school database, including alphabetical student listings with class details, student rosters per class level and color, parent-student associations, and a comprehensive teacher-parent-student overview. These queries facilitate efficient extraction and presentation of relevant information from the "school" database.

```sql
-- Task 5 - Query the data BY AM
-- Question 1
SELECT S.NAME AS STUDENT,
       CASE
         WHEN C.LEVEL = 1 THEN 'Turtles'
         WHEN C.LEVEL = 2 THEN 'Foxes'
         WHEN C.LEVEL = 3 THEN 'Monkeys'
         ELSE 'Unknown'
       END AS LEVEL,
       C.COLOR AS COLOR,
       C.PHONE AS CLASS_PHONE,
       T.NAME AS TEACHER
FROM STUDENT S
JOIN CLASS C ON S.CLASS_ID = C.ID
JOIN TEACHER T ON C.TEACHER_ID = T.ID
ORDER BY S.NAME;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

STUDENT	LEVEL	COLOR	CLASS_PHONE	TEACHER
Anna Lee	Turtles	Blue	555-1001	Alice Smith
Ben Smith	Turtles	Blue	555-1001	Alice Smith
Cathy Jones	Turtles	Green	555-1002	Bob Jones
Dan Brown	Turtles	Green	555-1002	Bob Jones
Ella Green	Foxes	Red	555-2001	Charlie Brown
Frank White	Foxes	Red	555-2001	Charlie Brown
Grace Kim	Foxes	Yellow	555-2002	David Lee
Harry Potter	Foxes	Yellow	555-2002	David Lee
Ivy Chen	Monkeys	Purple	555-3001	Emma Watson
Jack Black	Monkeys	Purple	555-3001	Emma Watson

```sql
-- Question 2
SELECT CASE
         WHEN C.LEVEL = 1 THEN 'Turtles'
         WHEN C.LEVEL = 2 THEN 'Foxes'
         WHEN C.LEVEL = 3 THEN 'Monkeys'
         ELSE 'Unknown'
       END AS AGE_LEVEL,
       C.COLOR AS COLOR,
       S.NAME AS STUDENT_NAME,
       S.DOB AS DOB
FROM STUDENT S
JOIN CLASS C ON S.CLASS_ID = C.ID
ORDER BY AGE_LEVEL, C.COLOR, S.NAME;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

AGE_LEVEL	COLOR	STUDENT_NAME	DOB
Foxes	Red	Ella Green	2020-05-05
Foxes	Red	Frank White	2020-06-06
Foxes	Yellow	Grace Kim	2020-07-07
Foxes	Yellow	Harry Potter	2020-08-08
Monkeys	Purple	Ivy Chen	2020-09-09
Monkeys	Purple	Jack Black	2020-10-10
Turtles	Blue	Anna Lee	2020-01-01
Turtles	Blue	Ben Smith	2020-02-02
Turtles	Green	Cathy Jones	2020-03-03
Turtles	Green	Dan Brown	2020-04-04

```sql
-- Question 3
SELECT P.NAME AS PARENT,
       P.HOME_PHONE AS HOME_PHONE,
       S.NAME AS STUDENT,
       CASE
         WHEN C.LEVEL = 1 THEN 'Turtles'
         WHEN C.LEVEL = 2 THEN 'Foxes'
         WHEN C.LEVEL = 3 THEN 'Monkeys'
         ELSE 'Unknown'
       END AS AGE_LEVEL,
       C.COLOR AS COLOR
FROM PARENT P
JOIN STUDENT_PARENT SP ON P.ID = SP.PARENT_ID
JOIN STUDENT S ON SP.STUDENT_ID = S.ID
JOIN CLASS C ON S.CLASS_ID = C.ID
ORDER BY P.NAME, S.NAME;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PARENT	HOME_PHONE	STUDENT	AGE_LEVEL	COLOR
Adam Lee	555-6001	Anna Lee	Turtles	Blue
Adam Lee	555-6001	Frank White	Foxes	Red
Beth Lee	555-6001	Anna Lee	Turtles	Blue
Beth Lee	555-6001	Frank White	Foxes	Red
Carl Smith	555-6002	Ben Smith	Turtles	Blue
Carl Smith	555-6002	Grace Kim	Foxes	Yellow
Dora Smith	555-6002	Ben Smith	Turtles	Blue
Dora Smith	555-6002	Grace Kim	Foxes	Yellow
Eric Jones	555-6003	Cathy Jones	Turtles	Green
Eric Jones	555-6003	Harry Potter	Foxes	Yellow
Fiona Jones	555-6003	Cathy Jones	Turtles	Green
Fiona Jones	555-6003	Harry Potter	Foxes	Yellow
Gary Brown	555-6004	Dan Brown	Turtles	Green
Gary Brown	555-6004	Ivy Chen	Monkeys	Purple
Helen Brown	555-6004	Dan Brown	Turtles	Green
Helen Brown	555-6004	Ivy Chen	Monkeys	Purple
Ian Green	555-6005	Ella Green	Foxes	Red
Ian Green	555-6005	Jack Black	Monkeys	Purple
Jenny Green	555-6005	Ella Green	Foxes	Red
Jenny Green	555-6005	Jack Black	Monkeys	Purple

Result 32 | Result 33 | Result 34 | Result 35

```sql
-- Question 4
SELECT T.NAME AS TEACHER, P.NAME AS PARENT, S.NAME AS STUDENT
FROM TEACHER T
JOIN CLASS C ON T.ID = C.TEACHER_ID
JOIN STUDENT S ON C.ID = S.CLASS_ID
JOIN STUDENT_PARENT SP ON S.ID = SP.STUDENT_ID
JOIN PARENT P ON SP.PARENT_ID = P.ID
ORDER BY T.NAME, P.NAME, S.NAME;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TEACHER	PARENT	STUDENT
Alice Smith	Adam Lee	Anna Lee
Alice Smith	Beth Lee	Anna Lee
Alice Smith	Carl Smith	Ben Smith
Alice Smith	Dora Smith	Ben Smith
Bob Jones	Eric Jones	Cathy Jones
Bob Jones	Fiona Jones	Cathy Jones
Bob Jones	Gary Brown	Dan Brown
Bob Jones	Helen Brown	Dan Brown
Charlie Brown	Adam Lee	Frank White
Charlie Brown	Beth Lee	Frank White
Charlie Brown	Ian Green	Ella Green
Charlie Brown	Jenny Green	Ella Green
David Lee	Carl Smith	Grace Kim
David Lee	Dora Smith	Grace Kim
David Lee	Eric Jones	Harry Potter
David Lee	Fiona Jones	Harry Potter
Emma Watson	Gary Brown	Ivy Chen
Emma Watson	Helen Brown	Ivy Chen
Emma Watson	Ian Green	Jack Black
Emma Watson	Jenny Green	Jack Black

CLASS 10 | STUDENT 11 | PARENT 12 | STUDENT_PARENT 13 | TEACHER 14 | CLASS 15 | STUDENT 16 | PARENT 17 | STUDENT_PARENT 18 | Result 19 | Result 20 | Result 21 | Result 2 | Read Only