**AHS School Database - Scenario**

Mrs Quinn, the secretary of the Ashton House School has access to a database that has details about the pupils and parents of the school. She had to create a new database with all the new pupils and parents starting in September 2024.

After populating the tables, had to create a few queries retrieving data ordered by pupils’ last name or their fathers’ last name.

The headteacher asked her to check the attendance of all new pupils when authorised absences are not over five (the maximum they can get) and not more than 9 unauthorised absences as they will have to contact local authorities. For this reason, she used a subquery to locate the pupil that has 9 unauthorised absences and over so she called the parents for a meeting with the Headteacher. Also, she created a view to select all pupils’ full names that have 5 authorised absences or more as they will need to be notified that they could not be granted more absences for the rest of the school year.

Furthermore, Mrs Quinn used DISTINT to call a specific pupil after having to fill out an accident form for him. Also, she had to select the full names and birthdays of the pupils who were currently in the early years class so that she could retrieve the 30-hour code to use for their fees.

After realising that she forgot to add the pupils’ email, she altered the pupils’ table and added a new column so that she could add the emails. But, a mistake happened and Mrs Quinn had to update a specific email. After a particular pupil left the school to relocate to Dubai, she had to delete his details from all tables (attendance, pupils, parents).

By using joins, the secretary was able to retrieve data that existed between the tables of pupils and parents (last name, phone) and between pupils and attendance (pupils’ full name, attendance). Also, she created a report with the max, min, and sum of both authorised and unauthorised absences in the school.

Just before she finished for the day, the Headteacher asked her to add a new parent so she used a stored procedure.

-- AHS School Database - Assignment 3

-- Creating and using a school database

CREATE DATABASE AHS;

USE AHS;

-- Creating tables for pupils, parents and attendance

-- Using constraints (primary key, foreign key, not null, unique, check)

CREATE TABLE PUPILS (

UPN INT NOT NULL,

ID\_PARENTS INT NOT NULL,

FIRST\_NAME VARCHAR(80) NOT NULL,

LAST\_NAME VARCHAR(80) NOT NULL,

DOB DATE NOT NULL,

YEAR\_GROUP VARCHAR(80) NOT NULL,

CONSTRAINT PK\_PUPILS PRIMARY KEY (UPN),

FOREIGN KEY(ID\_PARENTS) REFERENCES PARENTS(ID\_PARENTS)

);

CREATE TABLE PARENTS (

ID\_PARENTS INT NOT NULL,

MOTHER\_FIRST\_NAME VARCHAR(80) NOT NULL,

MOTHER\_LAST\_NAME VARCHAR(80) NOT NULL,

FATHER\_FIRST\_NAME VARCHAR(80) NOT NULL,

FATHER\_LAST\_NAME VARCHAR(80) NOT NULL,

PHONE CHAR(11) NOT NULL,

CONSTRAINT PK\_PARENTS PRIMARY KEY (ID\_PARENTS)

);

CREATE TABLE ATTENDANCE(

ID\_ATTEND INT PRIMARY KEY NOT NULL,

UPN INT NOT NULL,

ID\_PARENTS INT NOT NULL,

AUTHORISED INT NOT NULL,

UNAUTHORISED INT NOT NULL,

FOREIGN KEY(UPN) REFERENCES PUPILS(UPN),

FOREIGN KEY (ID\_PARENTS) REFERENCES PARENTS(ID\_PARENTS)

);

-- Populating our tables

INSERT INTO PUPILS

(UPN, ID\_PARENTS, FIRST\_NAME, LAST\_NAME, DOB, YEAR\_GROUP)

VALUES

(100, 001,'GEORGE', 'WITTON','2020-05-01','EYFS'),

(101, 002,'MATTIAS', 'KLIT', '2017-09-25','YEAR\_1'),

(102, 003,'MARIA', 'HUSTON','2018-06-10','YEAR\_6'),

(103, 004,'SARAH', 'KURAN','2017-08-20','YEAR\_5'),

(104, 005,'PAUL', 'TIMIN','2019-02-02','YEAR\_3'),

(105, 006,'CLARA', 'MANTA','2019-03-30','YEAR\_2'),

(106, 007,'GEORGIA', 'KLAMON','2021-12-25','YEAR\_4'),

(107, 008,'KIM', 'TUMON','2020-04-01','EYFS');

INSERT INTO PARENTS

(ID\_PARENTS,MOTHER\_FIRST\_NAME,MOTHER\_LAST\_NAME,FATHER\_FIRST\_NAME,FATHER\_LAST\_NAME, PHONE)

VALUES

(001,'TARA','LUTON','MIKE','WITTON','07847312950'),

(002,'EMMA','KARON','SOFOKLIS','KLIT','07847312851'),

(003,'ANVI','TIMID','AGAMEMNON','HUSTON','07897322955'),

(004,'ANTONIA','GIONI','AXILEAS','KURAN','07849212940'),

(005,'NEFELI','KLIF','ECTOR','TIMIN','07948313550'),

(006,'ATHINA','ENTON','THOUKIS','MANTA','07997312555'),

(007,'AFRODITA','TRIMIN','CHRISTIAN','KLAMON','07887333950'),

(008,'HIRA','LIMIN','MATTIAS','TUMON','07121312950');

INSERT INTO ATTENDANCE

(ID\_ATTEND,UPN,ID\_PARENTS,AUTHORISED,UNAUTHORISED)

VALUES

(298,100,001,5,6),

(275,101,002,8,2),

(288,102,003,6,0),

(308,103,004,1,6),

(777,104,005,3,4),

(922,105,006,3,9),

(110,106,007,6,7),

(596,107,008,5,5);

-- Retrieving data from the tables

SELECT \* FROM PUPILS ORDER BY LAST\_NAME;

SELECT \* FROM PARENTS ORDER BY FATHER\_LAST\_NAME;

SELECT \* FROM ATTENDANCE WHERE AUTHORISED > 5 AND UNAUTHORISED < 9 ORDER BY UPN;

-- Subquery to retrieve the details of the parents that their child has 9 or more unauthorised absences

SELECT PS.FATHER\_LAST\_NAME, PS.PHONE

FROM PARENTS PS

WHERE PS.ID\_PARENTS IN (SELECT A.ID\_PARENTS

FROM ATTENDANCE A

WHERE A.UNAUTHORISED >= 9);

-- View for retrieving names and phone numbers from parents that their children have more than 5 authorised absences

CREATE VIEW VW\_PARENTS\_EMAILS

AS

SELECT P.FIRST\_NAME, PS.FATHER\_LAST\_NAME, PS.PHONE, A.AUTHORISED

FROM PUPILS P

JOIN PARENTS PS ON P.ID\_PARENTS=PS.ID\_PARENTS

JOIN ATTENDANCE A ON P.UPN=A.UPN

WHERE A.AUTHORISED >=5

WITH CHECK OPTION;

select \* from VW\_PARENTS\_EMAILS;

-- Using DISTINCT to call the pupil's details with UPN=100

SELECT DISTINCT P.FIRST\_NAME, P.LAST\_NAME FROM PUPILS P

WHERE UPN=100

GROUP BY P.LAST\_NAME

ORDER BY P.LAST\_NAME DESC;

-- Using "CONCAT" in built function to call the full name and date of birth for pupils in EYFS

SELECT CONCAT(P.FIRST\_NAME, ' ', P.LAST\_NAME, ' ', P.DOB) AS 'FULL\_NAME\_PUPILS'

FROM PUPILS P

WHERE P.YEAR\_GROUP='EYFS'

ORDER BY P.LAST\_NAME;

-- Alter the Pupils table as the secretary forgot to add the email addresses

-- ALTER TABLE PUPILS

-- ADD EMAIL VARCHAR(90) UNIQUE;

1st way

-- UPDATE PUPILS

-- SET EMAIL= 'GWITTON@AHS.COM' WHERE UPN=100;

-- UPDATE PUPILS

-- SET EMAIL= 'MKLIT@AHS.COM' WHERE UPN=101;

-- UPDATE PUPILS

-- SET EMAIL= 'MHUSTON@AHS.COM' WHERE UPN=102;

-- UPDATE PUPILS

-- SET EMAIL= 'SKURAN@AHS.COM' WHERE UPN=103;

-- UPDATE PUPILS

-- SET EMAIL= 'PTIMIN@AHS.COM' WHERE UPN=104;

-- UPDATE PUPILS

-- SET EMAIL= 'CMANTA@AHS.COM' WHERE UPN=105;

-- UPDATE PUPILS

-- SET EMAIL= 'GKLAMON@AHS.COM' WHERE UPN=106;

-- UPDATE PUPILS

-- SET EMAIL= 'KTUMON@AHS.COM' WHERE UPN=107;

-- 2nd Way

UPDATE PUPILS

SET EMAIL=CASE

WHEN URN=100 THEN'GWITTON@AHS.COM'

WHEN URN=101 THEN'MKLIT@AHS.COM'

WHEN URN=102 THEN'MHUSTON@AHS.COM'

WHEN URN=103 THEN'SKURAN@AHS.COM'

WHEN URN=104 THEN'PTIMIN@AHS.COM'

WHEN URN=105 THEN'CMANTA@AHS.COM'

WHEN URN=106 THEN'GKLAMON@AHS.COM'

WHEN URN=107 THEN'KTUMON@AHS.COM'

END;

-- SELECT \* FROM PUPILS;

-- Update the email column in the pupils table

-- UPDATE PUPILS

-- SET EMAIL = 'KTUM@AHS.COM'

-- WHERE UPN=107;

-- SELECT \* FROM PUPILS;

-- Delete a pupil's details that left the school from all tables

-- but as PUPILS table it is connected with the table ATTENDANCE (foreign key)

-- I will have to delete the pupils details from the ATTENDANCE table first!

-- DELETE FROM ATTENDANCE WHERE UPN=100;

-- DELETE FROM PUPILS P WHERE UPN=100;

-- DELETE FROM PARENTS PS WHERE ID\_PARENTS=001;

-- SELECT \* FROM ATTENDANCE;

-- SELECT \* FROM PUPILS;

-- SELECT \* FROM PARENTS;

-- Inner Join between pupils and parents table

SELECT P.FIRST\_NAME, PS.FATHER\_LAST\_NAME, PS.PHONE FROM PUPILS P INNER JOIN PARENTS PS ON P.ID\_PARENTS=PS.ID\_PARENTS;

-- Left Outer Join between pupils and attendance

SELECT P.FIRST\_NAME, P.LAST\_NAME, A.AUTHORISED, A.UNAUTHORISED FROM PUPILS P LEFT OUTER JOIN ATTENDANCE A ON P.UPN=A.UPN;

-- Aggregate In- built Functions (Query with group by)

SELECT A.UPN,

MAX(A.AUTHORISED) AS 'MAX-AU',

MIN(A.UNAUTHORISED) AS 'MIN-UNAU',

SUM(A.AUTHORISED) AS 'SUM-AU',

SUM(A.UNAUTHORISED) AS 'SUM-UNAU'

FROM ATTENDANCE A

GROUP BY A.UPN

ORDER BY A.UPN DESC;

-- Stored Procedure to insert new parents' details

DELIMITER //

CREATE PROCEDURE INSERT\_NEW\_PARENT(

IN NEW\_ID\_PARENTS INT,

IN NEW\_MOTHER\_FIRST\_NAME VARCHAR(80),

IN NEW\_MOTHER\_LAST\_NAME VARCHAR(80),

IN NEW\_FATHER\_FIRST\_NAME VARCHAR(80),

IN NEW\_FATHER\_LAST\_NAME VARCHAR(80),

IN NEW\_PHONE CHAR(11)

)

BEGIN

INSERT INTO PARENTS(ID\_PARENTS,MOTHER\_FIRST\_NAME,MOTHER\_LAST\_NAME,FATHER\_FIRST\_NAME,FATHER\_LAST\_NAME, PHONE)

VALUES -- We give declared parameters instead of giving them the exact values so that we call more values every time there is a new parent

(NEW\_ID\_PARENTS, NEW\_MOTHER\_FIRST\_NAME, NEW\_MOTHER\_LAST\_NAME, NEW\_FATHER\_FIRST\_NAME, NEW\_FATHER\_LAST\_NAME, NEW\_PHONE);

END //

DELIMITER ;

CALL INSERT\_NEW\_PARENT(009,'KAPA','LULON','TYRON','WATCHON','07847311178'); -- Here we give the values for the new parent

SELECT \* FROM PARENTS; -- This will show all the new values we called above in the table PARENTS