

# **Student Database Management System**

**IFT 530: Advanced Database Management Systems**

**Systems Proposal Initial**

Shiva Kumar Poojari

Hariharan Balakrishnan

**Arizona State University**

**Instructor Name: Dr. Robert Rucker**

**Spring 2022**

## **TABLE OF CONTENTS**

<b>SECTION- 1: INTRODUCTION AND BACKGROUND</b>	<b>3</b>
Questions:	4
List of Entities and Attributes:	4
<b>SECTION 2: ORM DIAGRAM</b>	<b>6</b>
<b>SECTION 3: RELATIONAL SCHEMA</b>	<b>7</b>
<b>SECTION 4: TABLES CREATION</b>	<b>8</b>
<b>SECTION 5: ADDITIONAL CONSTRAINTS</b>	<b>12</b>
<b>SECTION 6: NO SQL</b>	<b>21</b>
<b>SECTION 7:</b>	<b>36</b>

## **SECTION- 1**

### **INTRODUCTION AND BACKGROUND**

A database management system is software that captures and analyzes data through interacting with end users, applications, and the database itself. The database management system software also includes the essential tools for managing the database. Database management systems are used by every organization, especially in the education domain.

A student database management system is automation of manual performance record management which enables the user to assess necessary data at any place and any time through the internet. The student web portal contains a login page where after providing the login details the home page is appeared for the user where it shows important notifications and activities in the college like semester fee payment dates, exam registration, change in exam timetable, workshops or fests to be held, etc. the student can check his attendance a, semester marks and mid-exam marks for every semester which enables them to improve his performance in forthcoming semester. The faculty can also make changes in marks in case of any mistake immediately which eliminates the time-consuming activities like registering a complaint and then faculty approving it then the administration making changes.

**Questions:**

- What is the current students list in the school ?
- Who are the faculties of a particular school ?
- What are the reports of each student in the school ?
- Which are the subjects taught by the faculty ?
- What are the resources that are available in the library ?
- What is the average grade of students from all subjects ?

**List of Entities and Attributes:****Student Management:**

Creating student profiles with unlimited custom categories and fields including demographic data, enrollment, attendance, schedule and more, and share academic records with faculties, and administrators.

**Courses:**

Enroll for academic courses and monitor the progress of students through the degree program with the flexibility to view course timetables and events in the calendar.

**Attendance:**

Allows teachers to mark attendance in the class and send attendance reports to administrators. Streamline attendance tracking and additionally further upgrade the system to automatically send notifications to parents through SMS, email, and

messaging.

**Grades:**

Enables students to check their academic track record from time to time and improve their performance and increase their GPA.

**Reports:**

Shows the results of every semester – the marks obtained in mid sectional exams and grades obtained in semester.

**Staff details:**

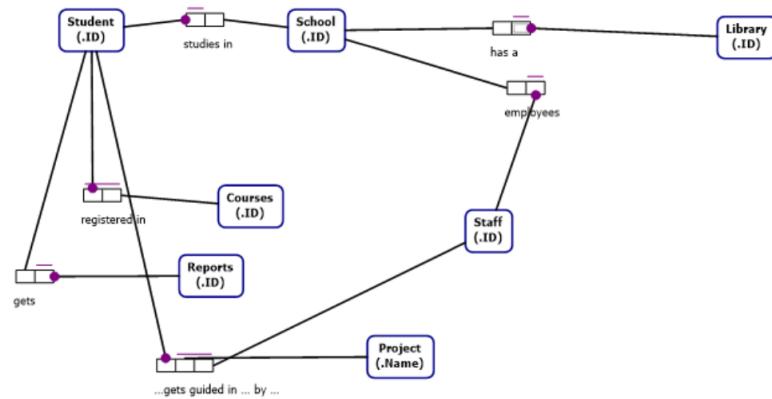
The faculty details include the contact details of each professor like e-mail address, office number and professional details like designation, specialization, qualification (department wise)

**Library:**

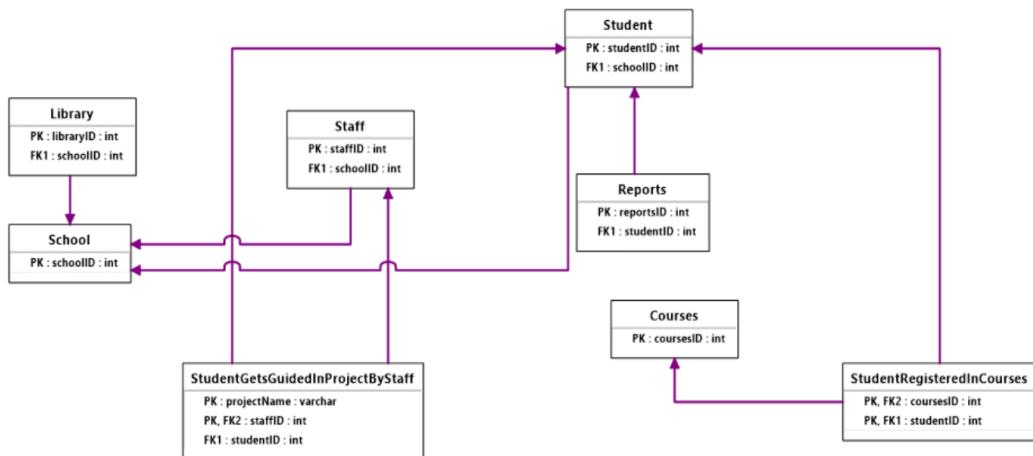
Provides data on the available books and journal papers in the library department wise and details of the library in charge and time of library availability

## SECTION 2

### ORM DIAGRAM

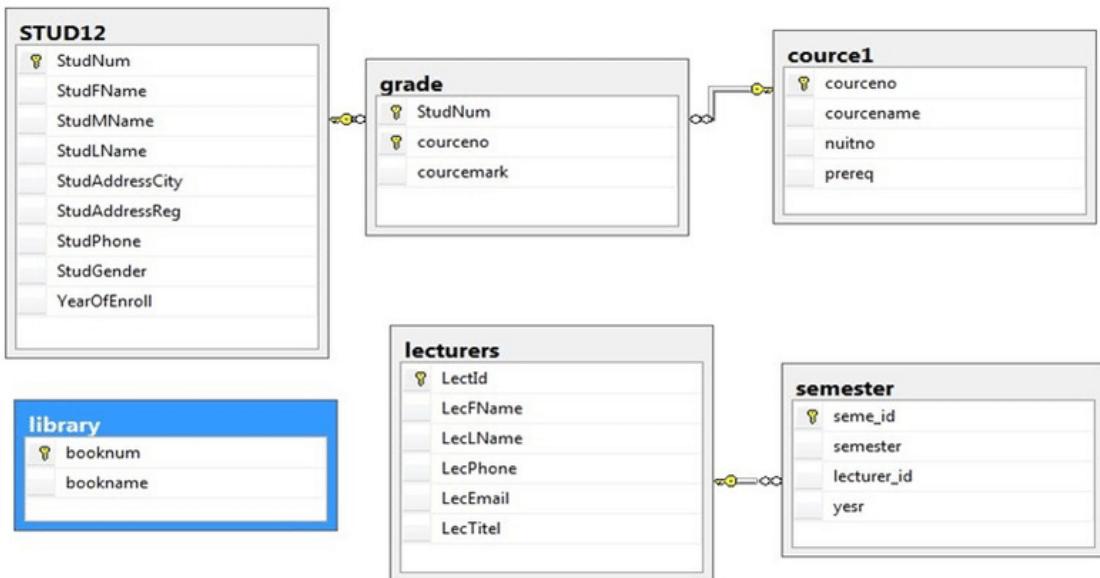


### THE RELATIONAL VIEW



## SECTION 3

### RELATIONAL SCHEMA



## SECTION 4

### TABLES CREATION

```
CREATE TABLE Student(Student_id int, school_name VARCHAR(20));
```

```
CREATE TABLE Library(library_id int, school_name VARCHAR(20));
```

```
CREATE TABLE Reports(reports_id int,student_id int);
```

```
CREATE TABLE courses(course_id int,school_id int);
```

```
CREATE TABLE school(school_name VARCHAR(20),school_id int);
```

```
CREATE TABLE staff(staff_id int,school_id int);
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery1.sql - SHNK\SQLEXPRESS01.student database management system (SHNK\shnk (66)) - Microsoft SQL Server Management Studio". The main window contains a query editor with the following SQL code:

```
CREATE TABLE Student(Student_id int, school_name VARCHAR(20));  
CREATE TABLE Library(library_id int, school_name VARCHAR(20));  
CREATE TABLE Reports(reports_id int,student_id int);  
CREATE TABLE courses(course_id int,school_id int);  
CREATE TABLE school(school_name VARCHAR(20),school_id int);  
CREATE TABLE staff(staff_id int,school_id int);
```

The Object Explorer on the left shows the database structure, including databases like AP, Lab, MS, MyGuitarShop, shiv, and student database management system. The Messages pane at the bottom right shows the execution results:

- (1 row affected)
- (1 row affected)
- (1 row affected)
- (1 row affected)

Completion time: 2022-05-01T23:50:15.1574033-07:00

Query executed successfully.

```
insert into student(student_id,school_name) VALUES(1,'ASU');
```

```
insert into student(student_id,school_name) VALUES(3,'CSU');
```

```
insert into student(student_id,school_name) VALUES(4,'UMC');
```

```
insert into student(student_id,school_name) VALUES(5,'LBHS');
```

```
insert into library(library_id,school_name) VALUES(101,'ASU');
```

```
insert into library(library_id,school_name) VALUES(102,'CSU');
```

```
insert into library(library_id,school_name) VALUES(103,'UMC');
```

```
insert into library(library_id,school_name) VALUES(104,'LBHS');
```

```
insert into Reports(reports_id,student_id) VALUES(121,1);
```

```
insert into Reports(reports_id,student_id) VALUES(122,3);
```

```
insert into Reports(reports_id,student_id) VALUES(123,4);
```

```
insert into Reports(reports_id,student_id) VALUES(124,5);
```

```
insert into courses(course_id,school_id) VALUES(500,111);
```

```
insert into courses(course_id,school_id) VALUES(598,222);
```

```
insert into courses(course_id,school_id) VALUES(530,333);
```

```
insert into courses(course_id,school_id) VALUES(540,444);
```

```
insert into school(school_name,school_id) VALUES('ASU',111);
```

```
insert into school(school_name,school_id) VALUES('CSU',222);
```

```
insert into school(school_name,school_id) VALUES('UMC',333);
```

```
insert into school(school_name,school_id) VALUES('LBHS',444);
```

```
insert into staff(staff_id,school_id)VALUES(11,111);
```

```
insert into staff(staff_id,school_id)VALUES(22,222);
```

```
INSERT INTO staff(staff_id,school_id)VALUES(33,333);
```

```
insert into staff(staff_id,school_id)VALUES(44,444);
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database structure. The right pane contains a query window titled 'SQLQuery1.sql - SHNK\SQLEXPRESS01.student database management system (SHNK\shnk (66))'.

```
insert into student(student_id,school_name) VALUES(1,'ASU');
insert into student(student_id,school_name) VALUES(3,'CSU');
insert into student(student_id,school_name) VALUES(4,'UNC');
insert into student(student_id,school_name) VALUES(5,'LBHS');

insert into library(library_id,school_name) VALUES(101,'ASU');
insert into library(library_id,school_name) VALUES(102,'CSU');
insert into library(library_id,school_name) VALUES(103,'UNC');
insert into library(library_id,school_name) VALUES(104,'LBHS');

insert into Reports(reports_id,student_id) VALUES(121,1);
insert into Reports(reports_id,student_id) VALUES(122,3);
insert into Reports(reports_id,student_id) VALUES(123,4);
insert into Reports(reports_id,student_id) VALUES(124,5);

insert into courses(course_id,school_id) VALUES (500,111);
insert into courses(course_id,school_id) VALUES(598,222);
insert into courses(course_id,school_id) VALUES(530,333);
insert into courses(course_id,school_id) VALUES(540,444);
```

The execution results show five rows affected for each of the four insert statements, indicating successful execution. The completion time is listed as 2022-05-01T23:50:15.1574033-07:00.

At the bottom of the query window, a message bar indicates 'Query executed successfully.'

SQLQuery1.sql - SHNK\SQLEXPRESS01.student database management system (SHNK\shnk (66)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

New Query | Execute | Quick Launch (Ctrl+Q)

student database manager

Object Explorer

IN\SQLEXPRESS01 (SQL Server 15.0.2000 - SHN Databases)

- System Databases
- Database Snapshots
- AP
- Lab
- MS
- MyGuitarShop
- shiv
- student database management system

Security

Server Objects

Replication

PolyBase

Management

XEvent Profiler

SQLQuery1.sql - S...m (SHNK\shnk (66))

```
insert into school(school_name,school_id) VALUES('ASU',111);
insert into school(school_name,school_id) VALUES('CSU',222);
insert into school(school_name,school_id) VALUES('UKC',333);
insert into school(school_name,school_id) VALUES('LBHS',444);

insert into staff(staff_id,school_id)VALUES(11,111);
insert into staff(staff_id,school_id)VALUES(22,222);
INSERT INTO staff(staff_id,school_id)VALUES(33,333);
insert into staff(staff_id,school_id)VALUES(44,444);
```

80 % Messages

(1 row affected)  
(1 row affected)  
(1 row affected)  
(1 row affected)

Completion time: 2022-05-01T23:50:15.1574033-07:00

Query executed successfully.

LN 57 Col 53 Ch 53 INS

SHNK\SQLEXPRESS01 (15.0 RTM) SHNK\shnk (66) student database manag... 00:00:00 0 rows

Ready

## SECTION 5

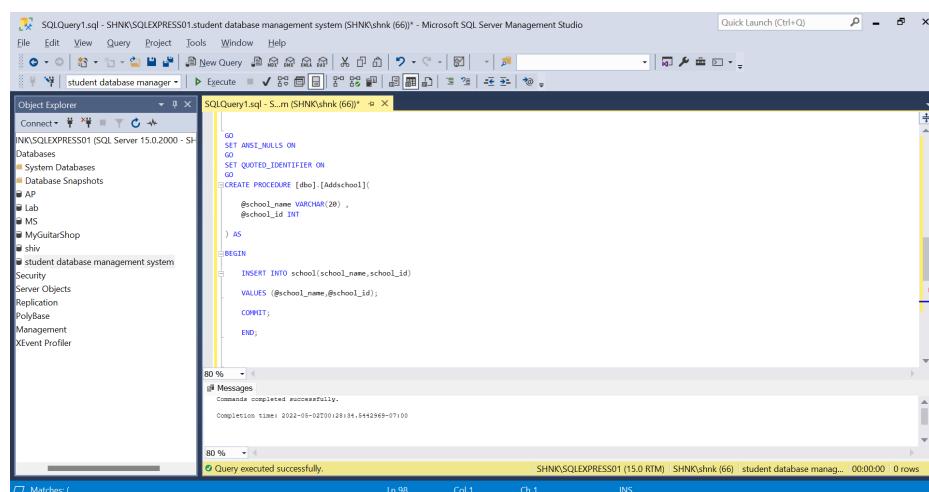
## Additional Constraints

## Stored Procedure

- #### 1. Creating a stored procedure for inserting data into school table

```
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[Addschool](
    @school_name VARCHAR(20) ,
    @school_id INT
) AS
BEGIN
    INSERT INTO school(school_name,schoo
VALUES (@school_name,@school_id);

    COMMIT;
END;
```



GO

```
DECLARE @return_value int
```

```
EXEC @return_value = [dbo].[AddSchool]
```

```
    @school_name = 'NSA',  
    @school_id = 555
```

```
SELECT 'Return Value' = @return_value
```

GO

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure, including databases like INKSQLEXPRESS01 and student database management system. The central pane contains a query window titled "SQLQuery1.sql - S...m (SHNK\shnk (66))". The query itself is:

```
GO  
DECLARE @return_value int  
EXEC @return_value = [dbo].[AddSchool]  
    @school_name = 'NSA',  
    @school_id = 555  
SELECT 'Return Value' = @return_value  
GO
```

Below the query window, the results pane shows the output:

```
80 % Results Messages  
(1 row affected)  
Completion time: 2022-05-02T00:33:21.0094901-07:00
```

A status bar at the bottom indicates "Query completed with errors." and provides other session details.

## 2. Creating a stored procedure for school and staff.

GO

```
SET ANSI_NULLS ON  
GO  
SET QUOTED_IDENTIFIER ON  
GO  
CREATE PROCEDURE [dbo].[Addstaff](
```

```
    @staff_id INT ,  
    @school_id INT  
) AS
```

```
BEGIN
```

```
    INSERT INTO staff(staff_id,school_id)
```

```
        VALUES (@staff_id,@school_id);
```

```
    COMMIT;
```

```
END;
```

```
GO
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery1.sql - SHNK\SQLEXPRESS01.student database management system (SHNK\shnk (66)) - Microsoft SQL Server Management Studio". The left pane is the Object Explorer, showing the database structure for "INNEXPRESS01 (SQL Server 15.0.2000 - SH)". The right pane contains a query window titled "SQLQuery1.sql - S...m (SHNK\shnk (66))". The code in the window is:

```
GO  
SET ANSI_NULLS ON  
GO  
SET QUOTED_IDENTIFIER ON  
GO  
CREATE PROCEDURE [dbo].[Addstaff]  
    (@staff_id INT ,  
     @school_id INT  
 ) AS  
BEGIN  
    INSERT INTO staff(staff_id,school_id)  
    VALUES (@staff_id,@school_id);  
    COMMIT;  
END;  
GO
```

The status bar at the bottom indicates "Query executed successfully." and "0 rows".

GO

DECLARE @return\_value int

EXEC @return\_value = [dbo].[Addstaff]

    @staff\_id = 55,

    @school\_id = 555

SELECT 'Return Value' = @return\_value

GO

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the connection path: INK\SQLEXPRESS01 (SQL Server 15.0.2000 - SHN\shnk). The right pane contains a query window titled 'SQLQuery1.sql - S...m (SHN\shnk (66))'. The query is:

```

GO
DECLARE @return_value int
EXEC    @return_value = [dbo].[Addstaff]
        @staff_id = 55,
        @school_id = 555
SELECT  'Return Value' = @return_value
GO

```

The results pane shows the message: 'Query completed with errors.' and 'Completion time: 2022-05-02T00:39:42.1702985-07:00'. The status bar at the bottom indicates 'LN 132' and '1 rows'.

## Triggers:

CREATE Trigger library\_id ON library

FOR INSERT

AS

```
DECLARE @libraryid int
```

```
DECLARE @schoolname VARCHAR(20)
```

```
SELECT @libraryid = library_id from library
```

```
SELECT @schoolname = school_name from library
```

```
IF @schoolname = 'ASU'
```

```
UPDATE libraryid
```

```
SET libraryid = 105 WHERE library_id = @libraryid;
```

```

CREATE Trigger library_id ON library
FOR INSERT
AS
    DECLARE @libraryid int
    DECLARE @schoolname VARCHAR(20)

    SELECT @libraryid = library_id from library
    SELECT @schoolname = school_name from library

    IF @schoolname = 'ASU'
        UPDATE library
        SET libraryid = 105 WHERE library_id = @libraryid;

```

Messages

Commands completed successfully.

Completion time: 2022-05-02T10:46:41.7108179-07:00

Query executed successfully.

**INSERT INTO library (library\_id,school\_name ) VALUES (105,'USF');**

**SELECT \* FROM library;**

```

INSERT INTO library (library_id,school_name ) VALUES (105,'USF');

SELECT * FROM library;

```

library_id	school_name
101	ASU
102	CSU
103	UMC
104	LBHS
105	USF

Results

Query executed successfully.

## Section - 5.1

### Retrieving Queries

1. `SELECT * FROM courses WHERE school_id = 333;`

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases like AP, Lab, MS, MyGuitarShop, shiv, and student database management system. The central pane displays the query `SELECT * FROM courses WHERE school_id = 333;`. The results pane shows a single row: course\_id 530 and school\_id 333. A message at the bottom indicates the query was executed successfully.

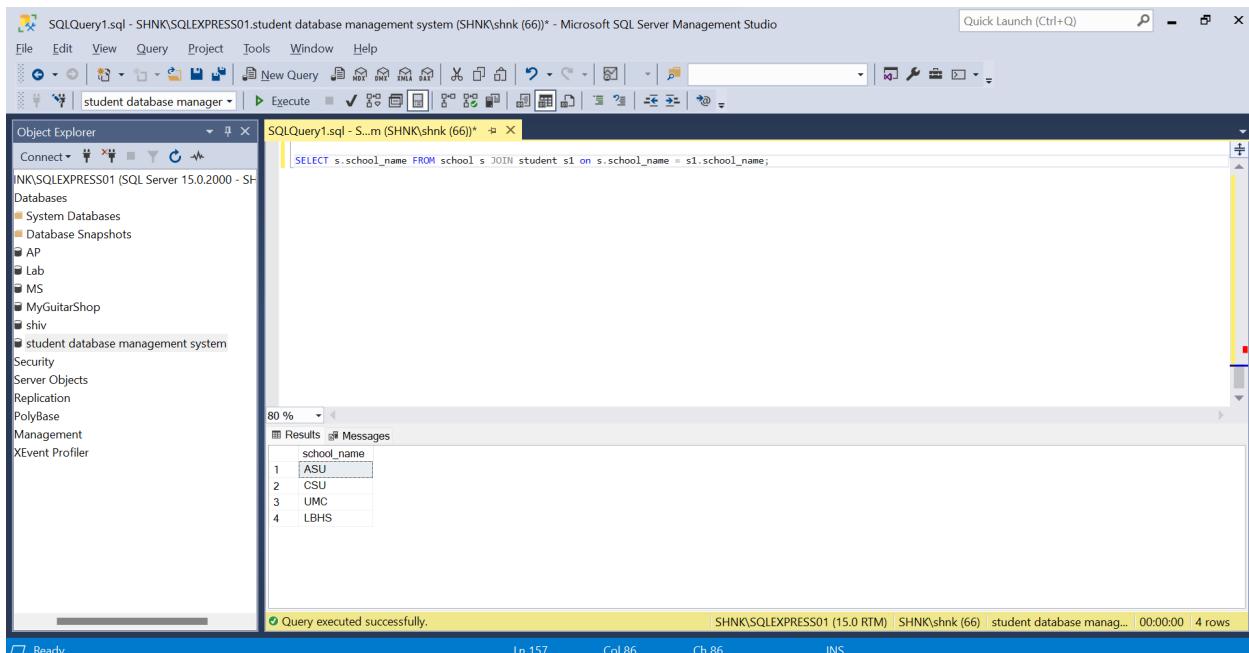
course_id	school_id
530	333

2. `SELECT school_name,school_id FROM school;`

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases like AP, Lab, MS, MyGuitarShop, shiv, and student database management system. The central pane displays the query `SELECT school_name,school_id FROM school;`. The results pane shows five rows of data: ASU (111), CSU (222), UMC (333), LBHS (444), and NSA (555). A message at the bottom indicates the query was executed successfully.

school_name	school_id
ASU	111
CSU	222
UMC	333
LBHS	444
NSA	555

3. `SELECT s.school_name FROM school s JOIN student s1 on s.school_name = s1.school_name;`



The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database structure. The right pane contains a query window titled "SQLQuery1.sql - S...m (SHNK\shnk (66))". The query is:

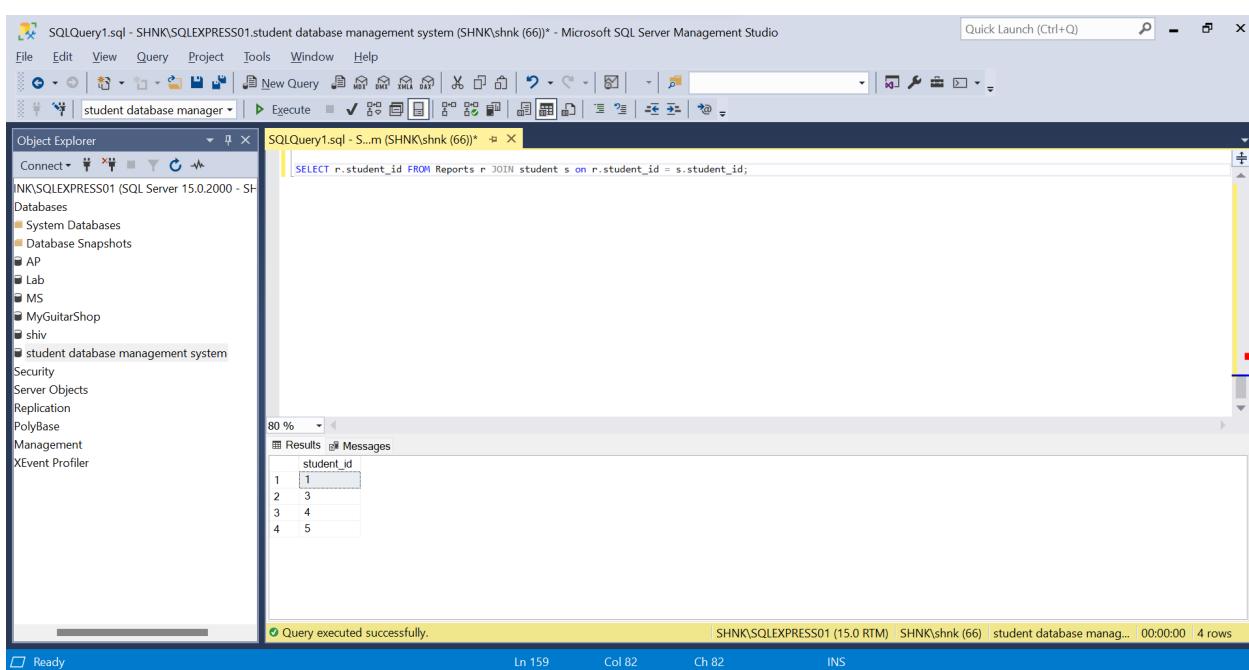
```
SELECT s.school_name FROM school s JOIN student s1 on s.school_name = s1.school_name;
```

The results pane shows the output of the query:

school_name
ASU
CSU
UMC
LBHS

Below the results, a message indicates: "Query executed successfully." and "4 rows".

4. `SELECT r.student_id FROM Reports r JOIN student s on r.student_id = s.student_id;`



The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database structure. The right pane contains a query window titled "SQLQuery1.sql - S...m (SHNK\shnk (66))". The query is:

```
SELECT r.student_id FROM Reports r JOIN student s on r.student_id = s.student_id;
```

The results pane shows the output of the query:

student_id
1
2
3
4
5

Below the results, a message indicates: "Query executed successfully." and "4 rows".

5. `SELECT s.school_id FROM staff s JOIN school s1 on s.school_id = s1.school_id;`

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery1.sql - SHNK\SQLEXPRESS01.student database management system (SHNK\shnk (66)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, Help. The toolbar has various icons for file operations like New Query, Save, Print, and Execute. The Object Explorer on the left lists databases: System Databases, Database Snapshots, AP, Lab, MS, MyGuitarShop, shiv, and student database management system. The Results pane on the right displays the output of the executed query:

```
SELECT s.school_id FROM staff s JOIN school s1 on s.school_id = s1.school_id;
```

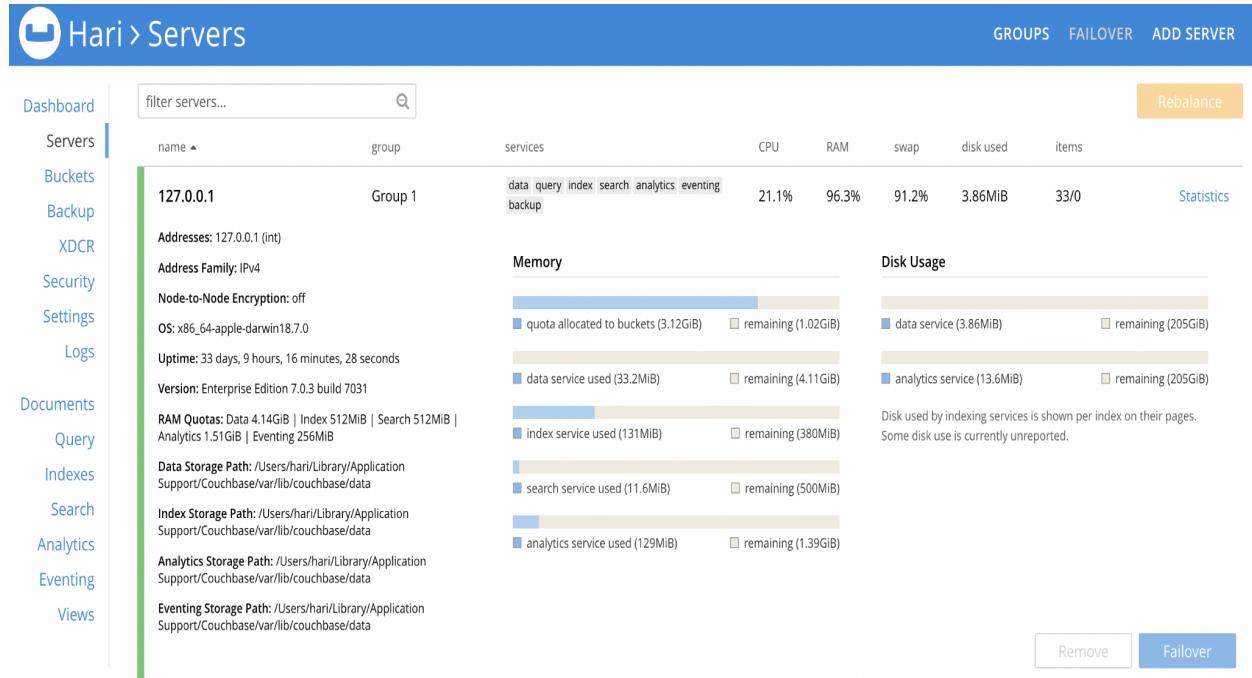
school_id
1 111
2 222
3 333
4 444
5 555

Below the results, a message bar indicates "Query executed successfully." with a green icon. The status bar at the bottom shows "SHNK\SQLEXPRESS01 (15.0 RTM) SHNK\shnk (66) student database manag... 00:00:00 5 rows".

## SECTION 6

### NO SQL

Used couchbase enterprise 7+ edition database to execute queries:



#### Section 6.1:

Created buckets/ Entities are:

The screenshot shows the Couchbase Bucket list. The left sidebar includes links for Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The main panel lists four buckets: Reports, School, Staff, and Students. Each bucket row includes columns for name, documents, document percentage, scopes, and disk usage. Buttons for 'Documents' and 'Scopes & Collections' are also present.

						Documents	Scopes & Collections
	Reports	0	100%	0	3.96MiB / 700MiB	542KiB	Documents Scopes & Collections
	School	2	100%	0	3.97MiB / 500MiB	550KiB	Documents Scopes & Collections
	Staff	0	100%	0	3.96MiB / 500MiB	542KiB	Documents Scopes & Collections
	Students	4	100%	0	3.99MiB / 500MiB	590KiB	Documents Scopes & Collections

## Section 6.2:

Students:

```
insert into Students(key, value)
```

```
values ("101",
```

```
{
```

```
    "AdmissionId": "101",
```

```
    "RollId": 11,
```

```
    "StudentName": "Joe",
```

```
    "SchoolName": "IraA Fulton",
```

```
    "SchoolID": 1,
```

```
    "phoneNumber": "4802284786",
```

```
    "DOB": "1998-10-17",
```

```
    "emailID": "joe17@gmail.com",
```

```
    "Rank": 4,
```

```
    "GPA": 3.90
```

```
}),("102",
```

```
{
```

```
    "AdmissionId": "102",
```

```
    "RollId": 12,
```

```
    "StudentName": "Chandler",
```

```
    "SchoolName": "W.P Carey",
```

```
    "SchoolID": 2,
```

```
"phoneNumber": "4802284790",
"DOB": "1998-01-09",
"emailID": "chandler09@gmail.com",
"Rank": 3,
"GPA": 4.00
```

```
}),("103",
```

```
{
```

```
  "AdmissionId": "103",
  "RollId": 13,
  "StudentName": "Mike",
  "SchoolName": "IraA Fulton",
  "SchoolID": 1,
  "phoneNumber": "4752284701",
  "DOB": "1998-10-02",
  "emailID": "mike02@gmail.com",
  "Rank": 2,
  "GPA": 4.11
```

```
}),(
```

```
  "104",
{
  "AdmissionId": "104",
  "RollId": 14,
  "StudentName": "Issac",
```

```

    "SchoolName": "W.P carey",
    "SchoolID": 2,
    "phoneNumber": "4752284769",
    "DOB": "1999-04-04",
    "emailID": "issac69@gmail.com",
    "Rank": 1,
    "GPA": 4.33
)

```

The screenshot shows the Hari > Query interface. On the left, a sidebar lists various database management options like Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query (which is selected), Indexes, Search, Analytics, Eventing, and Views. The main area is the 'Query Editor' containing the following code:

```

1 insert into Students (key, value)
2 values ("101",
3 {
4     "AdmissionId": "101",
5     "RollId": 11,
6     "StudentName": "Joe",
7     "SchoolName": "IraA Fulton",
8     "phoneNumber": "4802284786",
9     "DOB": "1998-10-17",
10    "emailID": "joe17@gmail.com",
11    "Rank": 4,
12    "GPA": 3.90
13 }, ("102",
14 {
15     "AdmissionId": "102",
16     "RollId": 12,
17     "StudentName": "Hariharan",
18     "SchoolName": "W.P carey",
19     "phoneNumber": "4752284769",
20     "DOB": "1999-04-04",
21     "emailID": "issac69@gmail.com",
22     "Rank": 1,
23     "GPA": 4.33
24 })

```

Below the editor, there are buttons for Execute, Run as TX, Index Advisor, Explain, and a status message indicating success just now | 69.2ms | 4 mutations. To the right, there's a 'query context' dropdown, an 'Explore Your Data' section with links to Customers, Hariharan, Orders, Reports, School, Staff, and Students, and a 'refresh' button.

Staff:

insert into Staff(key, value)

values ("701",

```
{  
    "StaffName": "Dr.Rucker",  
    "StaffID": 701,  
    "SchoolName": "IraA Fulton",  
    "SchoolID": 1,  
    "SubjectName": "ADBMS",  
    "emailID": "rucker@asu.edu"  
},("702",  
{  
    "StaffName": "Dr.Asha",  
    "StaffID": 702,  
    "SchoolName": "W.P Carey",  
    "SchoolID": 2,  
    "SubjectName": "Data Mining",  
    "emailID": "asha@asu.edu"  
},("703",  
{  
    "StaffName": "Dr.Atkinson",  
    "StaffID": 703,  
    "SchoolName": "IraA Fulton",  
    "SchoolID": 1,  
    "SubjectName": "NLP",  
    "emailID": "atkinson@asu.edu"
```

})

The screenshot shows the Hari Query interface. On the left, a sidebar menu includes options like Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query (which is selected), Indexes, Search, Analytics, Eventing, and Views. The main area is a "Query Editor" containing the following code:

```
1 insert into Staff(key, value)
2 values ("701",
3 {
4     "StaffName": "Dr.Rucker",
5     "StaffID": 701,
6     "SchoolName": "IraA Fulton",
7     "SchoolID": 1,
8     "SubjectName": "ADBMS",
9     "emailID": "rucker@asu.edu"
10 },("702",
11 {
12     "StaffName": "Dr.Asha",
13     "StaffID": 702,
14     "SchoolName": "W.P Carey",
15     "SchoolID": 2,
16     "SubjectName": "Data Mining",
17     "emailID": "ashar@asu.edu"
18 })
```

Below the code, there are buttons for "Execute", "Run as TX", "Index Advisor", and "Explain". A status message indicates "success just now | 62.3ms | 3 mutations". The results section shows a single row of data:

```
1 {
2   "results": []
3 }
```

On the right side, there's a sidebar titled "Explore Your Data" with a list of document types: Customers, Hariharan, Library, Orders, Reports, School, Staff, and Students. At the bottom right is a "Refresh" button.

## School:

insert into School(key, value)

values ("1",

{

"SchoolName": "IraA Fulton",

"SchoolID": 1,

"phoneNumber": "4802280000",

"emailID": "iraafulton@gmail.com"

}),("2",

{

"SchoolName": "W.P Carey",

"SchoolID": 2,

"phoneNumber": "4800804790",

```

        "emailID": "wpcarey@gmail.com"
    })

```

The screenshot shows the Futon interface with the following details:

- Header:** Hari > Query, IMPORT, EXPORT.
- Left Sidebar:** Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, **Query** (selected), Indexes, Search, Analytics, Eventing, Views.
- Query Editor:**

```

1  insert into School(key, value)
2  values ("1",
3  {
4      "SchoolName": "IraA Fulton",
5      "SchoolID": 1,
6      "phoneNumber": "4802280000",
7      "emailID": "iraafulton@gmail.com"
8  }),("2",
9  {
10     "SchoolName": "W.P Carey",
11     "SchoolID": 2,
12     "phoneNumber": "4800804790",
13     "emailID": "wpcarey@gmail.com"
14 })

```
- Status Bar:** history (45/45), query context dropdown, Execute, Run as TX, Index Advisor, Explain, success just now | 37.6ms | 2 mutations, format dropdown.
- Results:** Results tab selected, JSON tab active, showing the response:

```

1 { "results": []
2 }
3

```
- Explore Your Data:** Common document types, field names, and sample values from your data - by bucket, scope, collection.
  - Customers
  - Hariharan
  - Orders
  - Reports
  - School
  - Staff
  - Students

## Reports:

insert into Reports(key, value)

values ("81",

{

"StaffName": "Dr.Rucker",

"ReportID": 81,

"StudentName": "Issac",

"RollId": 14,

"SchoolName": "IraA Fulton",

"SchoolID": 1,

"SubjectName": "ADBMS"

```
} ), ("82",  
 {  
 "StaffName": "Dr.Asha",  
 "ReportID": 82,  
 "StudentName": "Joe",  
 "RollId": 11,  
 "SchoolName": "W.P Carey",  
 "SchoolID": 2,  
 "SubjectName": "Data Mining"  
 } ), ("83",  
 {  
 "StaffName": "Dr.Atkinson",  
 "ReportID": 83,  
 "StudentName": "Mike",  
 "RollId": 13,  
 "SchoolName": "IraA Fulton",  
 "SchoolID": 1,  
 "SubjectName": "NLP"  
 })
```

The screenshot shows the Hari Query interface. On the left, a sidebar lists various database management options like Dashboard, Servers, Buckets, etc. The main area has a "Query Editor" tab open, displaying a code snippet:

```

1 insert into Reports(key, value)
2 values ("81",
3 {
4     "StaffName": "Dr.Rucker",
5     "ReportID": 81,
6     "StudentName": "Issac",
7     "RollId": 14,
8     "SchoolName": "IraA Fulton",
9     "SchoolID": 1,
10    "SubjectName": "ADBMS"
11 }, "82",
12 {
13     "StaffName": "Dr.Asha",
14     "ReportID": 82,
15     "StudentName": "Joe",
16     "RollId": 11,
17     "SchoolName": "IraA Fulton",
18     "SchoolID": 1,
19     "SubjectName": "ADBMS"
20 })

```

Below the editor are buttons for "Execute", "Run as TX", "Index Advisor", and "Explain". The status bar indicates "success just now | 70.5ms | 3 mutations". The results table shows the following JSON output:

```

1 {
2   "results": []
3 }

```

On the right, there's a sidebar titled "Explore Your Data" with sections for Customers, Hariharan, Library, Orders, Reports, School, Staff, and Students. The Students section shows 1 collection and 4 docs.

## Section 6.3:

### JSON Document

The screenshot shows the Hari Documents interface. The sidebar includes options like Dashboard, Servers, Buckets, etc. The main area displays a table of student documents with columns for id, name, and details.

	id	name	details
	101	"AdmissionId": "101", "DOB": "1998-10-17", "GPA": 3.9, "Rank": 4, "RollId": 11, "SchoolID": 1, "SchoolName": "IraA Fulton", "StudentName": "Joe", "emailID": "joe17@gmail.com", "phoneNumber": "4802284786"	
	102	"AdmissionId": "102", "DOB": "1998-01-09", "GPA": 4, "Rank": 3, "RollId": 12, "SchoolID": 2, "SchoolName": "W.P Carey", "StudentName": "Chandler", "emailID": "chandler09@gmail.com", "phoneNumber": "4802284790"	
	103	"AdmissionId": "103", "DOB": "1998-10-02", "GPA": 4.11, "Rank": 2, "RollId": 13, "SchoolID": 1, "SchoolName": "IraA Fulton", "StudentName": "Mike", "emailID": "mike02@gmail.com", "phoneNumber": "4752284701"	
	104	"AdmissionId": "104", "DOB": "1999-04-04", "GPA": 4.33, "Rank": 1, "RollId": 14, "SchoolID": 2, "SchoolName": "W.P carey", "StudentName": "Issac", "emailID": "issac69@gmail.com", "phoneNumber": "4752284769"	

**Hari > Documents**

ADD DOCUMENT

Workbench Import

Dashboard

Servers

Buckets

Backup

XDCR

Security

Settings

Logs

Documents

Query

Indexes

Search

Analytics

Eventing

Views

Keyspace bucket.scope.collection

Staff

\_default

\_default

Limit ⓘ

10

Offset ⓘ

0

Document ID ⓘ

show range N1QL WHERE ⓘ

optional... no indexes available...

ADD DOCUMENT

Workbench Import

3 Results for Staff.\_default.\_default, limit: 10, offset: 0

enable field editing

&lt; prev batch | next batch &gt;

id

701  
702  
703

```
{"SchoolID":1,"SchoolName":"IraA Fulton","StaffID":701,"StaffName":"Dr.Rucker","SubjectName":"ADBMS","emailID":"rucker@asu.edu"}  

{"SchoolID":2,"SchoolName":"W.P Carey","StaffID":702,"StaffName":"Dr.Asha","SubjectName":"Data Mining","emailID":"asha@asu.edu"}  

{"SchoolID":1,"SchoolName":"IraA Fulton","StaffID":703,"StaffName":"Dr.Atkinson","SubjectName":"NLP","emailID":"atkinson@asu.edu")
```

Retrieve Docs

**Hari > Documents**

ADD DOCUMENT

Workbench Import

Dashboard

Servers

Buckets

Backup

XDCR

Security

Settings

Logs

Documents

Query

Indexes

Search

Analytics

Eventing

Keyspace bucket.scope.collection

School

\_default

\_default

Limit ⓘ

10

Offset ⓘ

0

Document ID ⓘ

show range N1QL WHERE ⓘ

optional... no indexes available...

ADD DOCUMENT

Workbench Import

2 Results for School.\_default.\_default, limit: 10, offset: 0

enable field editing

&lt; prev batch | next batch &gt;

id

1  
2

```
{"SchoolID":1,"SchoolName":"IraA Fulton","emailID":"iraafulton@gmail.com","phoneNumber":"4802280000"}  

{"SchoolID":2,"SchoolName":"W.P Carey","emailID":"wpcarey@gmail.com","phoneNumber":"4800804790")
```

Retrieve Docs

**Hari > Documents**

ADD DOCUMENT

Workbench Import

Dashboard

Servers

Buckets

Backup

XDCR

Security

Settings

Logs

Documents

Query

Indexes

Search

Analytics

Eventing

Keyspace bucket.scope.collection

Reports

\_default

\_default

Limit ⓘ

10

Offset ⓘ

0

Document ID ⓘ

show range N1QL WHERE ⓘ

optional... no indexes available...

ADD DOCUMENT

Workbench Import

4 Results for Reports.\_default.\_default, limit: 10, offset: 0

enable field editing

&lt; prev batch | next batch &gt;

id

81  
82  
83  
84

```
{"ReportID":81,"RollID":14,"SchoolID":1,"SchoolName":"IraA Fulton","StaffName":"Dr.Rucker","StudentName":"Issac","SubjectName":"ADBMS"}  

 {"ReportID":82,"RollID":11,"SchoolID":2,"SchoolName":"W.P Carey","StaffName":"Dr.Asha","StudentName":"Joe","SubjectName":"Data Mining"}  

 {"ReportID":83,"RollID":13,"SchoolID":1,"SchoolName":"IraA Fulton","StaffName":"Dr.Atkinson","StudentName":"Mike","SubjectName":"NLP"}  

 {"ReportID":84,"RollID":11,"SchoolID":1,"SchoolName":"IraA Fulton","StaffName":"Dr.Atkinson","StudentName":"Joe","SubjectName":"NLP")
```

Retrieve Docs

## Section 6.4:

Query 1: What is the current students list in the schools ?

Displayed list of the student and their information by roll identity number wise from Students bucket..

**SELECT \* From Students**

**Order by RollID ;**

The screenshot shows the Futaba NoSQL database interface. On the left is a sidebar with navigation links: Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The 'Analytics' link is currently selected. The main area has tabs for 'Query Editor' and 'Query Results'. In the 'Query Editor' tab, the following code is entered:

```
1 SELECT * From Students
2 Order by RollID ;
```

Below the code, there are buttons for 'Execute' (highlighted in blue), 'Explain', and 'Format'. A status message indicates success: '✓ success | elapsed: 196.01ms | execution: 156.94ms | docs scanned: 4 | docs returned: 4 | size: 912 bytes'. The 'Query Results' tab is active, showing the JSON output of the query. The output is a single document with an array of student objects:1 [
2 {
3 "Students": [
4 {
5 "AdmissionId": "101",
6 "DOB": "1998-10-17",
7 "GPA": 3.9,
8 "Rank": 4,
9 "RollId": 11,
10 "SchoolID": 1,
11 "SchoolName": "IraA Fulton",
12 "StudentName": "Joe",
13 "emailID": "joe17@gmail.com",
14 "phoneNumber": "4802284786"
15 }
16 ],
17 }

Query 2: Who are the faculties of a particular school ?

Displayed all the faculties according to their respective school using school ID from staff bucket.

Select StaffName, SchoolName, emailID  
From Staff  
Where SchoolID = 1 ;

The screenshot shows the Hari Analytics interface. On the left is a sidebar with various navigation options: Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics (which is selected), Eventing, and Views. The main area has tabs for 'Query Editor' and 'Query Results'. In the 'Query Editor' tab, there is a code editor with the following SQL-like query:

```

1 Select StaffName, SchoolName, emailID
2 From Staff
3 Where SchoolID = 1 ;

```

Below the code editor are buttons for 'Execute' (highlighted in blue), 'Explain', and 'query context'. The 'Execute' button has a status message: '✓ success | elapsed: 253.88ms | execution: 160.52ms | docs scanned: 3 | docs returned: 2 | size: 178 bytes'. To the right of the code editor is a 'format' icon. In the 'Query Results' tab, the results are displayed as JSON documents:

```

1 [
2   {
3     "StaffName": "Dr.Rucker",
4     "SchoolName": "IraA Fulton",
5     "emailID": "rucker@asu.edu"
6   },
7   {
8     "StaffName": "Dr.Atkinson",
9     "SchoolName": "IraA Fulton",
10    "emailID": "atkinson@asu.edu"
11  }
12 ]

```

At the top of the results table are buttons for 'JSON' (highlighted in blue), 'Table', 'Chart', 'Plan', and 'Plan Text'. There are also back and forward navigation buttons at the top of the results table.

Query 3: What are the reports of each student in the school ?

Extracted information of students and their reports by joining both buckets using JOIN syntax.

From Students s join Reports r on s.SchoolID = r.SchoolID  
 Select s.StudentName, s.RollID, s.GPA, r.ReportID, r.StaffName, r.SubjectName

The screenshot shows the Hari Analytics interface. On the left, there's a sidebar with various navigation links: Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics (which is selected), Eventing, and Views. The main area has tabs for 'Query Editor' and 'Query Results'. In the 'Query Editor' tab, a query is written:

```

1 From Students s join Reports r on s.SchoolID = r.SchoolID
2 Select s.StudentName, s.RollID, s.GPA, r.ReportID, r.StaffName, r.SubjectName

```

Below the query are buttons for 'Execute' (highlighted in blue), 'Explain', and 'query context'. The 'Execute' button has a status message: '✓ success | elapsed: 1.07s | execution: 934.14ms | docs scanned: 8 | docs returned: 8 | size: 844 bytes'. To the right of the editor is a 'Analytics' sidebar with sections for 'Reports', 'Local', 'School...', 'Staff...', 'Local', 'Students', and 'Local'. The 'Query Results' tab displays the following JSON data:

```

16 [
17   {
18     "StudentName": "Joe",
19     "GPA": 3.9,
20     "ReportID": 84,
21     "StaffName": "Dr. Atkinson",
22     "SubjectName": "NLP"
23   },
24   {
25     "StudentName": "Chandler",
26     "GPA": 4,
27     "ReportID": 82,
28     "StaffName": "Dr. Asha",
29     "SubjectName": "Data Mining"
30   },
31   {
32     "StudentName": "Mike",
33     "GPA": 3.7,
34     "ReportID": 81,
35     "StaffName": "Dr. Rajesh",
36     "SubjectName": "Machine Learning"
37   }
]

```

Query 4: Which are the subjects taught by the faculty ?

Extracted information of subjects handled by faculties from the staff bucket.

Select StaffName, SubjectName

From Staff ;

The screenshot shows the Hari Analytics interface. On the left, there's a sidebar with various navigation options like Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics (which is selected), Eventing, and Views. The main area has tabs for 'Query Editor' and 'Query Results'. In the 'Query Editor' tab, the code is:

```
1 Select StaffName, SubjectName
2 From Staff;
```

Below the code, there are buttons for 'Execute' (highlighted in blue), 'Explain', and 'Format'. A status message indicates success: '✓ success | elapsed: 1.97s | execution: 1.62s | docs scanned: 3 | docs returned: 3 | size: 163 bytes'. The 'Query Results' tab shows the following JSON output:

```
1 [
2   {
3     "StaffName": "Dr.Rucker",
4     "SubjectName": "ADBMS"
5   },
6   {
7     "StaffName": "Dr.Asha",
8     "SubjectName": "Data Mining"
9   },
10  {
11    "StaffName": "Dr.Atkinson",
12    "SubjectName": "NLP"
13  }
14 ]
```

On the right side, there are several collapsed sections labeled 'Analytics', 'Local', 'School', 'Staff', 'Student', and 'Local'.

Query 5: What is the average grade of students from subjects ?

Found the average grade points from all students in their enrolled subjects.

Select Avg(GPA) as Average\_Grade  
From Students ;

The screenshot shows the Hari Analytics interface. The sidebar is identical to the previous one. The 'Query Editor' tab contains the following SQL-like query:

```
1 Select Avg(GPA) as Average_Grade
2 From Students ;
```

Below the code, there are buttons for 'Execute' (highlighted in blue), 'Explain', and 'Format'. A status message indicates success: '✓ success | elapsed: 769.37ms | execution: 669.30ms | docs scanned: 4 | docs returned: 1 | size: 39 bytes'. The 'Query Results' tab shows the following JSON output:

```
1 [
2   {
3     "Average_Grade": 4.085000000000001
4   }
5 ]
```

## **SECTION 7:**

### **SUMMARY**

Our student management system aids in the administration, information, and scheduling in schools. A vast amount of data can be generated and used by a school system. This information must be provided to students, faculty, and parents in a suitable manner. A student management system aids schools in storing, managing, and disseminating this data.

Student management systems serve educational institutions in a variety of ways, the most important of which is centralized data administration and accessibility. Teachers will be able to input, maintain, and access student data more simply. Parents and guardians will have a better understanding of how their children perform in class. Compliance and other regulatory obligations at the state level are likewise significantly easier to meet. Instead of papers, an automated student database management system can be used.

## **CONCLUSION**

To sum up, working on this project taught us how to design a database structure. We can and should comprehend a wide range of DB skills. The goal of this project was to learn and understand how to convert an ORM diagram into a Relation schema that can then be converted into tables and databases.

We were able to learn as well as implement schema-oriented SQL by contributing to this project. We also learned what attributes are required for a successful SQL query. SQL's stored procedures, triggers, and functions can help you access critical information from a complex data

structure.

In Couchbase, we operated NoSQL by writing queries in our loaded data, writing SQL++ queries for buckets where the analytics service is active, extending the ORM with nested documents, and creating JSON documents for entities where all tasks were completed.

## **REFERENCES:**

1. TEXTBOOK : Murach's SQL Server 2016 for Developers, by Bryan Syverson and Joel Murach, June 2016.
2. WEBSITE : SQL++ For SQL Users by Don Chamberlin, September 2018 .

<https://www.couchbase.com/analytics-data>