****

**Assignment # 3**

**Presented By**

**M.Waqas (22I - 2469)**

**Abdullah Mansoor (22I - 8808)**

**To**

**Sir Bilal**

Table of Contents

[1 Question#1 3](#_Toc164881760)

[1.1 Entity Relation Diagram 3](#_Toc164881761)

[1.2 Enhanced Entity Relation Diagram 4](#_Toc164881762)

[1.3 Relational Data Model 5](#_Toc164881763)

[1.4 Queries 6](#_Toc164881764)

[2 Question#2 9](#_Toc164881765)

[2.1 Examples for Anomalies 9](#_Toc164881766)

[2.1.1 Insertion Anomaly 9](#_Toc164881767)

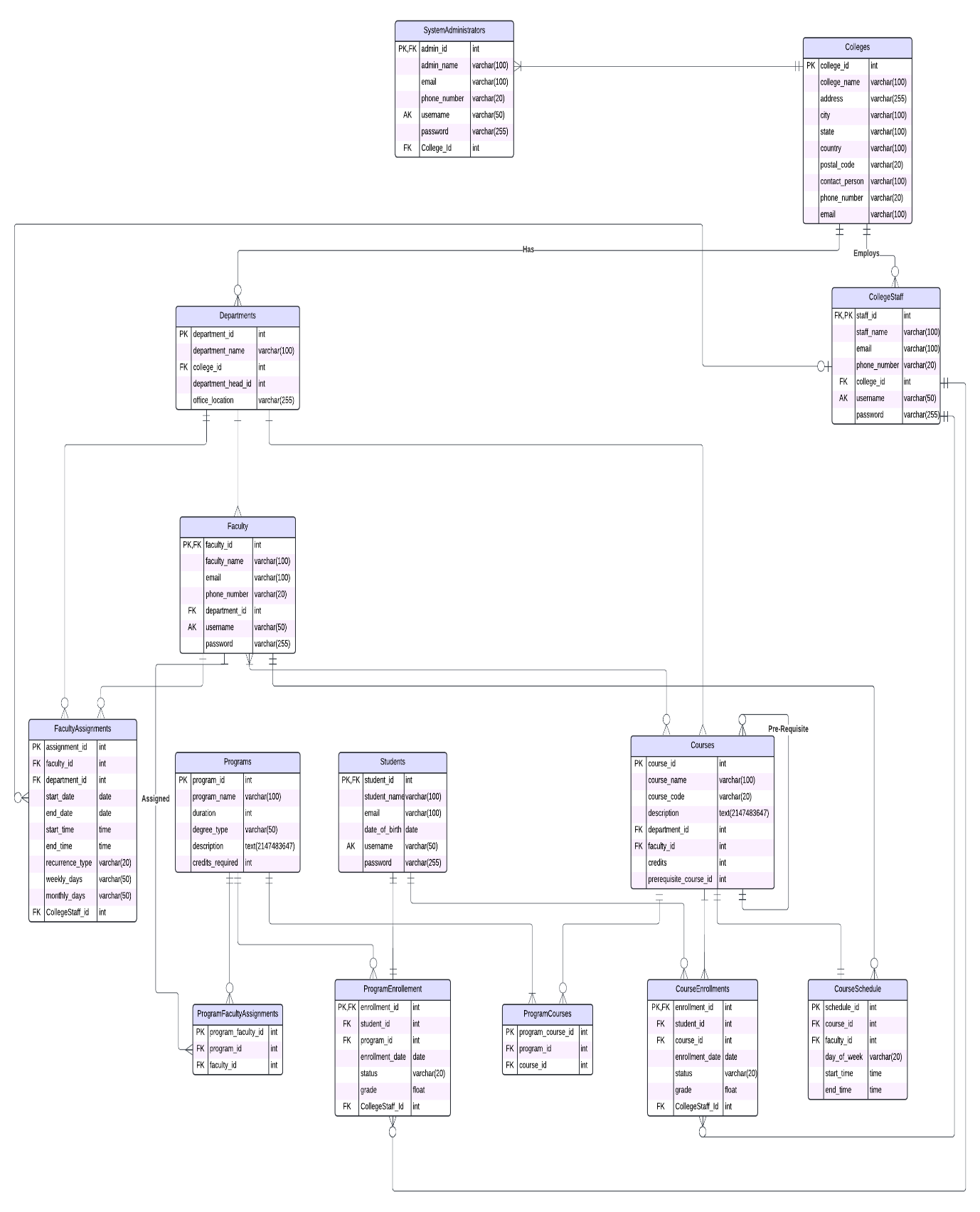
[2.1.2 Deletion Anomaly 9](#_Toc164881768)

[2.1.3 Update Anomaly 10](#_Toc164881769)

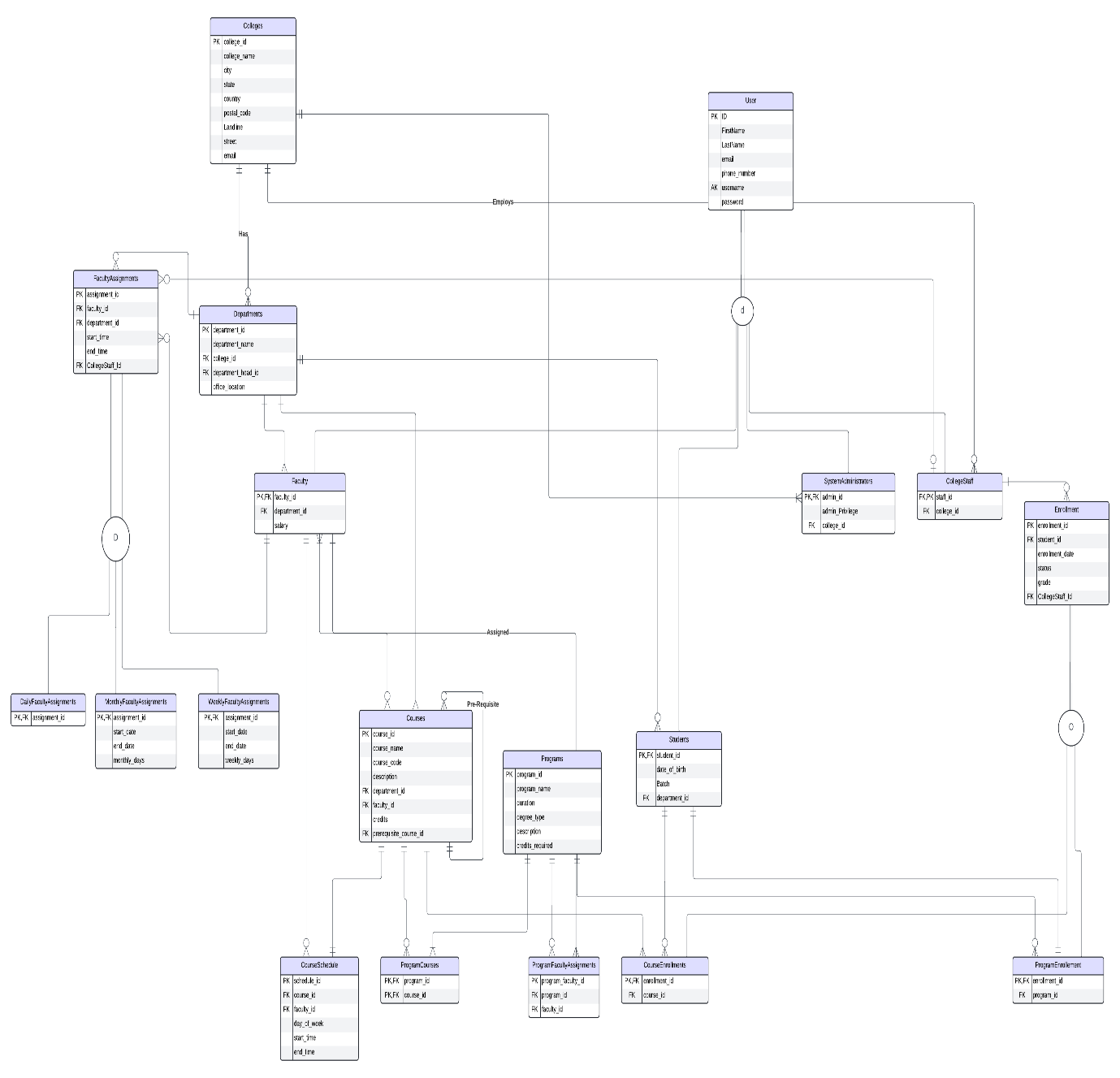
[2.2 Normalization 10](#_Toc164881770)

# 1 Question#1

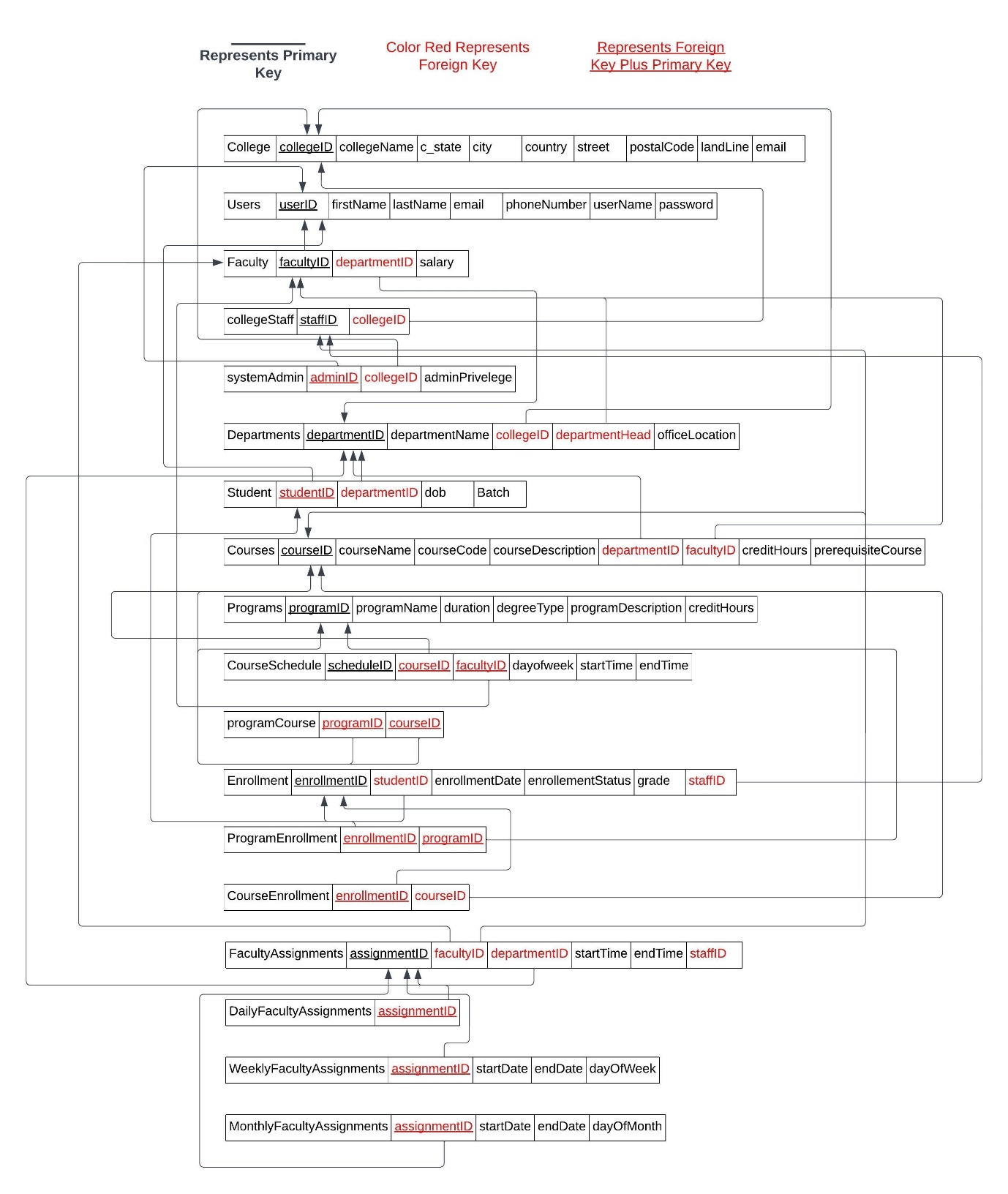
## 1.1 Entity Relation Diagram



## 1.2 Enhanced Entity Relation Diagram

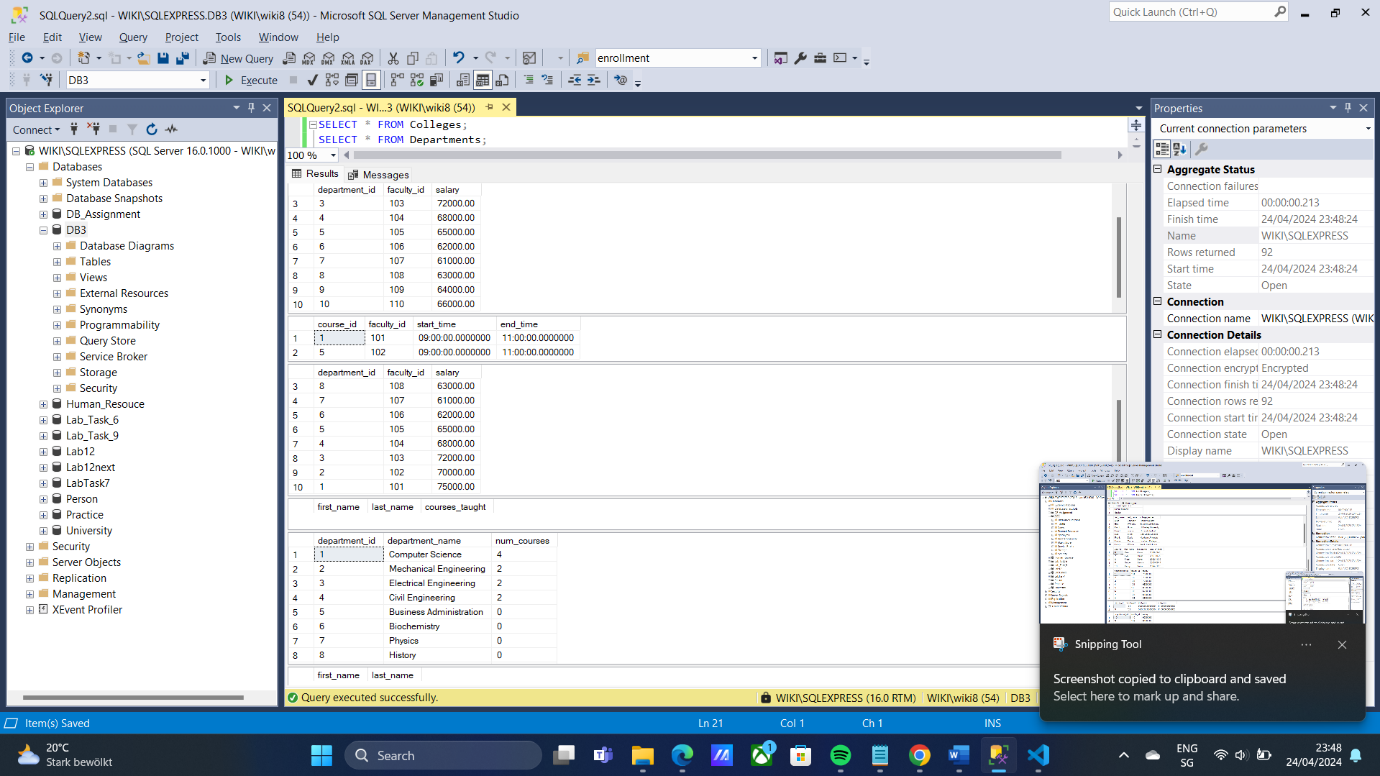
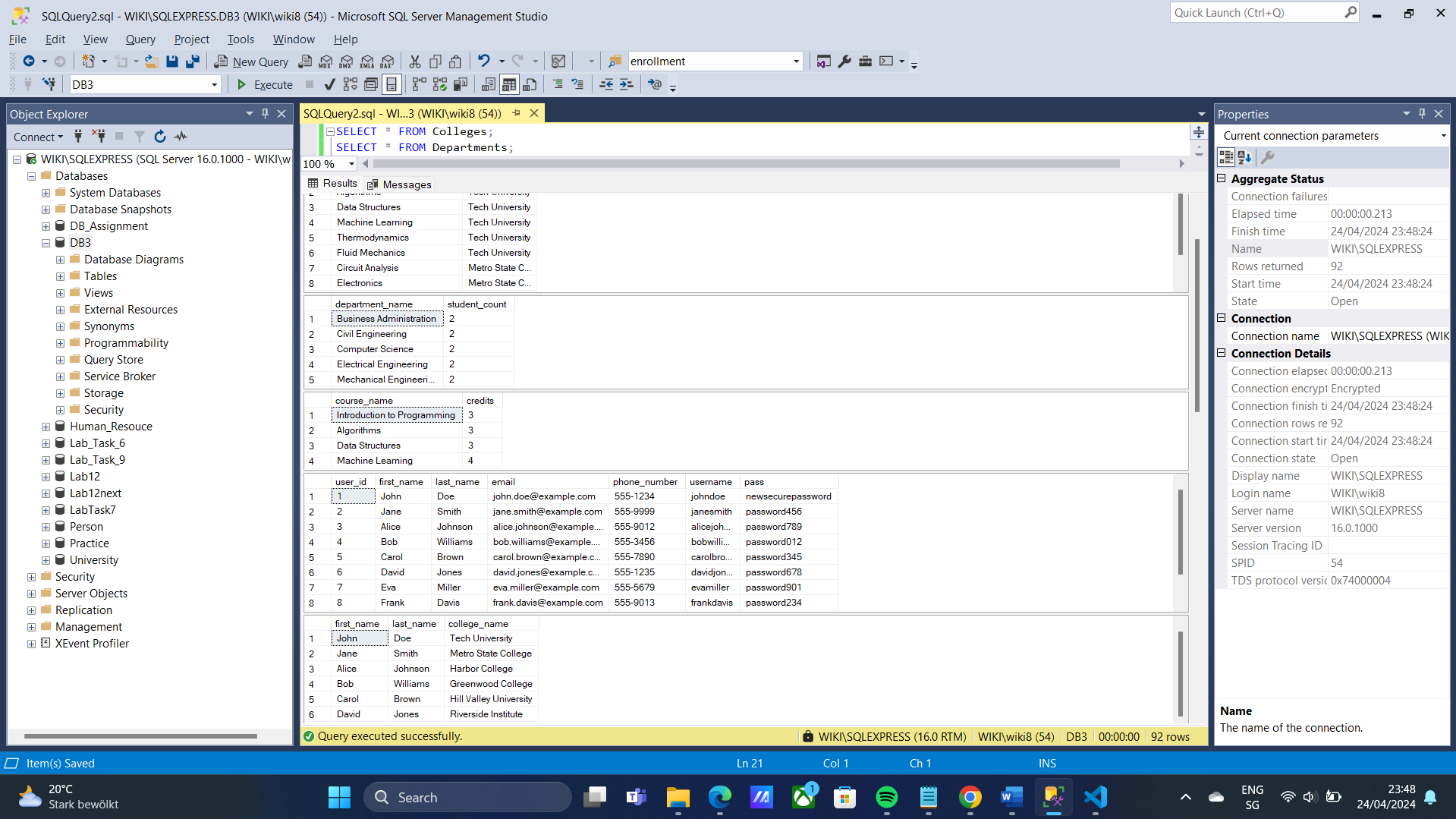
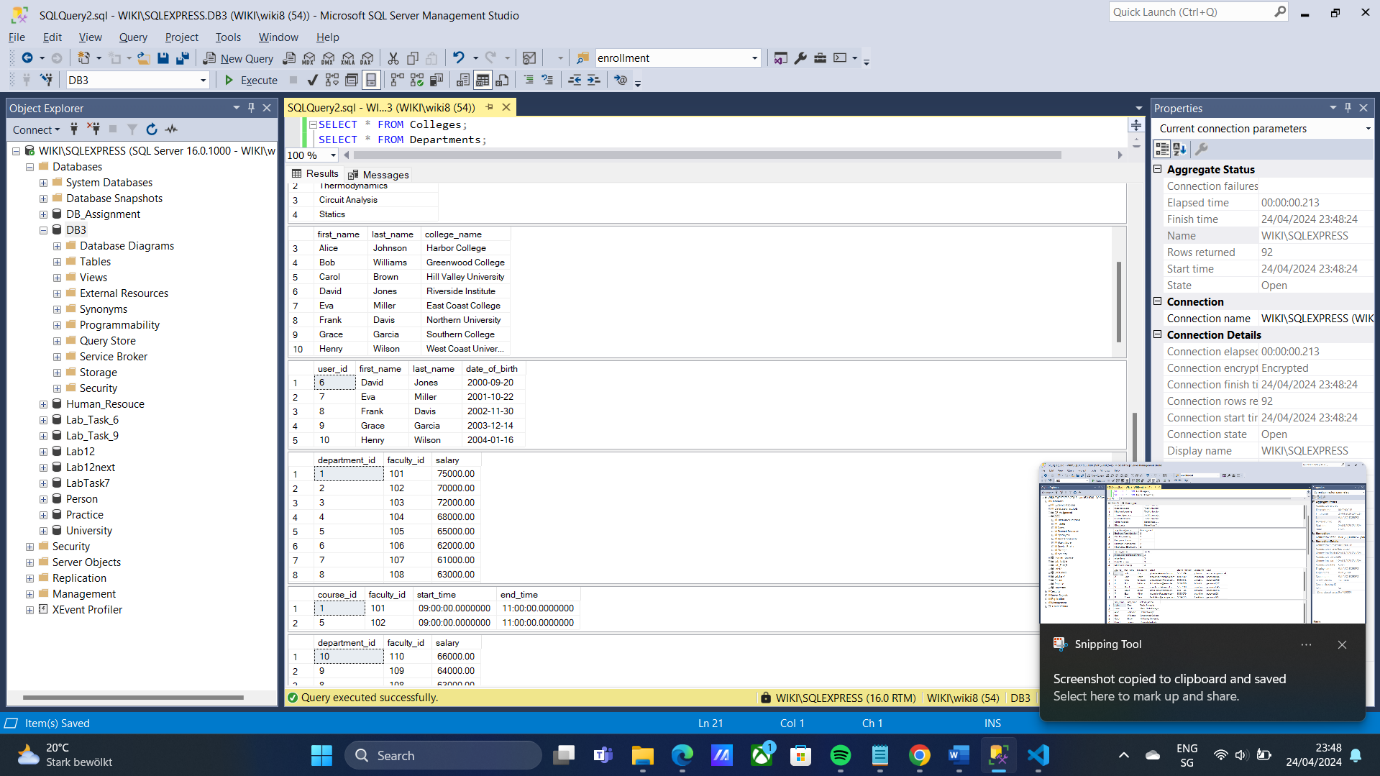


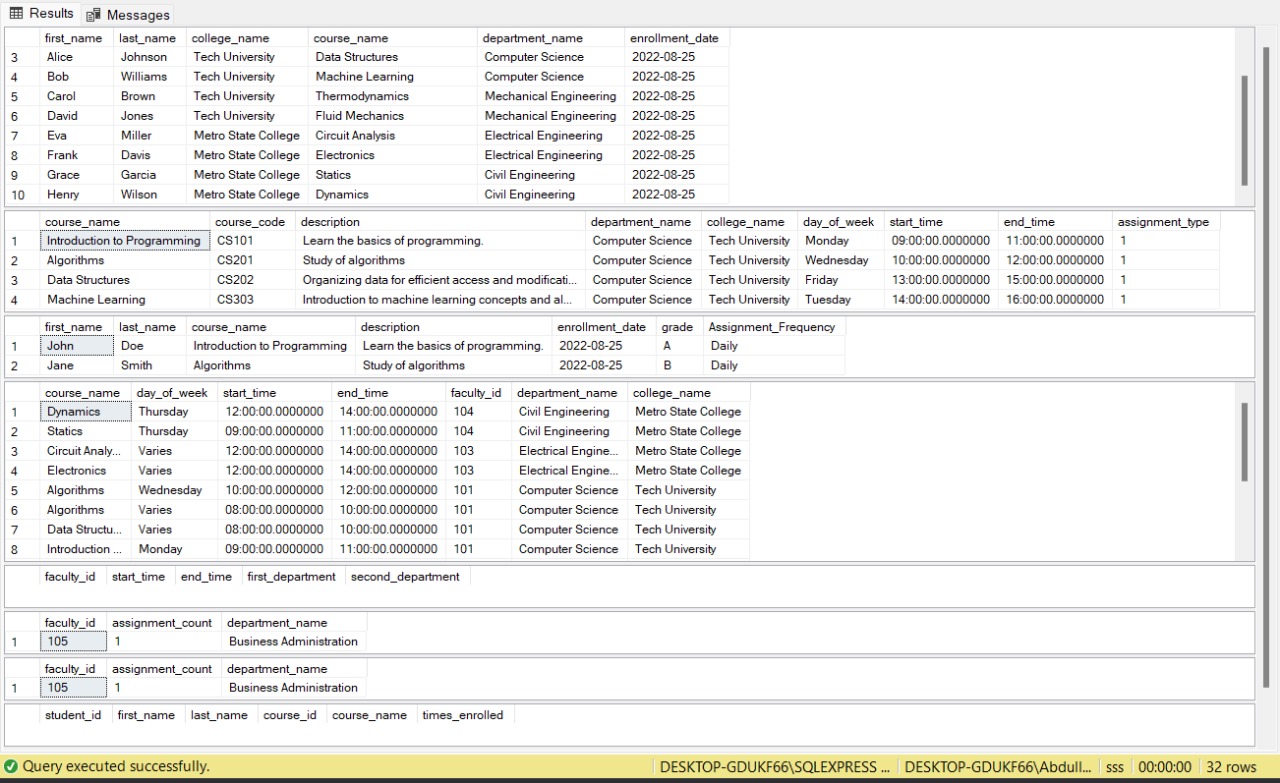
## 1.3 Relational Data Model



## 1.4 Queries

Our Queries:





# 2 Question#2

## 2.1 Examples for Anomalies

### 2.1.1 Insertion Anomaly

An insertion anomaly occurs when some data cannot be inserted into a table without the presence of other data.

1. If we want to insert a student into the table it would not be possible without inserting the course offer id and course id and course details will be repeated. This leads to inefficiency and redundancy.
2. For each course to be added there needs to be a student and course offer id otherwise the course cannot be inserted and in this way student data will be repeated many times leading to redundancy and inefficiency.
3. If we want to offer a course then it cannot be done without inserting data of the student and course leading to inefficiency and redundancy.

### 2.1.2 Deletion Anomaly

A deletion anomaly occurs when one row is deleted and it leads to deletion of other rows also.

1. If a course being offered by course offer id is decided to be no longer offered and removed by let’s say the HOD then all the rows with the course offer id leading to loss of data of students and courses.
2. If a course is decided to be removed from the program and decided to be never offered again let’s say by the HEC then the student data and historical data will be lost.
3. If suppose in summer semester courses were offered but nobody took any courses then deleting the semester data will lead to loss of other data in table.

### 2.1.3 Update Anomaly

An update anomaly occurs when a modification is made in one row and as a result multiple rows are affected.

1. If the name of course is changed as a result all the rows with the course name will be affected.
2. If a student changes his city as a result all the rows with his data will be affected.
3. If by mistake at insertion time the course offered year was entered wrong and now has to be corrected so as a result multiple rows will be affected.

## 2.2 Normalization

**1st Normal Form:**

Requirements:

1. No multivalued attribute must exist if so, they will be broken and moved to a new table and a foreign key will be set in the original first table. (All fields scalar values)
2. A primary key must be defined based on the condition it could be composite primary key keeping minimum number of columns possible.

Composite Primary Key: Student\_Id, Course\_Offer\_Id

Reason: As student id will always be unique with the combination of course offer id because a student cannot take same course offer id more than once.

Now 1NF achieved.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Student\_  Id | Student\_  City | Student\_  Name | Course\_  Offer\_Id | Course\_  Semester | Course\_  Offered\_Year | Course\_  Grade | Course\_  Id | Course\_  Name |

2NF:

Requirements:

1. Remove partial dependencies by making a new table if the primary key is composite and the primary key of new table will be foreign key in original old table.

Functional Dependencies:

1. {Student\_Id} -> {Student\_City}
2. {Student\_Id} -> {Student\_Name}
3. {Course\_Offer\_Id} -> {Course\_Semester}
4. {Course\_Offer\_Id} -> {Course\_Offered\_Year}
5. {Course\_Offer\_Id} -> {Course\_Id}
6. {Course\_Offer\_Id} -> {Course\_Name}
7. 1. {Student\_Id, Course\_Offer\_Id} -> {Course\_Grade}

We have named the original table as Enrollment which has student id and course offer id as both primary and foreign keys coming from respective tables.

In 2NF two new tables have been created each with their primary key underlined.

Enrollment

|  |  |  |
| --- | --- | --- |
| Student\_Id  ------------- | Course\_Offer\_Id  -------------------- | Course\_Grade |

Student

|  |  |  |
| --- | --- | --- |
| Student\_Id | Student\_City | Student\_Name |

Course\_Offer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Course\_Offer\_Id | Course\_Semester | Course\_Offered\_Year | Course\_Id | Course\_Name |

Now 2NF achieved.

3NF:

Requirements:

1. Remove transitive dependencies meaning no attribute depends on a non-primary key attribute which depends on a primary key. If so, the attributes which depend on the non-primary key attribute will be moved to a new table where the non-primary key attribute in the old table will be a primary key in the new table and foreign key in the old table.

The course id and course name attributes were moved to a new table and course id was made primary key in the new table namely Course. In the course offer table, the course id attribute is now a foreign key linked to Course table.

Enrollment

|  |  |  |
| --- | --- | --- |
| Student\_Id  ------------- | Course\_Offer\_Id  -------------------- | Course\_Grade |

Student

|  |  |  |
| --- | --- | --- |
| Student\_Id | Student\_City | Student\_Name |

Course\_Offer

|  |  |  |  |
| --- | --- | --- | --- |
| Course\_Offer\_Id | Course\_Semester | Course\_Offered\_Year | Course\_Id  ------------ |

Course

|  |  |
| --- | --- |
| Course\_Id | Course\_Name |

Now 3NF achieved.