

CSCI 5408

DATA MANAGEMENT AND

WAREHOUSING

ASSIGNMENT - 1

Banner ID: B00981016

GitLab Link: [GitLab Assignment 1](#)

Table of Contents

Problem – 1:	3
Entities, Attributes and Justification	3
Initial Entity-Relationship diagram	5
Identification Of Design Issues and Modifications	5
Final Entity-Relationship diagram	6
Enhanced Entity-Relationship diagram	6
Problem – 2:	8
Design principles	8
Evidence of Testing and Testcases	8
Task – A	8
Task – B1	9
Task – B2	9
Task – C	10
References	12

Problem – 1:

Entities, Attributes and Justification

Entity	Attributes	Justification
city_university	uni_id,name, contact_info, country	Represents the central institution itself which has educational programs and related services.
building	address, name, count_of_rooms floors	Captures the physical infrastructure of the university like classrooms, cafeteria, auditorium etc.
department	dept_id, dept_name, description, head_staff	Represents academic units within the university focusing on specific disciplines and offering related programs.
events	event_id, event_name, description, event_location	Represents various activities organized by the university or student groups, fostering engagement and community building.
application	app_id, status, applicant_country, applicant_name, applicant_email	Represents the process of submitting a request for admission to a program within the university.
library	lib_address, resources, opening_time, closing_time, services_offered,	Represents the central repository for books, journals, and other information resources supporting teaching and research.
career_service_center	location, services_offered, working_hours	Provides guidance and resources to students for career exploration, job search, and professional development.
job_board	job_id, job_title, description, requirements	Facilitates connecting employers with potential student candidates seeking job opportunities.
research	research_id, title, description, domain	Represents ongoing scholarly activities conducted by faculty and staff, contributing to knowledge creation.
laboratory	lab_id, location, availability, equipment_availability	Specialized facilities equipped for conducting experiments and research activities.
alerts	Alert_id, title, description, urgency, status	Provides a communication channel for sending important information and notifications to various stakeholders.
staff	staff_id, department, name, age	Represents university employees supporting various administrative, operational, and technical functions.
degree	degree_id, name, level	Represents academic qualifications awarded upon successful completion of a program, signifying mastery in a specific field.

program	program_id, program_name, program_duration, specialization	Represents a planned course of study leading to a specific degree, consisting of various courses and learning experiences.
course	course_id, course_name, domain, credit_hours	Represents individual units within a program focusing on specific topics and delivering essential knowledge and skills.
tuition	tuition_id, duration, fee_amount	Represents the financial fees charged by the university for attending a program.
alumni	alum_id, alumni_name, alumni_contact_info, alumni_grad_year, alumni_program	Represents individuals who have graduated from the university, maintaining an ongoing connection with the institution.
scholarship	scholarship_id, name, max_amount_sanctionable, eligibility_criteria	Represents financial aid awarded to students based on merit, need, or specific criteria, supporting their academic pursuits.
student	student_id, name, age	Represents individuals currently enrolled in the university's programs, pursuing educational qualifications and development.
housing	unit_number, rent, address, is_shared	Represents accommodation option chosen by the students

Table 1: Entities and attributes used for conceptual model and their justification

Initial Entity-Relationship diagram

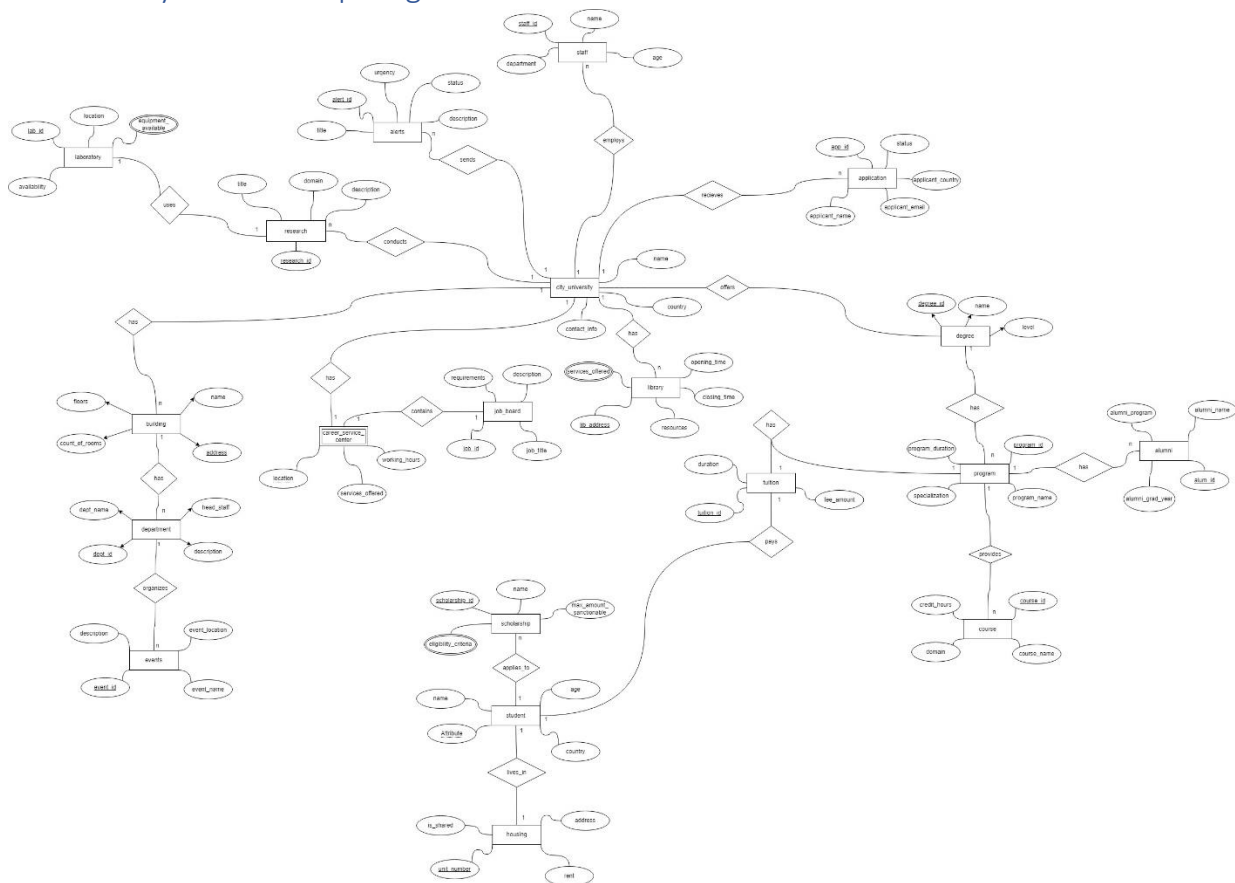


Figure 1: Initial ER diagram

Link for the full image: [ERD Initial](#)

Identification Of Design Issues and Modifications

Issue – 1: A chasm trap has been identified within the Degree, Program, and Course entities. In the existing ERD, while Degree is associated with Program, and Program offers Course, there lacks a clear definition of the relationship where Course also pertains to Degree, thereby creating ambiguity. To fix this issue, we can introduce a new relationship between Course and Degree entities, clarifying the connection between them. This relationship can be established as "course belongs to degree," representing a many-to-one (N:1) relationship, where multiple courses are affiliated with a single degree.

Issue – 2: The alumni entity in the ER diagram is an ever-changing entity which relies on historical data of students who graduate from the university. So, this entity has the issue of time-variant data. This can be resolved by maintaining history of alumni in a separate entity.

Final Entity-Relationship diagram

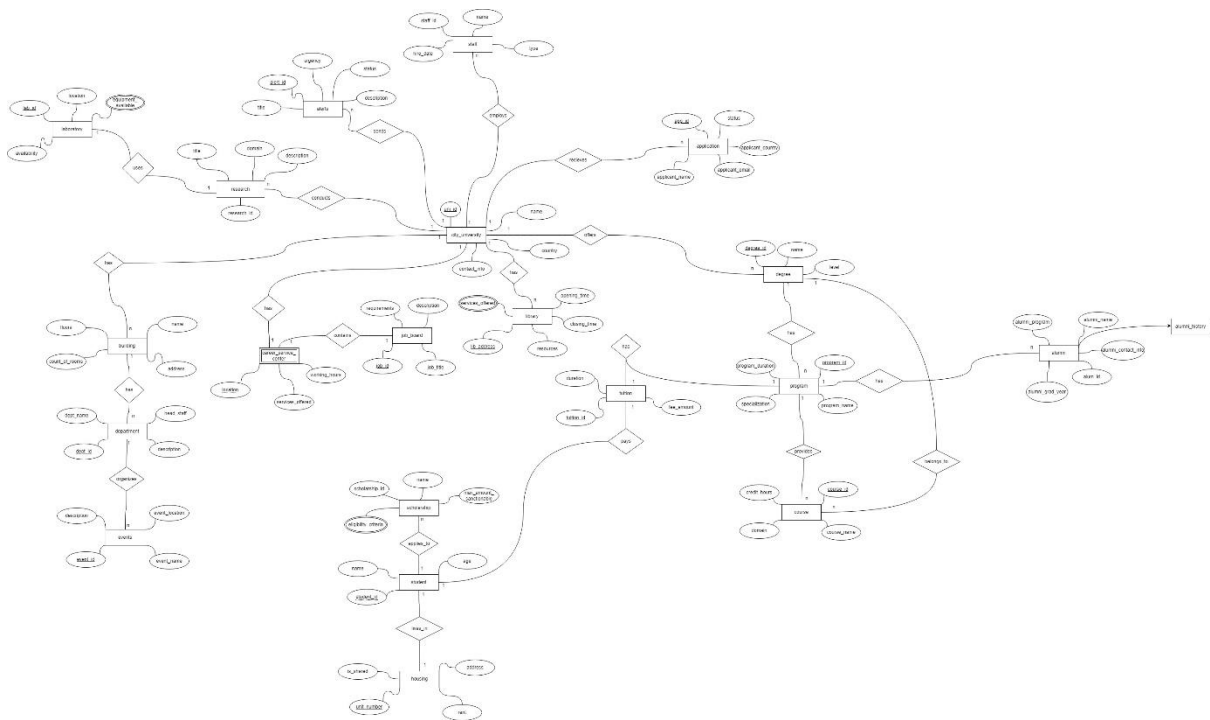


Figure 2: ER diagram after fixing design issues

Link to full image: [ERD_Final](#)

Enhanced Entity-Relationship diagram

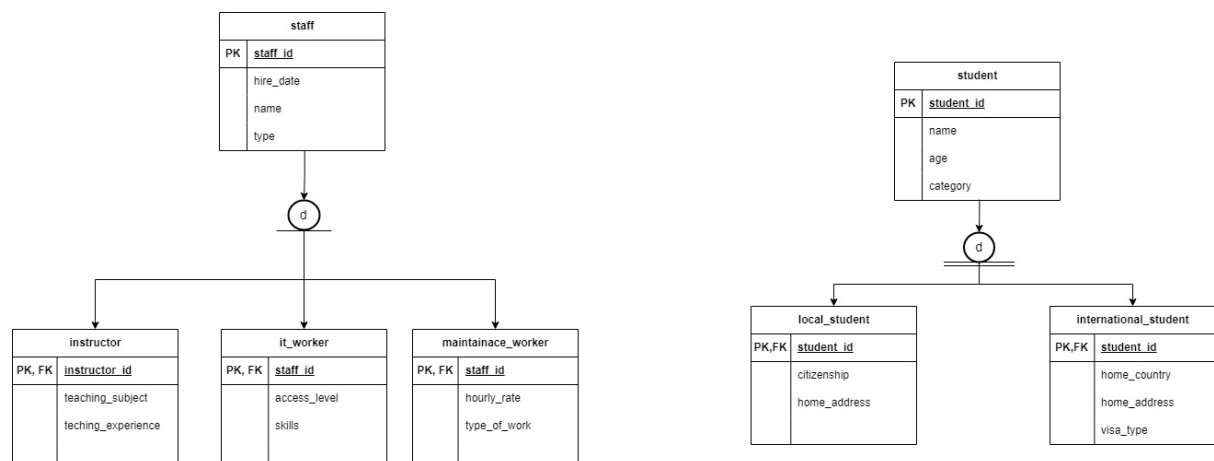


Figure 3: Extended/Enhanced Entity-Relationship Diagram

Link for the full image: [Final EERD](#)

In the final ER diagram, it is possible to extend entities such as staff and student a further. This extension of few entities by adding subtypes leads to creation of specialization hierarchy. In this case of university, student can be sub divided into “local_student” and “international_student”, since a student can either be a local or international, disjoint and complete participation constraints apply here. Staff entity can be further divided into subtypes “instructor”, “it_worker” and “maintainance_worker”, since this not an exhaustive list and one type of staff cannot be in other staff role, disjoint and partial participation constraints apply here.

Problem – 2:

Design principles

The code that is written to implement a light weight database management system (DBMS) is built upon some standard coding principles which helps to write clean, organized and easy-to-change code. The core principles the were followed to solve this problem are Object oriented principles and few of the SOLID principles.

Principle	Description
Single Responsibility Principle	A class (a chunk of code) should do one thing and one thing only.
Open-Closed Principle	Code should be easily extended (add new features) without changing what's already there.
Liskov Substitution Principle	If you're using a parent class, you should be able to swap in a child class without breaking anything.
Interface Segregation Principle	Break down big, bulky interfaces (code contracts) into smaller, more specific ones.
Dependency Inversion Principle	Code should depend on ideas (interfaces) and not concrete details.

Table 2: SOLID design principles

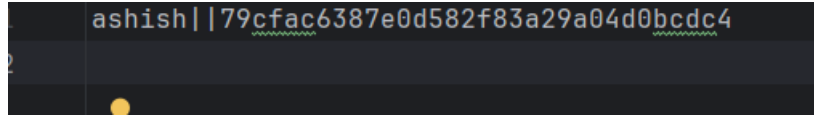
Evidence of Testing and Testcases

Task – A

```
1.Login      2.Sign up
1
Enter your username: ashish
Enter your password: kumar
Verify by entering the captcha: 1234
1234
wrong credentials!
1.Login      2.Sign up
2
Enter username: ashish
Enter password: kumar
Verify by entering the captcha: 1234
1234
DB created: D:\MACS\Term1\DMWA\Assignment - 1\Task2\ashish_db.txt
Username and password stored successfully.
Sign up successful! You can login Now!
1.Login      2.Sign up
1
Enter your username: ashish
Enter your password: kumar
Verify by entering the captcha: 1234
1234
Login successful!
1.Query      2.Transcation  3.Exit
```

Figure 4: Testing Authentication functionality

Task – B1

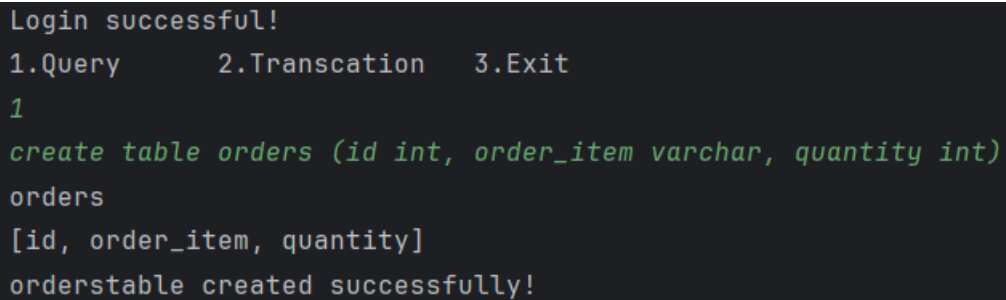


```
ashish||79cfac6387e0d582f83a29a04d0bcd4
```

Figure 5: Storing of user details in text file using custom delimiter

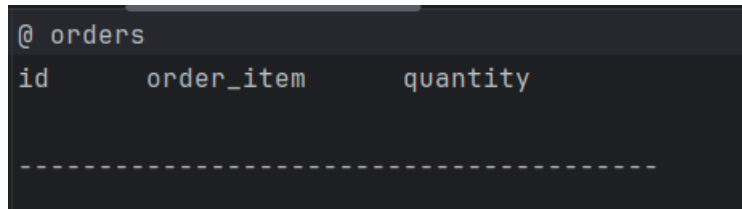
Task – B2

CREATE command



```
Login successful!  
1.Query      2.Transcation  3.Exit  
1  
create table orders (id int, order_item varchar, quantity int)  
orders  
[id, order_item, quantity]  
orderstable created successfully!
```

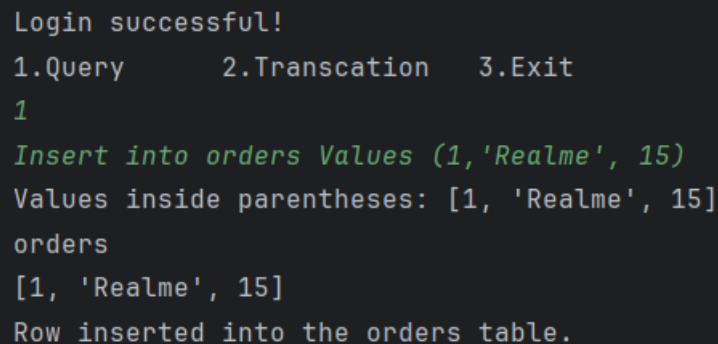
Figure 6: User inputs Create command with column names



```
@ orders  
id      order_item    quantity  
-----
```

Figure 7: Table gets created in text file with custom delimiters @ and -----

INSERT command



```
Login successful!  
1.Query      2.Transcation  3.Exit  
1  
Insert into orders Values (1,'Realme', 15)  
Values inside parentheses: [1, 'Realme', 15]  
orders  
[1, 'Realme', 15]  
Row inserted into the orders table.
```

Figure 8: User inputs insert command with desired values

```
@ orders
```

id	order_item	quantity
1	'Realme'	15

Figure 9: Row gets inserted into table with given values in the insert command

SELECT command

```
Login successful!
1.Query      2.Transcation  3.Exit
1
select * from orders
id      order_item    quantity
1       'Realme'        15
2       'Xiaomi'        23
```

Figure 10: User inputs Select command and the rows from the table are displayed

Task – C

```
Login successful!
1.Query      2.Transcation  3.Exit
2
Use 'Begin Transaction' to start and 'End Transaction' to complete
Begin Transaction
create table cart (cart_id int, cart_items varchar, quantity int)
Insert into cart Values (1,'Realme', 15)
Insert into cart Values (1,'iPhone', 55)
Insert into cart Values (1,'Google Pixel', 13)
commit

create table laptops (id int, name varchar, quantity int)
rollback
End Transaction
```

Figure 11: Transaction input with insert, create, commit and rollback commands

```
@ cart
cart_id  cart_items  quantity
1        'Realme'    15
1        'iPhone'    55
1        'Google Pixel' 13
-----
```

Figure 12: Output of the given transaction statements (rollback was performed and new table was not created)

References

[1] Educative, "What are the SOLID principles in Java?", educative.io.
<https://www.educative.io/answers/what-are-the-solid-principles-in-java> (Accessed: Feb 26, 2024)