CS 411 Project Stage 2 Report – College Employment & Salary Analysis Platform

Group50-Hajimi

1. Entity Descriptions and Assumptions

Entity	Description / Assumption
User	This entity is used to present user information. Each user can bookmark programs and compare them. Each user can have preferences on job, location and major.
University	This entity is used to present university information. Each university can provide multiple programs. Each university has its Name, Location, and Region. Each university has a unique UniversityID
Major	This entity is used to present major information like Economics, CS, and so on. Each major can be provided by multiple programs. Each major has a unique major ID.
Program	This entity is used to present what programs each university provides. Each program is related to a major. Each program must be provided by one university, and each university can provide multiple programs. Each program can have a unique ProgramID.
Job	This entity is used to present the job information. Each job has a unique JobID. Each job contains a job title, company, company location, and average salary.
Comparison	This entity is used to store the comparisons each user selected to compare. Each comparison stores two selected programs by using ProgramID1 and ProgramID2. Each comparison must be assigned to one user, and each user can create multiple comparisons. Each comparison has a unique ComparisonID. We model comparison as an entity rather than a relationship because each comparison represents a distinct user-created record that stores not only the pair of programs being compared but also additional attributes. Each comparison includes a note attribute where

	users can write their thoughts about the comparison. Users can revisit these saved comparisons later to review or update their thoughts.

2. Relationship Explanations

Relationship	Type	Explanation
User-Comparison	1-to-Many	A user can create many comparisons, but each comparison belongs to one user based on UserId.
User-Program (Bookmarks)	Many-to-Many	Users can bookmark many programs, and each program can be bookmarked by many users. We will implement it by creating a Bookmark relation.
University-Program	1-to-Many	Each university offers many programs, but each program is offered by exactly one university.
Major-Program	1-to-Many	Each major can appear in many programs, but each program is tied to one major.
Major-Job (relates-to)	Many-to-Many	A major can lead to multiple job types, and a job can be related to multiple majors. We will implement it via the relates-to relation.

Job-User (prefers)	Many-to-Many	Users can express interest in multiple jobs, and each job can be preferred by multiple users. We will implement it via the prefers relation.

3. Normalization Justification

Our database design adheres to 3NF, ensuring data integrity and minimal redundancy:

User
Primary Key: UserID
Functional Dependencies (FDs): UserID → Username, Email, PasswordHash, PreferredMajor, PreferredLocation, PreferredJob
1NF: All attributes are atomic.
2NF: No partial dependency since the key is a single attribute.
3NF: No transitive dependency among non-key attributes.
Conclusion: User is in 3NF (and BCNF).

University
Primary Key: UniversityID
FDs: UniversityID → Name, Location, Region, Tuition
1NF: Attributes are atomic.
2NF: No partial dependency since the key is a single attribute.
3NF: No transitive dependency exists.
Conclusion: University is in 3NF (and BCNF).

Major
Primary Key: MajorID
FDs: MajorID → MajorName, Field
1NF: All attributes are atomic.
2NF: Single-attribute key, no partial dependency.
3NF: No transitive dependency.
Conclusion: Major is in 3NF (and BCNF).

Job
Primary Key: JobID
FDs: JobID → JobTitle, Company, Location, AvgSalary
1NF: Attributes are atomic.
2NF: No partial dependency.
3NF: No transitive dependency among non-key attributes.
Conclusion: Job is in 3NF (and BCNF).

Program
Primary Key: ProgramID
FDs: ProgramID →Name, UniversityID, MajorID, MedianSalary, DegreeType
1NF: Attributes are atomic.
2NF: No partial dependency since the key is a single attribute.
3NF: No transitive dependency among non-key attributes.
Conclusion: Program is in 3NF (and BCNF).

Comparison
Primary Key: ComparisonID
FDs: ComparisonID → UserID, ProgramID1, ProgramID2, NoteFromUser
1NF: All attributes are atomic.
2NF: No partial dependency since the key is a single attribute.
3NF: No transitive dependency.
Conclusion: Comparison is in 3NF (and BCNF).

Bookmark
Primary Key: (UserID, ProgramID)
FDs: (UserID, ProgramID) → (none)
1NF: Attributes are atomic.
2NF: No partial dependency—both attributes form the full key.
3NF: No transitive dependency.
Conclusion: Bookmark is in 3NF (and BCNF).

MajorJob
Primary Key: (MajorID, JobID)
FDs: (MajorID, JobID) → (none)
1NF: Attributes are atomic.
2NF: No partial dependency.
3NF: No transitive dependency.
Conclusion: MajorJob is in 3NF (and BCNF).

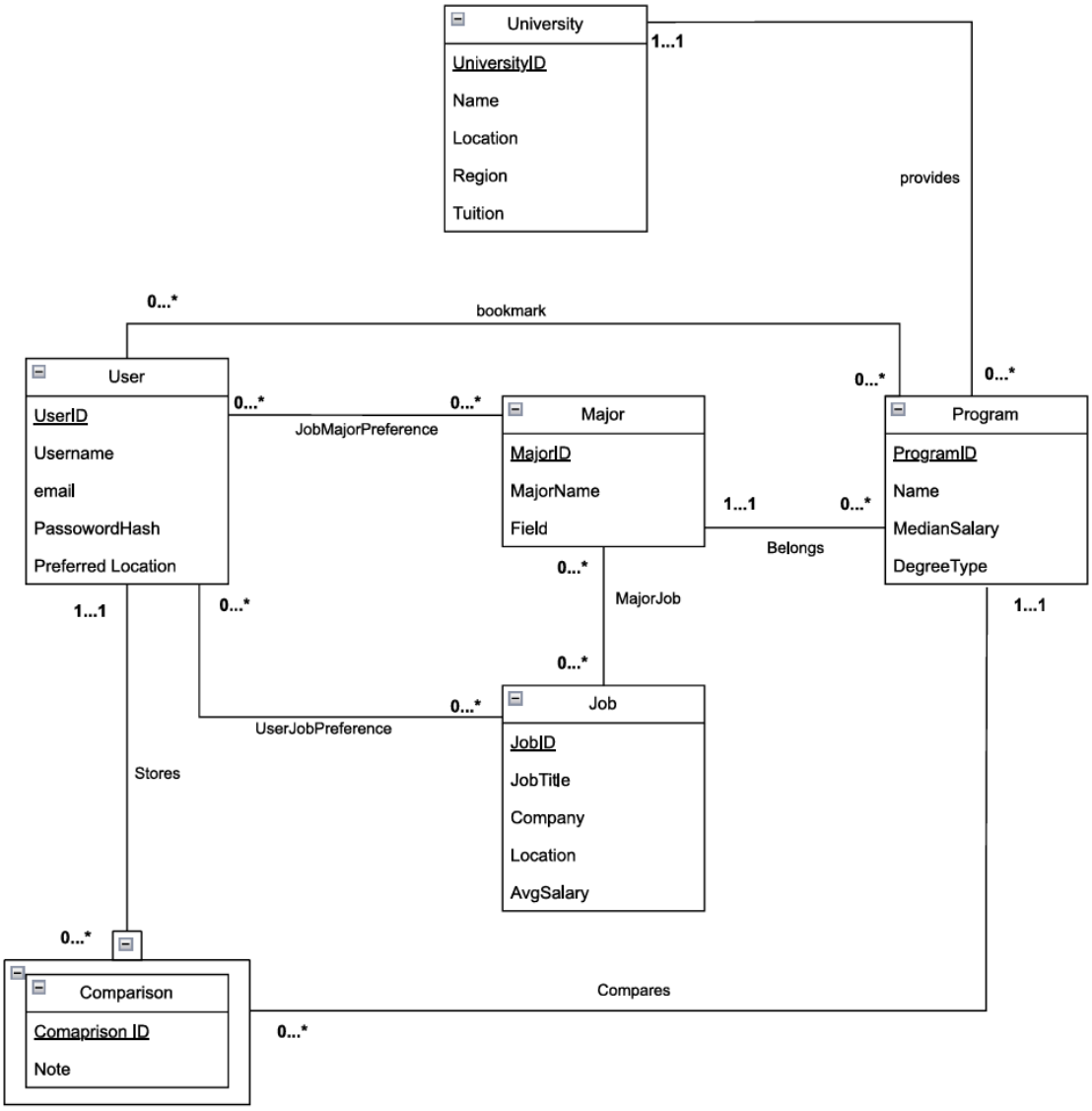
UserJobPreference
Primary Key: (UserID, JobID)
FDs: (UserID, JobID) → (none)
1NF: Attributes are atomic.
2NF: No partial dependency.
3NF: No transitive dependency.
Conclusion: UserJobPreference is in 3NF (and BCNF).

4. Logical Design Relational Schema

```
 User(
      UserID: INT [Primary Key],
      Username: VARCHAR(255),
      Email: VARCHAR(255),
      PasswordHash: VARCHAR(255),
      PreferredMajor: INT [Foreign Key to Major.MajorID],
      PreferredLocation: VARCHAR(255),
      PreferredJob: INT [Foreign Key to Job.JobID])

University(
      UniversityID: INT [Primary Key],
      Name: VARCHAR(255),
      Location: VARCHAR(255),
      Region: VARCHAR(255),
      Tuition: INT)

Major(
      MajorID: INT [Primary Key],
      MajorName: VARCHAR(255),
      Field: VARCHAR(255))

Program(
```

```
      ProgramID: INT [Primary Key],
      Name: VARCHAR(255)
      UniversityID: INT [Foreign Key to University.UniversityID],
      MajorID: INT [Foreign Key to Major.MajorID],
      MedianSalary: INT,
      DegreeType: VARCHAR(255))

Job(
      JobID: INT [Primary Key],
      JobTitle: VARCHAR(255),
      Company: VARCHAR(255),
      Location: VARCHAR(255),
      AvgSalary: INT)

Comparison(
      ComparisonID: INT [Primary Key],
      UserID: INT [Foreign Key to User.UserID],
      ProgramID1 [Foreign Key to Program.ProgramID],
      ProgramID2 [Foreign Key to Program.ProgramID]
      NoteFromUser: VARCHAR(255))
Bookmark(
      UserID: INT [Foreign Key to User.UserID] [PK],
      ProgramID: INT [FK to Program.ProgramID] [PK])
```

```
MajorJob(
      MajorID: INT [Foreign Key to Major.MajorID] [PK],
      JobID: INT [Foreign Key to Job.JobID] [PK])

UserJobPreference(
      UserID: INT [FK to User.UserID] [PK],
      JobID: INT [FK to Job.JobID] [PK])

UserMajorPreference(
      UserID: INT [FK to User.UserID] [PK],
      MajorID: INT [FK to Major.MajorID] [PK])
```


University
UniversityID
Name
Location
Region
Tuition
1...1
provides
0...*
bookmark
0...*
0...*
User
UserID
Username
email
PassowordHash
Preferred Location
0...*
JobMajorPreference
0...*
Major
MajorID
MajorName
Field
1...1
0...*
Belongs
Program
ProgramID
Name
MedianSalary
DegreeType
1...1
0...*
MajorJob
0...*
1...1
0...*
0...*
UserJobPreference
Job
JobID
JobTitle
Company
Location
AvgSalary
Stores
0...*
Comparison
Comaprison ID
Note
Compares
0...*