# ER Diagram Normalization and Relational Schema

- **BCNF Normalization:**
  - ➢ Explanation:
    - We choose BCNF over 3NF because, although 3NF preserves dependencies and avoids information loss, it will create redundancies. Given the functional dependencies in our database schema. if we are to convert these functional dependencies into a minimal basis, there will be too many new relations for us to handle. On the other hand, BCNF is able to reduce these redundancies and avoid information loss. Therefore, we choose BCNF as our normalization method.
  - ➢ Process:
    - Unnormalized Relational Schema:
      - Restaurant(RestaurantId:INT[PK], RestaurantName:VARCHAR(255), Stars:REAL, Hours:VARCHAR(255), Category:VARCHAR(127), LocationId:INT[FK to Location.LocationId])
      - Review(RestaurantId:INT[PK, FK to Restaurant.RestaurantId], UserId:INT[PK, FK to User.UserId], Date:DATETIME[PK], Star:REAL, Text:MEDIUMTEXT)
      - User(UserId:INT[PK], Taste:VARCHAR(127), Password:VARCHAR(31), Name:VARCHAR(31))

- Location(LocationId:INT[PK], Neighborhood:VARCHAR(255), Latitude:REAL, Longitude:REAL, PostalCode:VARCHAR(31), State:VARCHAR(15), City:VARCHAR(31))
- Dish(DishId:INT[PK], RestaurantId:INT[PK, FK to Restaurant.RestaurantId], Price:REAL, Name:VARCHAR(127))
- Crime(CrimeId:INT[PK], Count:INT, Type:INT, Cleared:INT, LocationId:INT[FK to Location.LocationId])
- Occurrence(RestaurantId:INT[PK, FK to Restaurant.RestaurantId], CrimeId:INT[PK, FK to Crimes.CrimeId])

■ Functional Dependencies:

- Restaurants:
  - ◆ RestaurantId → RestaurantName, Stars, Hours, Category, Neighborhood
- Reviews:
  - ◆ RestaurantId, UserId, Date → Star, Text
- Users:
  - ◆ UserId → Taste, Password, Name
- Location:
  - ◆ LocationId → Neighborhood, Latitude, Longitude, PostalCode, State, City
- Dishes:
  - ◆ DishId, RestaurantId → Price, Name
- Crimes:

        - CrimeId → Count, Type, Cleared, Neighborhood
    - Normalization:
        - For every relation, every functional dependency satisfies the definition for BCNF, such that:
            - Whenever there is a nontrivial functional dependency $A_1A_2…A_n \rightarrow B$, $A_1A_2…A_n$ is a superkey for R.

❖ **Final Relational Schema:**

- Restaurant(RestaurantId:INT[PK], RestaurantName:VARCHAR(255), Stars:REAL, Hours:VARCHAR(255), Category:VARCHAR(127), LocationId:INT[FK to Location.LocationId])
- Review(RestaurantId:INT[PK, FK to Restaurant.RestaurantId], UserId:INT[PK, FK to User.UserId], Date:DATETIME[PK], Star:REAL, Text:MEDIUMTEXT)
- User(UserId:INT[PK], Taste:VARCHAR(127), Password:VARCHAR(31), Name:VARCHAR(31))
- Location(LocationId:INT[PK], Neighborhood:VARCHAR(255), Latitude:REAL, Longitude:REAL, PostalCode:VARCHAR(31), State:VARCHAR(15), City:VARCHAR(31))
- Dish(DishId:INT[PK], RestaurantId:INT[PK, FK to Restaurant.RestaurantId], Price:REAL, Name:VARCHAR(127))
- Crime(CrimeId:INT[PK], Count:INT, Type:INT, Cleared:INT, LocationId:INT[FK to Location.LocationId])
- Occurrence(RestaurantId:INT[PK, FK to Restaurant.RestaurantId], CrimeId:INT[PK, FK to Crimes.CrimeId])