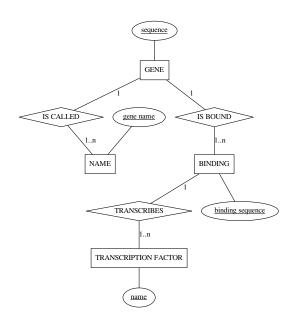
- 1. What is the difference between a "Data Warehouse" and a "Federated Database"?
  - Data Warehouse: Is a central repository for all or significant parts of the data that an enterprise's various business systems collect. Typically, a data warehouse is housed on an enterprise mainframe server or increasingly, in the cloud.
  - Federated Database: Is a database management system, which transparently maps multiple autonomous database systems into a single federated database. This means that multiple databases appears as one database to the user.
- 2. Name and describe at least four typical advantages of DBMS based data storage over file system based data management. Give one real world example for each.
  - (a) A locking mechanism for concurrent access (If users access the same item in the database there is no RW or WR errors.)
  - (b) The ability to swiftly recover from crashes and errors, including restartability and recoverability (If the System Crashes due to for example power failures the Logging system recovers know changes hence reducing loss)
  - (c) Robust data integrity capabilities (Data is the same from all access points).
  - (d) Logging and auditing of activity (If changes are discarded before saving the system can re-roll.)
- 3. Design and paint an Entity Relationship (ER) diagram for a small database dedicated to store gene regulatory interactions of a bacterium. The following entities are necessary: genes (with ID and one or more (!) names), transcription factors (with ID and one name) and binding sites (with binding sequence of 10...30 bp length). Include the following relations:
  - (a) A transcription factor is encoded by one gene.
  - (b) A transcription factor may regulate a gene by binging to one or more binding sites.



- 4. Download and install MySQL. Design and write DDL code for MySQL that implements the in exercise 3 designed database model. Afterwards, provide DML code that fills the database with arbitrary data. Finally, give the corresponding SQL code for the following two queries:
  - (a) A list of all genes that are regulated by two ore more transcription factors.

```
SELECT * FROM GENE WHERE

(
SELECT gene_id FROM TRANSCRIPTION

WHERE COUNT (gene_id) > 1

5 ) = id;
```

(b) A list of all gene regulations. Output style: transcription factor, target gene, binding site

```
SELECT * FROM TRANSCRIPTION
ORDER BY TRANSCRIPTION.name, GENE.name, BINDING_SITE.id
```

Our DDL and DML SQL code is in the appendix of this document.

- 5. Name and describe one major advantage and one major disadvantage of an ontological, generalized data structure over "standard", explicit data structures. When would you use which approach. Explain why (give example if necessary).
  - Advantage: More flexible when adding new concepts to the database.
  - **Dis-advantage:** Hard to search due to structure. Might miss information.

## Which approach?

- Ontological: If I would need a quick overview, where I don't know if there is any structure in the data or what the structure is.
- Standard: Once a structure of the data has revealed itself.

## A SQL code

```
Definitions
       DROP DATABASE IF EXISTS bioInfo;
       CREATE DATABASE bioInfo; -- create new database
       USE bioInfo; -- use that new database
  8
       CREATE TABLE GENE (
         g_sequence CHAR(32) NOT NULL,
id INT PRIMARY KEY AUTO_INCREMENT
10
12
       CREATE TABLE TRANSCRIPTION_FACTOR (
13
         tf_name CHAR(32) NOT NULL,
id INT PRIMARY KEY AUTO_INCREMENT,
14
15
16
         FOREIGN KEY (b_id) REFERENCES (BINDING.id)
17
18
       CREATE TABLE BINDING (
\frac{19}{20}
         id INT PRIMARY KEY AUTO_INCREMENT,
\frac{1}{21}
         b_sequence CHAR(32) NOT NULL,
22 \\ 23 \\ 24 \\ 25 \\ 26 \\ 27
         FOREIGN KEY (g_id) REFERENCES (GENE.id)
       CREATE VIEW regulation AS
         SELECT g_id AS id,tf_name,b_sequence,g_sequence FROM TRANSCRIPTION_FACTOR AS A JOIN BINDING AS B JOIN GENE AS C
28
29
         ON A.b_id = B.id AND C.id = B.g_id;
       CREATE VIEW is_called AS
30
31
32
33
34
         SELECT g_id AS id, g_name AS gene_name, g_sequence AS gene_sequence FROM GENE AS A JOIN GENE_NAME AS B
         ON A.id = B.g_id;
35
       CREATE VIEW encodes AS
36
         SELECT b_id AS id,g_sequence AS gene_sequence, tf_name as transcription_factor
37
38
39
         FROM GENE AS A JOIN TRANSCRIPTION_FACTOR AS B
         ON A.id = B.b_id;
40
       -- Filler data
       INSERT INTO GENE (g_sequence) VALUES ("ATTAT");
INSERT INTO GENE (g_sequence) VALUES ("GTDAT");
41
43
       INSERT INTO GENE (g_sequence) VALUES ("GGADT");
44
       INSERT INTO GENE_NAME (g_name,g_id) VALUES ("k10022",1);
INSERT INTO GENE_NAME (g_name,g_id) VALUES ("1072er",2);
45
46
47
       INSERT INTO GENE_NAME (g_name,g_id) VALUES ("xx06a70",3);
       INSERT INTO TRANSCRIPTION_FACTOR (tf_name,b_id) VALUES ("protein0",1); INSERT INTO TRANSCRIPTION_FACTOR (tf_name,b_id) VALUES ("protein1",2);
49
50
51
52
53
       INSERT INTO TRANSCRIPTION_FACTOR (tf_name,b_id) VALUES ("protein2",3);
       INSERT INTO BINDING (b_sequence,g_id) VALUES ("AATTTAT",1);
INSERT INTO BINDING (b_sequence,g_id) VALUES ("ATTGGTTA",2);
       INSERT INTO BINDING (b_sequence,g_id) VALUES ("DDTTAAG",3);
```