T-Closeness & L-Diversity
From a dataset we can form equivalence classes by collecting records together that match on a set of quasi-identifiers. The distribution of values for any sensitive attributes may differ from the global distribution within any equivalence class. T-closeness is a restriction that the distribution is not more different than some threshold t.

L-diversity guarantees that there will be at least l different values for any sensitive attribute in an e-class. However, the distribution of these values could still be significantly different from the global population and give the attacker the opportunity to learn something. For example, an attacker might learn that people living in South Kensington still suffer from a variety of illnesses but at an increased risk of having a particular ailment.

Unicity and Unique Attack
Uniqueness attacks and unicity attacks are applied to different kinds of data. In a small data dataset an attacker can find so unsensitive information about a target, quasi-identifiers, that when combined together uniquely identifies them in the dataset and allows them to perform a 'uniqueness attack' to discover sensitive information about the target.

A unicity attack is relevant in a big data scenario where all data about an individual is potentially sensitive. If an attacker is able to obtain a portion of a user's data, they are able to perform a unicity attack if they can uniquely identify their targets trace in the dataset.

Unicity and Population Size
Unicity is convex, and decreases rather slowly with population size. Moreover, even if 2 points are not great to identify you, 4 points are more than sufficient.

Hash Function Secure?
the hash algorithms are called secure because, for a given algorithm, it is computationally infeasible (1) to find a message that corresponds to a given message digest, or (2) to find two different messages that produce the same message digest. These algorithms enable the determination of a message's integrity: any change to a message will, with a very high probability, result in a different message digest.

Hash Function Disclose the Salt placement & Length?

Given a sufficiently long salt, you can freely disclose the hash function, length and placement, keeping only the value of the salt secret. This is because disclosing the length and placement only have polynomial increases in the amount of time taken, whereas increasing the length of the salt has an exponential effect on the runtime to produce the lookup table. If the salt is 10,000 characters long, knowing where it’s placed won’t help you.

Secret Formula a Bad Idea?
An attacker could easily find the polynomial using Lagrange interpolation. And degree 2 only needs 3 different points

Malicious in Garbled Circuit
a malicious adversary who constructs the garbled circuit may construct a garbling of a different circuit computing a different function, and this cannot be detected (due to the garbling). In order to solve this problem, many circuits are sent and some of them are opened to check that they are correct while the others are evaluated. This methodology, called cut-and-choose, introduces significant overhead, both in computation and in communication

Malicious in Garbled Circuit Example
A malicious P1 can construct a single incorrect circuit that computes the following function: if the first bit of P2’s input equals 0 then output random garbage; else compute the correct function. Now, if this circuit is not opened (which happens with probability 1/2) and if the first bit of P2’s input equals 0, 1 then P2 will receive a different output in this circuit and in the others. In contrast, if the first bit of P2’s input equals 1 then it always receives the same output in all circuits. Thus, if the protocol instructs P2 to abort if it receives different outputs, then P1 will learn the first bit of P2’s input (based on whether or not P2 aborts).

Garbled Circuit Output Authenticity
how does the protocol prevent Bob from lying about the output to Alice, in the case that Bob doesn't withhold her output altogether? There are several approaches but one I like is to just have Bob present the output wire labels back to Alice. Wire labels have an "authenticity" property: the evaluator only learns one on each wire and cannot guess the complementary one. Hence they act as a kind of "proof" to Alice that this was indeed the correct output.

Comparison between Secret Sharing and Garbled Circuits

Secret Sharing	**Garbled Circuits**
Express function to be computed as an **arithmetic circuit** (i.e., Addition and Multiplication)	Express function to be computed as an "encrypted" Boolean circuit
Better suited for multi-party (3+) outsourced service setups	More efficient for 2-party joint processing setups
High communication costs (i.e., when using multiplication in BGW protocol)	
Any secret sharing scheme can theoretically to be used, but prefer homomorphic encryption	Assume underlying encryption scheme is secure
For honest-but-curious parties (i.e., semi-honest passive), tolerant <n/2 parties. For malicious parties (i.e., Byzantine, Active), <n/3 corrupt parties	

Information Theoretically Secure: System cannot be broken even if adversary ahs unlimited computational power (i.e., Perfect Security / Unconditional Security)
Computationally Secure: Adversary would require an unreasonably large amount of computational power to break the system
Kerchoff's Principle: Algorithm is public knowledge. *Security through obscurity (propriety non-public algorithms)* is not advocated by cryptographers.

Yao's Millionaires' Problem:
1. User B chose large random number r, send C = EpubA(r) – b to user A
2. User A decrypt Y[k] = DprivA(C+k). For k = b, it will decrypt to the exact same r again!
3. User A generate Z[k] = Y[k] mod p for k <= a, Z[k] = Y[k] mod p + 1 for k > a.
4. User B check Z[b] is r or not. If it is true, then A is richer or equal to Bob's wealth. If it is false, A is poorer than User B.

Lagrange Interpolation: Use T+1 points to reconstruct a unique Lagrange Interpolation Polynomial of degree T using Recombination Vector. => This recombination vector with dimension 1 x T + 1 can be quickly used to compute the S(0) secret value via summation.

BGW Protocol:

1. Any computable function with a fixed-length input can specified as a polynomial-sized Boolean circuit using AND and NOT gates.
2. **Perfect Privacy:** Polynomial of degree T with secret S cannot be constructed with T or less shares, thus achieved information-theoretically secure. For example, for 12 parties with a random polynomial of degree 6, it needs at least 7 colluded parties to recover the secret. This is called **(7, 12) threshold sharing scheme.**
3. Needed **Prime > # Parties, 2T < # Parties.**
4. The reason why A+B is hidden is because **homomorphic property**: Homomorphic encryption is a form of encryption allowing one to perform calculations on encrypted data without decrypting it first. The result of the computation is in an encrypted form, when decrypted the output is the same as if the operations had been performed on the unencrypted data.
5. **For Multiplication Gate: N^2 messages are sent for each MUL gate.**

Garbled Circuit:

1. Two parties. Alice is circuit garbler, Bob is circuit evaluator
   A. Compiles function f(x,y) into a composition of logic gates like AND, OR, XOR using Karnaugh Maps
   B. Alice sends the following things to Bob
      i. Key corresponding to Alice's input. Assume the input for A is 0, so the key that Alice sends for Wire A is k[a, 0]
      ii. Permutated garbled tables (e.g., with AND)
      iii. Decryption mapping table (i.e., with 2 possible keys for output wire c to their plaintext bits)