CLAIMS:

1. A computer-implemented method comprising:

a) receiving from users, respective primary entity-IDs, each primary entity-ID comprising a first and second bit string;

b) generating respective mapped-IDs from each of the primary entity-IDs, such that

i) each of the mapped-IDs is unique, and

ii) the mapped-IDs each comprise the first bit string of the associated primary entity-ID and a third bit string different from the second bit string of the associated primary entity-ID; and

c) generating, for each primary entity-ID, using a hash function, a shard key from the primary entity-ID or the associated mapped-ID,

wherein the hash function generates the shard key based on a value of the first bit string, and the hash function generates the same shard key regardless of whether the primary entity-ID or the associated mapped-ID is received as input to the hash function.

2. The method according to claim 1, wherein the first bit strings have a bit length N, and the primary entity-IDs and the mapped-IDs comprise the first bit strings as the N least significant bits.

3. The method according to claim 1 or 2, wherein the bit length N of each first bit string is 10, and each mapped-ID has a bit length of 64.

4. The method according to any preceding claim and comprising maintaining a counter, wherein the third bit strings of the mapped-IDs comprise counter information obtained from the counter.

5. The method according to any preceding claim, further comprising, for each primary entity-ID, a step of storing the primary entity-ID, the associated mapped-ID, and the shard key in the same row of a database.

6. The method according to claim 5, wherein the database is configured as a partitioned database comprising a plurality of shards, each shard being associated with a different shard key value, and wherein storing the primary entity-ID, the mapped-ID, and the shard key in the same row of a database comprises storing the primary entity-ID, the mapped-ID, and the shard key in the same row of a shard associated with the value of the shard key.

7. A computer-implemented method of resolving conflicts between a first and a second database, the first and second databases having respective pluralities of rows, each row storing an entity-ID and a mapped-ID associated with the entity-ID, the method comprising:

(a) generating a new mapped-ID from a new entity-ID according to the method of any of claims 1-6;

(b) storing the new entity-ID and the new mapped-ID associated with the entity-ID in a new row of the first database;

(c) determining a conflict between the new entity-ID and new mapped-ID and an existing entity-ID and associated existing mapped-ID stored in the second database;

(d) determining a valid mapped-ID associated with the new entity-ID, and

(e) conditional on determining that the valid mapped-ID and the existing mapped-ID stored in the second database are different, updating the second database based on the new entity-ID and the associated valid mapped-ID.

8. The method according to claim 7, wherein step (c) comprises determining that the new entity-ID and the existing entity-ID are the same, and that the new mapped-ID and the existing mapped-ID are different.

9. The method according to claim 7 or 8, wherein a third database has a further plurality of rows, each row storing an entity-ID and a mapped-ID associated with the entity-ID, and step (e) comprises:

retrieving, from the third database, a further mapped-ID associated with the new mapped-ID, and

selecting as the valid mapped-ID associated with the new entity-ID, one of the new mapped-ID, the existing mapped-ID stored in the second database and the further mapped-ID retrieved from the third database.

10. The method according to claim 9, wherein the step of selecting the valid mapped-ID comprises:

determining a counter value for each of the new mapped-ID, the existing mapped-ID stored in the second database and the further mapped-ID retrieved from the third database, and selecting, as the valid mapped-ID, the mapped-ID with the lowest counter value.

11. The method according to claim 9, wherein the first, second and third databases further store in each row a creation timestamp specifying a time when the mapped entity-ID stored in said row was created, and the step of selecting the valid mapped-ID comprises:

selecting from the new mapped-ID, the existing mapped-ID stored in the second database and the further mapped-ID retrieved from the third database the one with the earliest timestamp.

12. The method according to any one of claims 7 to 11 further comprising:

(f) conditional on determining that the valid mapped-ID and the new mapped-ID are different, updating the first database based on the new entity-ID and the associated valid mapped-ID; and

(g) conditional on determining that the valid mapped-ID and the further mapped-ID retrieved from the third database are different, updating the third database based on the new entity-ID and the associated valid mapped-ID.

13. A computer-implemented method of processing, by a database server, a request from a client device for providing a mapped-ID associated with a primary entity-ID, the database server storing a primary copy of a database having a plurality of rows, each row comprising one of a plurality of entity-IDs and a mapped-ID associated with the entity-ID, and the database server further storing first and second cached copies of the database, the method comprising:

(a) receiving the request from the client device, the request comprising the primary entity-ID and service information specifying whether the request is associated with a first or second service;

(b) conditional on determining that the service information specifies that the request is associated with the first service:

(b1) determining whether the mapped-ID associated with the primary entity-ID is stored in the first cached copy;

(b2) conditional on determining that the mapped-ID is not stored in the first cached copy, determining whether the mapped-ID associated with the primary entity-ID is stored in the primary copy of the database;

(b3) conditional on determining that the mapped-ID is stored in one of the first cached copy and the primary copy of the database, obtaining the mapped-ID and providing the mapped-ID to the client device;

(c) conditional on determining that the service information specifies that the request is associated with the second service:

(c1) determining whether the mapped-ID associated with the primary entity-ID is stored in the second cached copy;

(c2) conditional on determining that the mapped-ID is not stored in the second cached copy, determining whether the mapped-ID associated with the primary entity-ID is stored in the primary copy of the database;

(c3) conditional on determining that the mapped-ID is stored in one of the second cached copy and primary copy of the database, obtaining the mapped-ID and transmitting the mapped-ID to the client device; and

(d) conditional on determining that the mapped-ID is not stored in the primary copy of the database:

(d1) generating the mapped-ID associated with the primary entity-ID;

(d2) transmitting the mapped-ID to the client device, and

(d3) storing the primary entity-ID and the associated mapped-ID in a new row of the database,

wherein the mapped-IDs have been generated from the primary entity-IDs according to the method of any one of claims 1-6.

14. The method according to claim 13, wherein the first and second cached copies fully cache the database, and the database server further stores a first partially cached copy of the first cached copy and a second partially cached copy of the second cached copy;

wherein step (b1) of the method comprises:

determining whether the mapped-ID associated with the primary entity-ID is stored in the first partially cached copy;

conditional on determining that the mapped-ID is not stored in the first partially cached copy, determining whether the mapped-ID associated with the primary entity-ID is stored in the first cached copy; and

wherein the step (c1) of the method comprises:

determining whether the mapped-ID associated with the primary entity-ID is stored in the second partially cached copy, and

conditional on determining that the mapped-ID is not stored in the second partially cached copy, determining whether the mapped-ID associated with the primary entity-ID is stored in the second cached copy.

15. A computer-implemented method, the method comprising:

(a) receiving, from a client device, a first request for providing an application with a mapped-ID associated with a primary entity-ID, a mapped-ID having been generated from said primary entity-ID according to the method of any one of claims 1-6, the request comprising the primary entity-ID;

(b) retrieving the mapped-ID associated with the primary entity-ID from a database;

(c) providing, to the application, the mapped-ID;

(d) processing the primary entity-ID and the mapped-ID with an encryption key to generate an encrypted primary entity-ID and an encrypted mapped-ID; and

(e) transmitting, to the client device, the encrypted entity-ID, the encrypted mapped-ID and expiration information specifying a lifetime of the encrypted entity-ID and the encrypted mapped-ID.

16. The method according to claim 15, further comprising:

(f) receiving, from the client device, a second request for providing the application with the mapped-ID associated with the primary entity-ID, the request comprising the primary entity-ID, the encrypted entity-ID, the encrypted mapped-ID and expiration information;

(g) determining, based on the expiration information, whether the encrypted primary entity-ID and the encrypted mapped-ID are valid, and

(h) conditional on determining that the encrypted primary entity-ID and the encrypted mapped-ID are valid, decrypting, using the encryption key, the received encrypted primary entity-ID and the encrypted mapped-ID, and providing the decrypted mapped-ID to the application.

17. A system comprising one or more computers and one or more storage devices storing instructions that when executed by the one or more computers cause the one or more computers to perform the method of any one of claims 1-16.

18. One or more non-transitory computer storage media storing instructions that when executed by one or more computers cause the one more computers to perform the operations of any one of claims 1-16.

19. A system comprising first, and second database servers, the first and second database servers storing respectively first and second databases, the databases each having a plurality of rows, each row storing an entity-ID and a mapped-ID associated with the entity-ID;

the first database server configured to:

store a new entity-ID and a new mapped-ID associated with the entity-ID in a new row of the first database, and

transmit the new entity-ID and the new mapped-ID to the second database server; and

the second database server configured to:

determine a conflict between the received new entity-ID and new mapped-ID and an existing entity-ID and associated existing mapped-ID stored in the second database;

determine a valid mapped-ID associated with the new entity-ID, and

conditional on determining that the valid mapped-ID and the existing mapped-ID stored in the second database are different, update the second database based on the new entity-ID and the associated valid mapped-ID,

wherein the first database server is further configured to generate the new mapped-ID from the new entity-ID according to the method of any one of claims 1-6.

20. The system according to claim 19, wherein the second database server is configured to determine a conflict between the received the new entity-ID and new mapped-ID and an existing entity-ID and associated existing mapped-ID stored in the second database by determining that the received new entity-ID and existing entity-ID are the same, and that the received new mapped-ID and the existing mapped-ID are different

21. The system according to claim 19 or 20 and further comprising a third database server storing a third database having a plurality of rows, each row storing an entity-ID and a mapped-ID associated with the entity-ID, and

wherein the first database server is further configured transmit the new entity-ID and the new mapped-ID to the third database server, and

the second database server being configured to determine a valid mapped-ID associated with the new entity-ID by:

retrieving from the third database server a further mapped-ID which is associated with the new mapped-ID and stored in the third database, and

selecting as the valid mapped-ID associated with the new entity-ID, one of the new mapped-ID received from the first database server, the existing mapped-ID stored in the second database, and the further mapped-ID retrieved from the third database server.

22. The system according to claim 21, wherein the second database server is configured to select the valid mapped-ID by:

determining a counter value for each of the new mapped-ID received from the first database server, the existing mapped-ID stored in the second database, and the further mapped-ID retrieved from the third database server, and select, as the valid mapped-ID, the mapped-ID with the lowest counter value.

23. The system according to claim 21, wherein the first, second and third databases are further configured to store in each row a creation timestamp specifying the time when the mapped entity-ID stored in said row was created;

the first, second, and third database servers further configured to transmit corresponding creation timestamps to the second database server when transmitting the new entity-ID and the new mapped-ID and the further mapped-ID respectively; and wherein the second database server being configured to select the valid mapped-ID by:

selecting from the new mapped-ID received from the first database server, the existing mapped-ID stored in the second database, and the further mapped-ID retrieved from the third database server, and the mapped-ID with the lowest earliest timestamp.

24. The system according to any one of claims 19 to 23, wherein the second database server is further configured to transmit the new entity-ID and the valid mapped-ID to the first and third database servers;

the first database server is further configured to, conditional on determining that the valid mapped-ID and the new mapped-ID are different, update the first database based on the new entity-ID and the associated valid mapped-ID, and

the third database server is further configured to, conditional on determining that the valid mapped-ID and the further mapped-ID are different, update the third database based on the new entity-ID and the associated valid mapped-ID.

25. A database server for processing a request from a client device for providing a mapped-ID associated with a primary entity-ID, the database server configured to store:

a primary copy of a database having a plurality of rows, each row comprising one of a plurality of entity-IDs and a mapped-ID associated with the entity-ID, and

first and second cached copies of the database;

the database server further configured to:

receive the request from the client device, the request comprising the primary entity-ID and service information specifying whether the request is associated with a first or second service;

conditional on determining that the service information specifies that the request is associated with the first service:

determine whether the mapped-ID associated with the primary entity-ID is stored in the first cached copy;

conditional on determining that the mapped-ID is not stored in the first cached copy, determine whether the mapped-ID associated with the primary entity-ID is stored in the primary copy of the database;

conditional on determining that the mapped-ID is stored in one of the first cached copy and the primary copy of the database, obtain the mapped-ID and providing the mapped-ID to the client device;

conditional on determining that the service information specifies that the request is associated with the second service:

determine whether the mapped-ID associated with the primary entity-ID is stored in the second cached copy;

conditional on determining that the mapped-ID is not stored in the second cached copy, determine whether the mapped-ID associated with the primary entity-ID is stored in the primary copy of the database;

conditional on determining that the mapped-ID is stored in one of the second cached copy and primary copy of the database, obtain the mapped-ID and transmitting the mapped-ID to the client device; and

conditional on determining that the mapped-ID is not stored in the primary copy of the database:

generate the mapped-ID associated with the primary entity-ID;

transmit the mapped-ID to the client device, and

store the primary entity-ID and the associated mapped-ID in a new row of the database,

wherein the mapped-IDs have been generated from the entity-IDs according to the method of any one of claims 1-6.

26. The database server according to claim 25, wherein the first and second cached copies fully cache the database, and the database server further stores a first partially cached copy of the first cached copy and a second partially cached copy of the second cached copy;

wherein the database server is configured to determine whether the mapped-ID associated with the primary entity-ID is stored in the first cached copy by:

determining whether the mapped-ID associated with the primary entity-ID is stored in the first partially cached copy;

conditional on determining that the mapped-ID is not stored in the first partially cached copy, determining whether the mapped-ID associated with the primary entity-ID is stored in the first cached copy; and

wherein the database server is configured to determine whether the mapped-ID associated with the primary entity-ID is stored in the second cached copy by:

determining whether the mapped-ID associated with the primary entity-ID is stored in the second partially cached copy, and

conditional on determining that the mapped-ID is not stored in the second partially cached copy, determining whether the mapped-ID associated with the primary entity-ID is stored in the second cached copy.