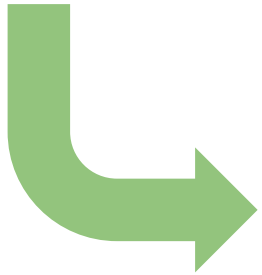
Joining Tables in SQL

Data across multiple tables

Very often data will be spread across **multiple tables**. In order to perform the analyses you need to it will be important to **combine the data**. There are many different ways to combine the data and each serves a slightly different purpose.

Table A

id	col_1	col_2
1	23-B	12
2	435	45
3	AB145	23
4	BB	56
5	435	123



merge_table

id	col_1	col_2	col_a
1	23-B	12	a
2	435	45	a
3	AB145	23	b
4	BB	56	b
5	435	123	a

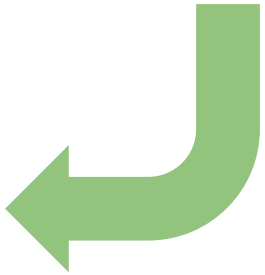


Table B

id	lookup_id	col_a
a1	1	a
a2	2	a
b1	3	b
b2	4	b
a3	5	a

Keywords

- JOIN
  - INNER
  - LEFT
  - RIGHT
  - FULL
  - CROSS
  - self
  - semi
  - anti
- USING

Keys and Referential Integrity

Relationships between tables are encoded through the use of **keys**:

- **Primary key:** One or more columns which uniquely identifies each row. A table will have only one primary key.
- **Foreign key:** One or more columns in another table which refer to the primary key in another table. A table can contain more than one foreign key.

Referential integrity means that when a foreign key value is used, it must reference a valid, existing primary key in the parent table. A breakdown in referential integrity can have undesirable side effects:

- Incomplete data being returned, usually with no indication of an error/"lost" records
- Strange results appearing in reports (such as products without an associated company).

Entity Relationship Diagram (ERD)

Displays the tables and how those tables relate/are connected together.

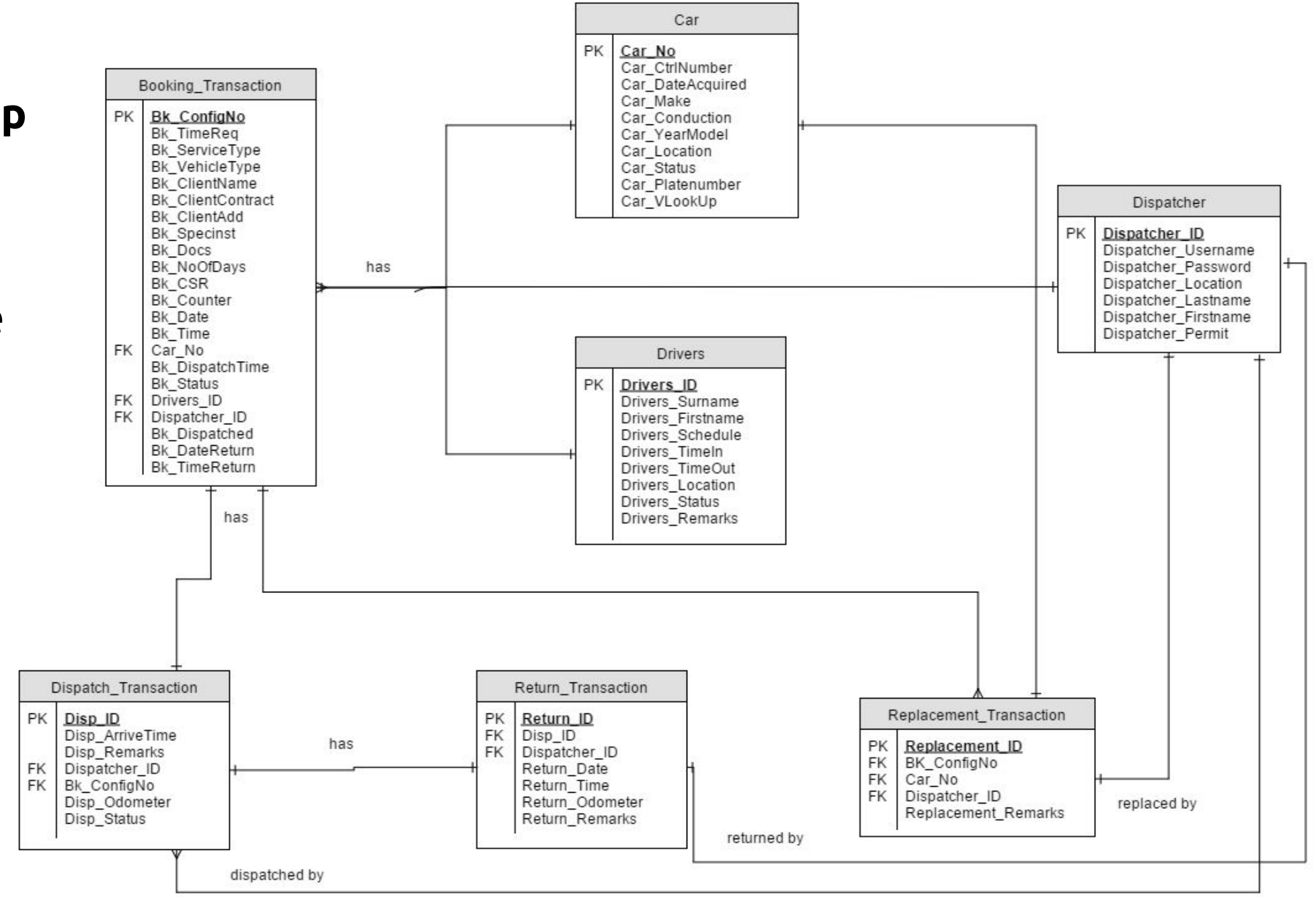


Image: https://sites.google.com/site/dracvbms/entity-relationship-diagram

Normalization

Goals: reduce data redundancy and improve data integrity

If a customer address changes, you should only have to update it in one table.

Allows for extending the database structure with minimal impacts to the existing structure.

The properties are encoded in the 6 database “normal forms”.

There are tradeoffs - more normalization leads to less redundancy, but more tables and more complicated queries.

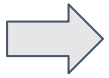
To learn more about datbase normalization and the normal forms, see https://en.wikipedia.org/wiki/Database_normalization

Normalization

Common practice is to use **Third Normal Form(3NF).** The State_Crimes table on the left below can be normalized by creating a state table and a crime type table. Tables like State and Crime are sometimes called lookup tables.

State_Crimes

State	Crime_type	Value	Urban_pop
Alabama	Murder	13.2	58
Alabama	Assault	236	58
Alabama	Rape	21	58
Alaska	Murder	10	48
Alaska	Assault	263	48
Alaska	Rape	44.5	48
Arizona	Murder	8.2	80
Arizona	Assault	294	80



State_Crimes

State_id	Crime_id	Value
1	1	13.2
1	2	236
1	3	21
2	1	10
2	2	263
2	3	44.5
3	1	8.2
3	2	294

State

id	Name	Urban_pop
1	Alabama	58
2	Alaska	48
3	Arizona	80

Crime

id	Name
1	Murder
2	Assault
3	Rape

https://en.wikipedia.org/wiki/Third_normal_form

ACID Properties

These define the key characteristics that SQL databases use to ensure database modifications are saved in a consistent, safe, and robust manner.

- **A**tomic:
  - A database transaction either succeeds or fails.
  - A transaction cannot be completed only partially.
- **C**onsistent:
  - Use of rules and constraints so state is always valid.
  - The data saved can’t violate any of the database’s integrity.
- **I**solation:
  - Transactions happen in isolation; No “mid-air collisions.”
- **D**urability:
  - Once committed a transaction is permanent, regardless of a subsequent system failure.