Column Name	Data Type	Constraints	Example	Purpose	Notes
id	TEXT (string)	PRIMARY KEY	em3G4M0fRpC6G00	Provide a unique identifier for users	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
username	TEXT (string)	NOT NULL, UNIQUE	john_doe	Give an easily readable and unique user chosen identifier for display	Should be between 3 and 16 characters (inclusive) in length. Can be changed at user discretion via a settings page. Should only include alphanumeric characters, dashes and underscores. The characters [?, &, =, /, #, and %], reserved for use in URLs, are not permitted as they cannot be encoded in all situations where they are required
hashed_password	TEXT (string)	NOT NULL	$argon2id$v=19$m=6553	Securely store a password which can be used to authenticate a user	Uses default argon2id hashing parameters. Hashed using the `password` object included in the Bun standard library.
session_id	TEXT (string)	NOT NULL, PRIMARY KEY	1414290f442b9e138c50	Provide a unique identifier for auth sessions	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
user_id	TEXT (string)	NOT NULL, REFERENCES users(id)	john_doe	Link the auth session to a specific user	Should be between 3 and 16 characters (inclusive) in length. Can be changed at user discretion via a settings page. Should only include alphanumeric characters, dashes and underscores. The characters [?, &, =, /, #, and %], reserved for use in URLs, are not permitted as they cannot be encoded in all situations where they are required
expires_at	INTEGER (date)	NOT NULL	1742226459	Store the time at which the session will no longer be valid, so that on a request we can check and delete the session	Stores the time in ms since unix epoch at the point of session creation + 30 days. Every time a user logs in between 0 and 14 days of expiration time, the session is extended by another 30 days. We expire the session after a period of prolonged inactivity because a user's intention to sign in somewhere they will not be frequently accessing the service is likely not permanent.
session_ip	TEXT (string)		203.0.113.25	Store assorted information related to the device the session belongs to. We later present this to the user, so they can evaluate whether a session needs to be revoked or not	Information should be presented via Cloudflare's request headers.
session_country	TEXT (string)		US		
session_city	TEXT (string)		New York		
session_region	TEXT (string)		NY		
device_type	TEXT (string)		Desktop		
user_agent	TEXT (string)		Mozilla/5.0 (Windows		
jti	TEXT (string)	NOT NULL, PRIMARY KEY	em3G5jAtHdEmo00	Provide a unique identifier for refresh tokens	we store a unique identifier for the JSON Web Token (JWT) created for desktop app authentication. This way, we can ensure that only one refresh token exists for each user, and accomodate for rotation of tokens should the user's refresh token become compromised. The refresh token is used to exchange for a short lived access token at runtime and encodes user data (username) to cut down on database calls from the desktop app. The JTI is generated in the same manner as a userId
user_id	TEXT (string)	NOT NULL, UNIQUE, REFERENCES users(id)	john_doe	Link the refresh token to a specific user	Should be between 3 and 16 characters (inclusive) in length. Can be changed at user discretion via a settings page. Should only include alphanumeric characters, dashes and underscores. The characters [?, &, =, /, #, and %], reserved for use in URLs, are not permitted as they cannot be encoded in all situations where they are required
telemetry_session_id	INTEGER (int)	NOT NULL, PRIMARY KEY	enitSU39sqGsH00	Provide a unique identifier for sessions	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
session_uid	INTEGER (int)	NOT NULL	0	Store the correct index of the player car in the data, in case data needs to be extracted later from a more complete dataset containing all cars	This should be 0 in all time trial cases (only supported mode at present). Stored mainly as a precaution and as prep for supporting additional modes
start_date	TEXT (date)	NOT NULL	2025-02-15T15:49:04.4	Logs the time at which the session started for sorting purposes	ISO-8601 formatted timestamp

end_date	TEXT (date)		2025-02-22T18:37:21.4	Logs the time at which the session ended, in order to calculate session duration	ISO-8601 formatted timestamp
total_distance	REAL (float)	NOT NULL	14000	Logs the distance travelled for display on the session page, can be used to evaluate against training goals	Stored in meters
weather	INTEGER (int)	NOT NULL	4	Logged so that we can display a weather icon on the session, and filter sessions by weather (rain times !== dry times)	// Weather - 0 = clear, 1 = light cloud, 2 = overcast, 3 = light rain, 4 = heavy rain, 5 = storm
time_of_day	INTEGER (int)	NOT NULL	0	Logged so that we can display an icon for time of day on the session	Minutes since midnight (track time)
total_laps	INTEGER (int)	NOT NULL	2	Store the amount of laps completed in the session, for display purposes	While this could be counted manually from individual lap entries, this would likely mean more database calls and the information is already exposed by the game's UDP interface
track_id	INTEGER (int)	NOT NULL, REFERENCES tracks(id)	3	Stores the id of the track that the session took place at. Links a track and its metadata to a session.	Integer ID exposed by the game's UDP interface. Stored database track information follows the same ID structure as the game, to conform to the IDs outlined by the UDP spec.
assists	INTEGER (int)	NOT NULL, REFERENCES assists(mask)	63195	Links the assists used to the corresponding session, via the session id	The assists are stored as a bitmask, meaning that we assign a set of bits in one larger integer to the value for a specific assist, and then to extract a specific assist value later we shift the integer along by the correct number of bits, and perform a bitwise AND operation for the length of bits we need, hence retrieving the value

lap_id	INTEGER (int)	NOT NULL, PRIMARY KEY*	1	Indicates the lap # (e.g. lap 1, 2, etc.) for sequencing purposes	*Component of a composite primary key, as no session will ever have multiple lap 1's, lap 2's etc.
session_uid	TEXT (string)	NOT NULL, PRIMARY KEY*	0	Ties the lap to a specific telemetry session	*Component of a composite primary key, as many laps could have the same # in their session's sequence, but belong to a different session.
lap_time_in_ms	INTEGER (int)	NOT NULL	68406	Stores the total time of the lap in ms for comparison, ranking and sorting purposes	
sector_1_time_in_ms	TEXT (date)	NOT NULL	17239	Stores the individual sector time for comparison and learning purposes	We should store the individual sector times as well for the purpose of more fine grained comparison between laps. This way, we can measure where the biggest potential time saves are and focus advice / training potentially on a sector by sector basis.
sector_2_time_in_ms	INTEGER (int)	NOT NULL	30283		
sector_3_time_in_ms	INTEGER (int)	NOT NULL	20883		
lap_valid_bit_flags	INTEGER (int)	NOT NULL	0	Stores the validity of the lap in a bitmask (?) for display and filtering purposes	We should exclude invalid (corner cutting) laps from any sort of ranked dataset, but they should still be saved and visible in the history of a session. They should additionally not affect ratings. The flags themselves are not well documented by the UDP spec and will have to be investigated manually

session_id	TEXT (string)	PRIMARY KEY, REFERENCES telemetry_sessions (uid)	enitSU39sqGsH00	Link the assist values to a specific session	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
mask	INTEGER (int)	NOT NULL	63195	Compactly stores all assists that the user used in a session	The assists are stored as a bitmask, meaning that we assign a set of bits in one larger integer to the value for a specific assist, and then to extract a specific assist value later we shift the integer along by the correct number of bits, and perform a bitwise AND operation for the length of bits we need, hence retrieving the value

track_id	INTEGER (int)	PRIMARY KEY	8	Uniquely identifies a specific location (track)	Matches what is shown on the UDP spec provided by Codemasters.

gp_name	TEXT (string)	UNIQUE, NOT NULL	Belgian Grand Prix	Stores the name used internally by the game for reference	Usually in the form `X Grand Prix` where X is a Country. With exceptions e.g. Melbourne and Australian GP existing. Matches what is shown on the UDP spec.
first_gp	TEXT (date)	NOT NULL	2004-04-04T14:30:00+0	Stores the date-time of the first held gp, to be shown on the track page	Information likely to be sourced from wikipedia or other internet sources. Real lap record could also be used for comparison to ingame laps (inaccurate/misleading?)
real_lap_record	INTEGER (int)	NOT NULL	91447	Stores the real life lap record in a f1 gp (in ms) for display on the track page	
country	TEXT (string)	UNIQUE, NOT NULL	BE	Stores the country of the track, so that a country flag can be displayed where appropriate	ISO-3166 2 digit country code representation
location	TEXT (string)	NOT NULL	Stavelot	Stores the city/area of a track, for display on the track page	Information sourced from wikipedia
track_name	TEXT (string)	UNIQUE, NOT NULL	Circuit de Spa-Franco	Stores the actual name of the race track, for display on the track page	Information sourced from wikipedia
track_length	INTEGER (int)	NOT NULL	7004	Stores the track length in meters for display on the track page	Information sourced from game's UDP interface (may not be accurate to real life)

Column Name	Data Type	Constraints	Example	Purpose	Notes
id	TEXT (string)	PRIMARY KEY	em3G4M0fRpC6G00	Provide a unique identifier for users	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
username	TEXT (string)	NOT NULL, UNIQUE	john_doe	Give an easily readable and unique user chosen identifier for display	Should be between 3 and 16 characters (inclusive) in length. Can be changed at user discretion via a settings page. Should only include alphanumeric characters, dashes and underscores. The characters [?, &, =, /, #, and %], reserved for use in URLs, are not permitted as they cannot be encoded in all situations where they are required
hashed_password	TEXT (string)	NOT NULL	$argon2id$v=19$m=6553	Securely store a password which can be used to authenticate a user	Uses default argon2id hashing parameters. Hashed using the `password` object included in the Bun standard library.

Column Name	Data Type	Constraints	Example	Purpose	Notes
id	TEXT (string)	NOT NULL, PRIMARY KEY	1414290f442b9e138c50c	Provide a unique identifier for auth sessions	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
user_id	TEXT (string)	NOT NULL, REFERENCES users(id)	john_doe	Link the auth session to a specific user	Should be between 3 and 16 characters (inclusive) in length. Can be changed at user discretion via a settings page. Should only include alphanumeric characters, dashes and underscores. The characters [?, &, =, /, #, and %], reserved for use in URLs, are not permitted as they cannot be encoded in all situations where they are required
expires_at	INTEGER (date)	NOT NULL	1742226459	Store the time at which the session will no longer be valid, so that on a request we can check and delete the session	Stores the time in ms since unix epoch at the point of session creation + 30 days. Every time a user logs in between 0 and 14 days of expiration time, the session is extended by another 30 days. We expire the session after a period of prolonged inactivity because a user's intention to sign in somewhere they will not be frequently accessing the service is likely not permanent.
session_ip	TEXT (string)		203.0.113.25	Store assorted information related to the device the session belongs to. We later present this to the user, so they can evaluate whether a session needs to be revoked or not	Information should be presented via Cloudflare's request headers.
session_country	TEXT (string)		US		
session_city	TEXT (string)		New York		
session_region	TEXT (string)		NY		
device_type	TEXT (string)		Desktop		
user_agent	TEXT (string)		Mozilla/5.0 (Windows		

Column Name	Data Type	Constraints	Example	Purpose	Notes
jti	TEXT (string)	NOT NULL, PRIMARY KEY	em3G5jAtHdEmo00	Provide a unique identifier for refresh tokens	we store a unique identifier for the JSON Web Token (JWT) created for desktop app authentication. This way, we can ensure that only one refresh token exists for each user, and accomodate for rotation of tokens should the user's refresh token become compromised. The refresh token is used to exchange for a short lived access token at runtime and encodes user data (username) to cut down on database calls from the desktop app. The JTI is generated in the same manner as a userId
user_id	TEXT (string)	NOT NULL, UNIQUE, REFERENCES users(id)	john_doe	Link the refresh token to a specific user	Should be between 3 and 16 characters (inclusive) in length. Can be changed at user discretion via a settings page. Should only include alphanumeric characters, dashes and underscores. The characters [?, &, =, /, #, and %], reserved for use in URLs, are not permitted as they cannot be encoded in all situations where they are required

Column Name	Data Type	Constraints	Example	Purpose	Notes
id	INTEGER (int)	NOT NULL, PRIMARY KEY	enitSU39sqGsH00	Provide a unique identifier for sessions	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
session_uid	INTEGER (int)	NOT NULL	0	Store the correct index of the player car in the data, in case data needs to be extracted later from a more complete dataset containing all cars	This should be 0 in all time trial cases (only supported mode at present). Stored mainly as a precaution and as prep for supporting additional modes
start_date	TEXT (date)	NOT NULL	2025-02-15T15:49:04.4	Logs the time at which the session started for sorting purposes	ISO-8601 formatted timestamp
end_date	TEXT (date)		2025-02-22T18:37:21.4	Logs the time at which the session ended, in order to calculate session duration	ISO-8601 formatted timestamp
total_distance	REAL (float)	NOT NULL	14000	Logs the distance travelled for display on the session page, can be used to evaluate against training goals	Stored in meters
weather	INTEGER (int)	NOT NULL	4	Logged so that we can display a weather icon on the session, and filter sessions by weather (rain times ≢ dry times)	// Weather - 0 = clear, 1 = light cloud, 2 = overcast, 3 = light rain, 4 = heavy rain, 5 = storm
time_of_day	INTEGER (int)	NOT NULL	0	Logged so that we can display an icon for time of day on the session	Minutes since midnight (track time)
total_laps	INTEGER (int)	NOT NULL	2	Store the amount of laps completed in the session, for display purposes	While this could be counted manually from individual lap entries, this would likely mean more database calls and the information is already exposed by the game's UDP interface
track_id	INTEGER (int)	NOT NULL, REFERENCES tracks(id)	3	Stores the id of the track that the session took place at. Links a track and its metadata to a session.	Integer ID exposed by the game's UDP interface. Stored database track information follows the same ID structure as the game, to conform to the IDs outlined by the UDP spec.
assists	INTEGER (int)	NOT NULL, REFERENCES assists(mask)	63195	Links the assists used to the corresponding session, via the session id	The assists are stored as a bitmask, meaning that we assign a set of bits in one larger integer to the value for a specific assist, and then to extract a specific assist value later we shift the integer along by the correct number of bits, and perform a bitwise AND operation for the length of bits we need, hence retrieving the value

Column Name	Data Type	Constraints	Example	Purpose	Notes
id	INTEGER (int)	NOT NULL, PRIMARY KEY*	1	Indicates the lap # (e.g. lap 1, 2, etc.) for sequencing purposes	*Component of a composite primary key, as no session will ever have multiple lap 1's, lap 2's etc.
session_uid	TEXT (string)	NOT NULL, PRIMARY KEY*	0	Ties the lap to a specific telemetry session	*Component of a composite primary key, as many laps could have the same # in their session's sequence, but belong to a different session.
lap_time_in_ms	INTEGER (int)	NOT NULL	68406	Stores the total time of the lap in ms for comparison, ranking and sorting purposes	
sector_1_time_in_ms	TEXT (date)	NOT NULL	17239	Stores the individual sector time for comparison and learning purposes	We should store the individual sector times as well for the purpose of more fine grained comparison between laps. This way, we can measure where the biggest potential time saves are and focus advice / training potentially on a sector by sector basis.
sector_2_time_in_ms	INTEGER (int)	NOT NULL	30283		
sector_3_time_in_ms	INTEGER (int)	NOT NULL	20883		
lap_valid_bit_flags	INTEGER (int)	NOT NULL	0	Stores the validity of the lap in a bitmask (?) for display and filtering purposes	We should exclude invalid (corner cutting) laps from any sort of ranked dataset, but they should still be saved and visible in the history of a session. They should additionally not affect ratings. The flags themselves are not well documented by the UDP spec and will have to be investigated manually

Column Name	Data Type	Constraints	Example	Purpose	Notes
session_id	TEXT (string)	PRIMARY KEY, REFERENCES telemetry_sessions (uid)	enitSU39sqGsH00	Link the assist values to a specific session	A 15 character, base-62 encoded identifier generated from a combination of the current timestamp (to make id's sequential) and 6 random bytes. The trailing end of the id is padded with 0's to fill out to 15 characters.
mask	INTEGER (int)	NOT NULL	63195	Compactly stores all assists that the user used in a session	The assists are stored as a bitmask, meaning that we assign a set of bits in one larger integer to the value for a specific assist, and then to extract a specific assist value later we shift the integer along by the correct number of bits, and perform a bitwise AND operation for the length of bits we need, hence retrieving the value

Column Name	Data Type	Constraints	Example	Purpose	Notes
id	INTEGER (int)	PRIMARY KEY	8	Uniquely identifies a specific location (track)	Matches what is shown on the UDP spec provided by Codemasters.
gp_name	TEXT (string)	UNIQUE, NOT NULL	Belgian Grand Prix	Stores the name used internally by the game for reference	Usually in the form `X Grand Prix` where X is a Country. With exceptions e.g. Melbourne and Australian GP existing. Matches what is shown on the UDP spec.
first_gp	TEXT (date)	NOT NULL	2004-04-04T14:30:00+0	Stores the date-time of the first held gp, to be shown on the track page	Information likely to be sourced from wikipedia or other internet sources. Real lap record could also be used for comparison to ingame laps (inaccurate/misleading?)
real_lap_record	INTEGER (int)	NOT NULL	91447	Stores the real life lap record in a f1 gp (in ms) for display on the track page	
country	TEXT (string)	UNIQUE, NOT NULL	BE	Stores the country of the track, so that a country flag can be displayed where appropriate	ISO-3166 2 digit country code representation
location	TEXT (string)	NOT NULL	Stavelot	Stores the city/area of a track, for display on the track page	Information sourced from wikipedia
track_name	TEXT (string)	UNIQUE, NOT NULL	Circuit de Spa-Franco	Stores the actual name of the race track, for display on the track page	Information sourced from wikipedia
track_length	INTEGER (int)	NOT NULL	7004	Stores the track length in meters for display on the track page	Information sourced from game's UDP interface (may not be accurate to real life)