1. **SQL**

**Логические операторы AND OR NOT**

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.**

**Операторы сравнения**

Операторы сравнения определены для всех данных, которые имеют смысл. Возвращают тип Boolean, при этом выражение 1<2<3 недопустимо т.к. оператор сравнения не умеет сравнивать с boolean

1. BETWEEN равнозначно ***a ∊ [x, y] a >= x AND a <= y***

BETWEEN SYMETRIC то же самое, но если аргумент слева от AND больше правого, то аргументы меняются местами: X BETWEEN 250 AND 1

1. Все операторы сравнения при сравнении с Null выдают Null. Но есть альтернатива:

IS DISTINCT FROM равнозначно <>. Один null = true. Оба Null = false

IS NOT DISTINCT FROM равнозначно =. Один null = false. Оба Null = true

1. ISNULL NOTNULL – очевидно. Однако проверка x = null вернет null

Если с NULL сравнить строку, то ISNULL = true, если все строки NULL, false, если НЕ все.

1. IS TRUE IS FALSE IS UNKNOWN и их обратные – проверка Boolean выражения
2. Поиск по шаблону LIKE, SIMILAR, posix

‘abc’ LIKE ‘\_b\_’ это true

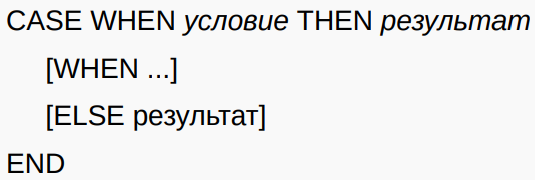
% - от 0 до много; \_ - ровно один; [] – символы на выбор; [^] – кроме этих; [-] – диапазон.

SIMILAR TO = LIKE (similar комбинирует LIKE с регулярными выражениями)

\*-0 или более раз +-1 или более раз ?-0 или 1 раз и т.д…..

POSIX

~ соответствие; !~ не соответствие; ~\* !~\* - без учета регистра



Если WHEN не выполняется, а ELSE отсутствует, возвращает NULL

1. CAST(x AS type)
2. X IN [1,2,3] – true если равен одному из чисел

**2. Соединение таблиц**

Типы соединений:

**CROSS JOIN** – Декартово произведение таблиц

**INNER JOIN** – сопоставляет строку t1 и t2 согласно JOIN ON

**LEFT OUTER JOIN** – выполняет INNER JOIN и добавляет строки левой таблицы, у которых нет соответствия

**RIGHT OUTER JOIN** – то же самое, но добавляет строки правой таблицы

**FULL OUTER JOIN** – добавляет строки обеих таблиц

**JOIN USING –** для таблиц с одинаковыми именами выводит совпадения

**NATURAL JOIN –** упрощенное USING: образует USING из всех столбцов. Если таких нет, то работает как CROSS JOIN

Теперь из интересного:

1. **JOIN t2 ON t1.num = t2.num AND t2.value = 'xxx';**

Условие WHERE неравносильно т.к. WHERE выполняется после соединения. Это имеет значение только при внешних соединениях

**JOIN t2 ON t1.num = t2.num WHERE t2.value = 'xxx';**

1. Можно задать **псевдоним** для таблицы FROM таблица AS псевдоним

Это нужно при соединении таблицы с собой, либо для читаемости.

Когда псевдоним задан, мы больше не можем обращаться по настоящему имени

Также можно задавать псевдонимы столбцов FROM employees AS e (id, name, salary)

1. **Подзапросы**

FROM () формирует внутри скобок таблицу или столбец

Подзапросы обычно обрабатываются “снизу-вверх”

**Коррелированные** подзапросы зависят от внешнего запроса. Строки внешней таблицы извлекаются по одной, и для каждой выполняется подзапрос. Если строка удовлетворяет условию подзапроса, то она включается в результат

Варианты подзапросов:

**Выражение IN (подзапрос):** возвращает false, если нет соответствия, либо подзапрос вернул пустую строку. Возвращает NULL, если выражение NULL, либо нет соответствия, а в подзапросе есть хотя бы один null. (пустая строка и null это разные вещи)

**EXISTS (подзапрос):** если подзапрос возвращает строку или null – true, не выводит строк – false.

**Выражение оператор ANY/SOME(подзапрос):** Полностью повторяет IN но сравнение происходит по оператору

**Выражение оператор ALL(подзапрос):** true – истина для всех строк, либо подзапрос пустой. False – хотя бы одна строка ложна. NULL – все строки ложны и хотя бы одна null.

1. **Представления**

Представление VIEW– именованный запрос. Не хранит результаты, а берет их при каждом вызове

Материализованное представление MATERIALIZED VIEW – Хранит результаты в БД

REFRESH MATERIALIZED VIEW имя – обновить данные

1. **Последовательности**

CREATE [TEMP] SEQUENCE имя [INCREMENT шаг] [MINVALUE мин\_знач] [START начало] (см. подробнее в документации)

создает таблицу, содержащую 1 строку.

После создания можно вызвать функции nextval, currval, setval

Последовательность может быть зацикленной или нет. Если цикла нет, то при достижении предельного значения, nextval будет давать ошибку