

Подготовка модели данных (модуль автокорреляции)

- Модуль подготовки модели данных с помощью ИИ (Фронт)
- Модуль подготовки модели данных с помощью ИИ (ML)

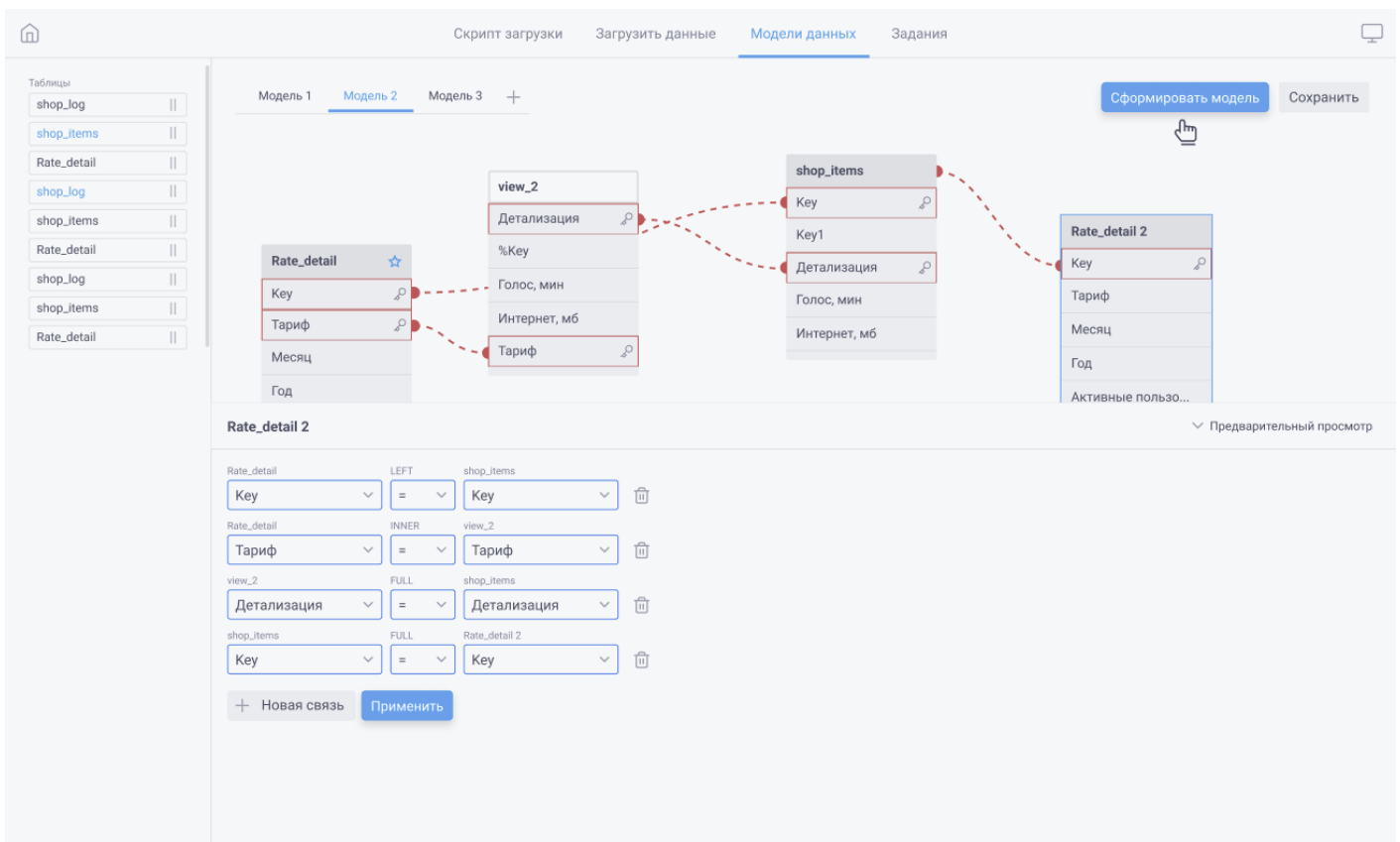
Модуль подготовки модели данных с помощью ИИ (Фронт)

Цель

Автоматическое формирование связей между загруженными таблицами в модели данных: определение ключевых полей и типов соединений между ними в зависимости от их данных.

Концепт для интерфейса

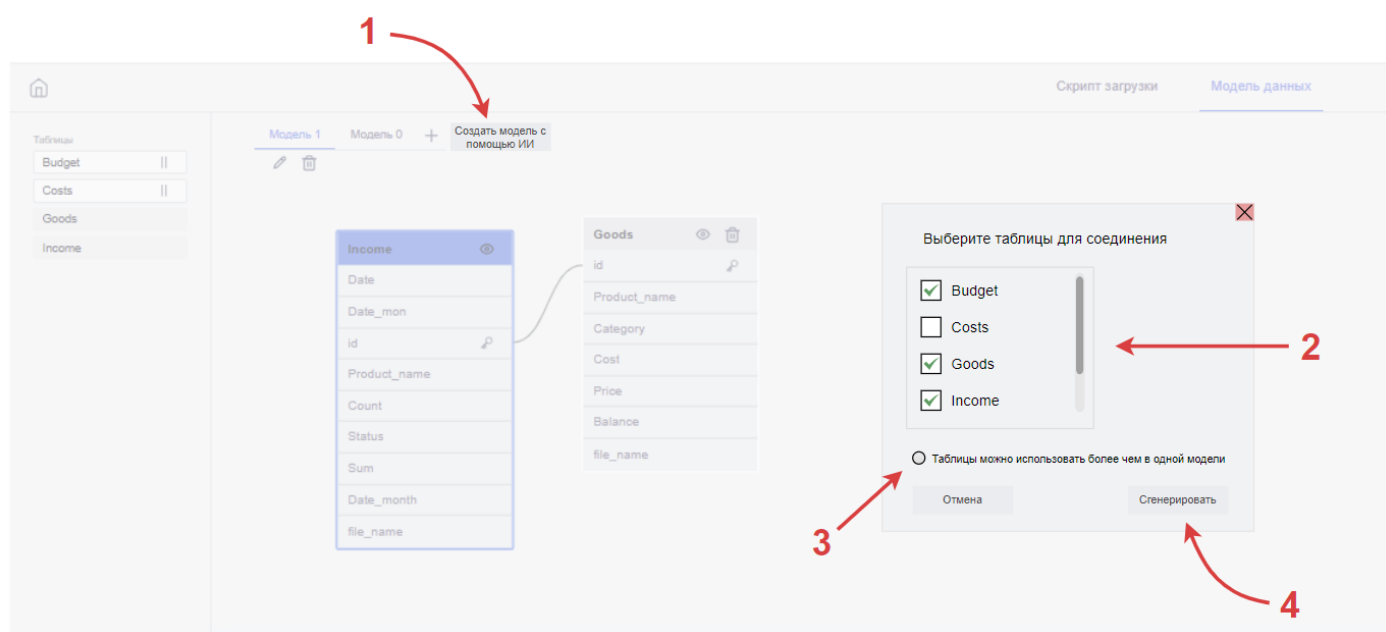
Рисунок первичного концепта, текущий см. в разделе **"Задание на разработку для фронта"**

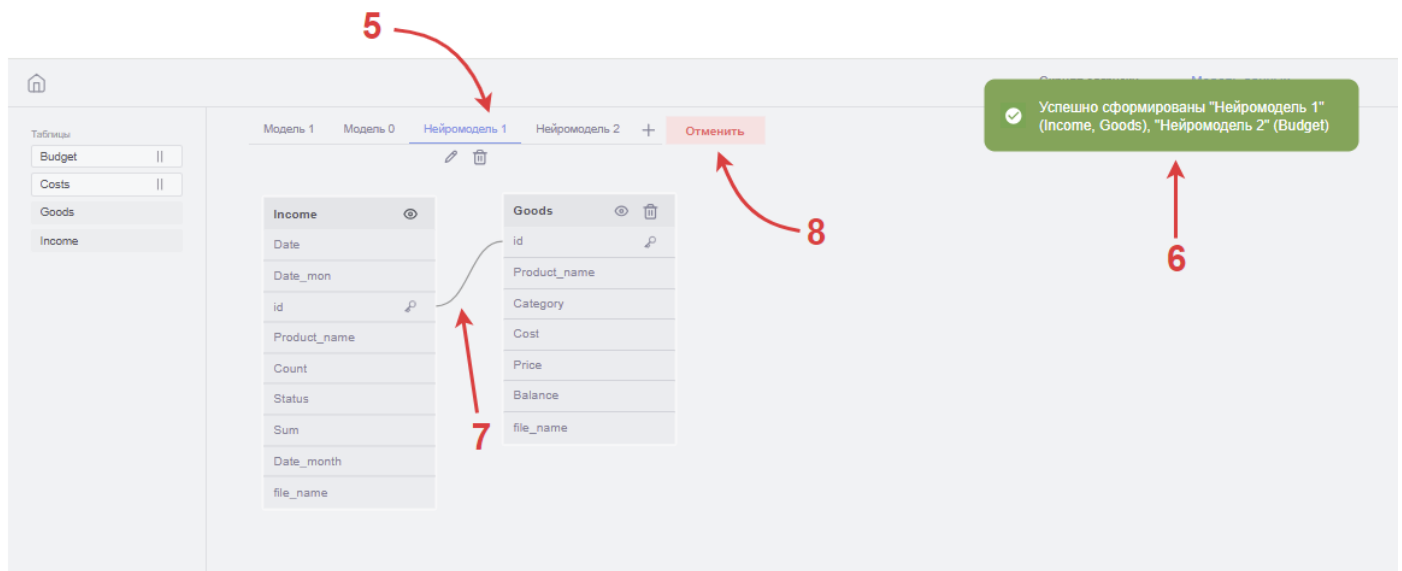


В диспетчере данных в разделе Модель данных необходимо добавить управляющую кнопку "Создать модель данных с помощью ИИ". При нажатии на кнопку должно открываться модальное окно со списком таблиц для соединения и кнопкой, разрешающей использовать таблицы более чем в одной модели. Выбор таблиц можно отменить или нажать на кнопку "Сгенерировать", после чего в модели данных появятся группы, созданные ИИ. Об успешной работе нейросети должно сообщаться системным сообщением, в котором необходимо перечислить созданные модели.

После создания таких моделей кнопка "Создать модель данных с помощью ИИ" трансформируется в кнопку "Отмена", которая позволяет одним нажатием вернуть модель данных в исходное состояние. Чтобы сохранить результат работы нейросети необходимо так же как и в случае с ручным проектированием нажать на кнопку "Сохранить" в правом верхнем углу окна. После нажатия кнопки "Отмена" или "Сохранить" возвращается возможность создать ещё одну модель с помощью ИИ.

Задание на разработку для фронта





Кнопка "**Создать модель с помощью ИИ**" **(1)** – расположена рядом с кнопкой добавления модели данных, перемещается вместе с ней при добавлении/удалении моделей. При нажатии на кнопку открывается модальное окно с выбором таблиц для соединения.

Окно выбора таблиц

В данном окне должен быть список с названиями всех таблиц **(2)**, полученных из скрипта загрузки. Рядом с каждым названием необходимо установить флажок для возможности множественного выбора. Если названия таблиц не помещаются в видимое поле списка, то по правой его границе должен размещаться ползунок для скrolла по списку.

Под списком должна быть кнопка-переключатель "Таблицы можно использовать более чем в одной модели" **(3)**. Если кнопка неактивна, то каждая таблица сможет появиться только в одной модели – в наиболее подходящей по логике применения. Если кнопка активна, то число повторов таблицы в разных моделях неограниченно, но в рамках одной модели таблица может появиться лишь однократно (нейросеть не создаёт копии таблиц). *По умолчанию* кнопка неактивна.

При нажатии на кнопку "Отмена" внизу окна выбора таблиц окно должно закрываться, все введённые изменения отменяются, подтверждать отмену не требуется. При нажатии на кнопку "Сгенерировать" **(4)** на бэкэнд отправляются следующие данные из скрипта загрузки:

- Названия таблиц вида Table
- Названия столбцов с принадлежностью таблице вида Table.Column
- Типы данных для каждого из столбцов вида Int32, String, Decimal (2)
- Сами значения из столбца, если объём невелик, или подвыборка из столбца (н-р, как результат сэмплирования), если данных достаточно много. Если строк в таблице больше 1 000 000, то необходимо ограничить выборку до этого значения с помощью Limit или Sample (второй вариант предпочтительнее ввиду необходимости передавать статистические данные, которые могут быть неравномерно распределены по таблице).

- Информация о полях в сортировке и первичных ключах вида Order By (список полей со всех таблиц) и Primary Key (список полей со всех таблиц)
- Разрешено ли использовать одну таблицу более чем в одной модели (булево) – результат нажатия на кнопку (3)

После отправки данных окно выбора таблиц закрывается, а в модели данных происходит ряд изменений.

Нейромодели

Нейросеть возвращает набор связанных таблиц, разбитых по группам. Каждая такая группа станет отдельной нейромоделью **(5)**, которая добавится в список существующих моделей.

Текущее рабочее название – "Нейромодель n", где n – порядковый номер модели, созданной нейросетью, без учёта других моделей (созданных вручную).

(Дополнительно) Возможное рабочее название – должно предлагаться нейросетью в зависимости от контекста использования данной группы таблиц.

Результат успешного создания нейромоделей должен сопровождаться системным сообщением **(6)** – "Успешно сформированы {Список моделей} ({Список таблиц в каждой модели})"

Внутри каждой группы для каждой пары таблиц нейросеть передаёт ключевые поля, по которым строится связь, и тип JOIN между ними. Эта информация должна использоваться для формирования связей между таблицами на фронте и отрисовки соединений по ключевым полям **(7)**. Поскольку модели формируются автоматически, необходимо определять положение таблиц в модели данных. Таблицы должны добавляться в том порядке, в котором они возвращаются из бэкенда. Первая таблица возникает в том месте, где она возникла бы, если бы была добавлена вручную пользователем нажатием на её название в списке таблиц слева от модели данных. Последующие таблицы должны возникать правее предыдущих на расстоянии $1,5 \cdot x \cdot (n-1)$ от левой границы первой таблицы, где x – ширина таблицы,

n – порядковый номер таблицы.

Созданные нейросетью модели ведут себя аналогично обычным: в них можно добавлять и удалять таблицы и связи. Сохраняется результат стандартной кнопкой "Сохранить" в правом верхнем углу. Чтобы удалить все созданные нейросетью модели можно воспользоваться кнопкой "Отменить" – при нажатии на неё должны удаляться все нейромодели, даже если они были любым образом изменены и переименованы. Не требует подтверждения.

Работа на кластере (на будущее)

Поскольку в КХ соединение JOIN для больших объёмов данных работает плохо, необходимо минимизировать их применение, соединив наибольшее возможное число таблиц с фактами в одну (например, с помощью UNION ALL).

Порядок действий:

- Нейросеть должна определить, какие из имеющихся таблиц можно отнести к справочникам (к которым будут подтягиваться данные из одной общей таблицы). Справочник можно идентифицировать по наличию пар "ключ-значение", согласно которым возможно формирование связей с таблицей фактов. Остальные таблицы относятся к фактическим.
- Фактические таблицы объединяются скриптом в одну таблицу:
`SELECT Столбцы FROM Таблица1 UNION ALL Столбцы FROM Таблица2 UNION ALL Столбцы FROM Таблица3...`
Скрипт не запускается, а используется для создания таблицы предварительного просмотра, демонстрируемой в отдельном модальном окне.
- Если пользователя устраивает структура, он нажимает кнопку "Продолжить". Иначе – "перегенерировать" (модель запускает ещё одну попытку классификации таблиц на справочники и факты).
- После подтверждения срабатывает сгенерированный скрипт как в обычной ситуации.

Модуль подготовки модели данных с помощью ИИ (ML)

Цель

Автоматическое формирование связей между загруженными таблицами в модели данных: определение ключевых полей и типов соединений между ними в зависимости от их данных.

Концепт для ML

Модель получает из интерфейса набор данных и метаданных по столбцам таблиц в итоговой модели: названия столбцов, их типы данных, количество уникальных значений, поля в сортировке. Используя эту информацию, а также некоторые вычисляемые характеристики, необходимо определить поля для создания связей между предложенными таблицами, тип связи (вид Join) и условия соединения (On). Для этого можно выполнить:

- Анализ уникальности полей
- Частотный анализ
- Оценку названий полей
- Определение полей в сортировке
- Поиск совпадающих значений
- Методы машинного обучения

Кроме того, при формировании связей существует ряд ограничений. Так, не должно быть создано модели, в которой существует хотя бы одно "кольцо" – замкнутая связь между 3 и более таблицами.

В качестве выхода модели необходимо предоставить пары полей в таблицах, тип Join (Inner, Left, Right или Full) и условие соединения On (=, >, < и т.д.).

Задание на разработку ML

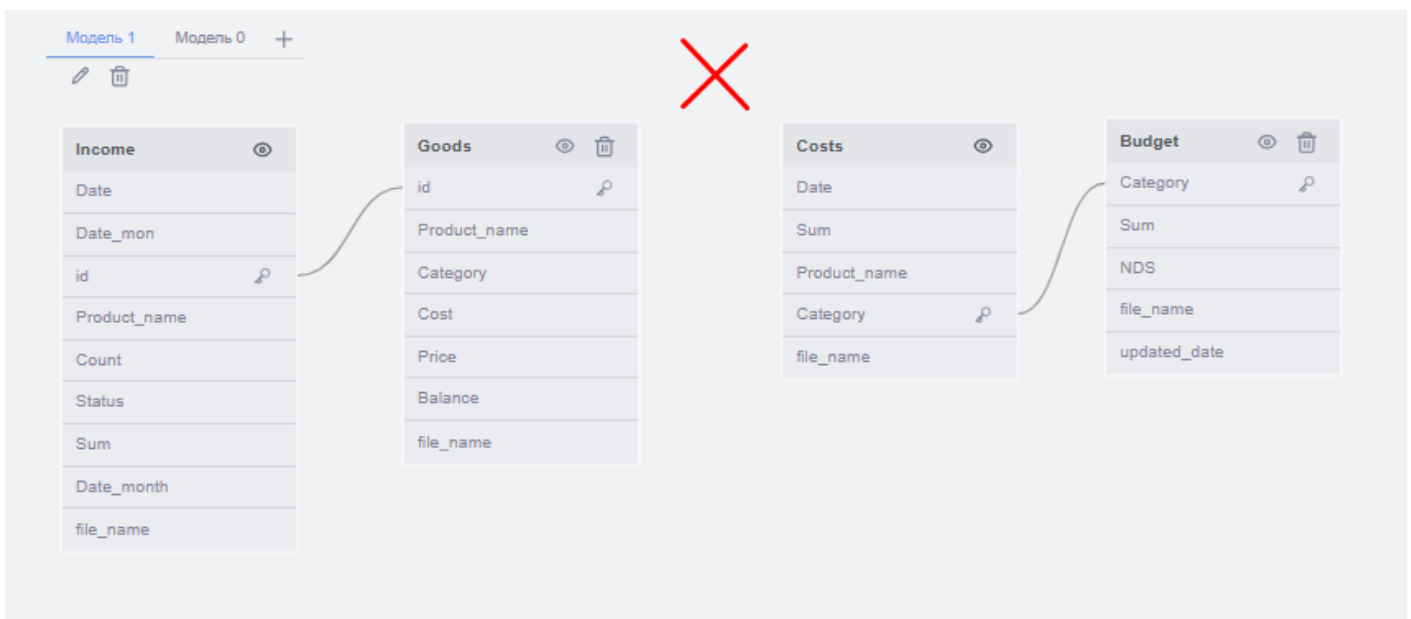
На вход модели ИИ попадают следующие данные из итоговой модели данных проекта:

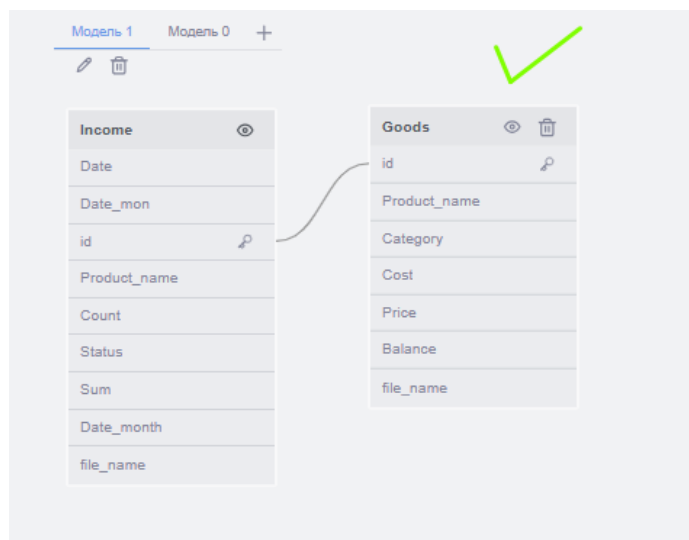
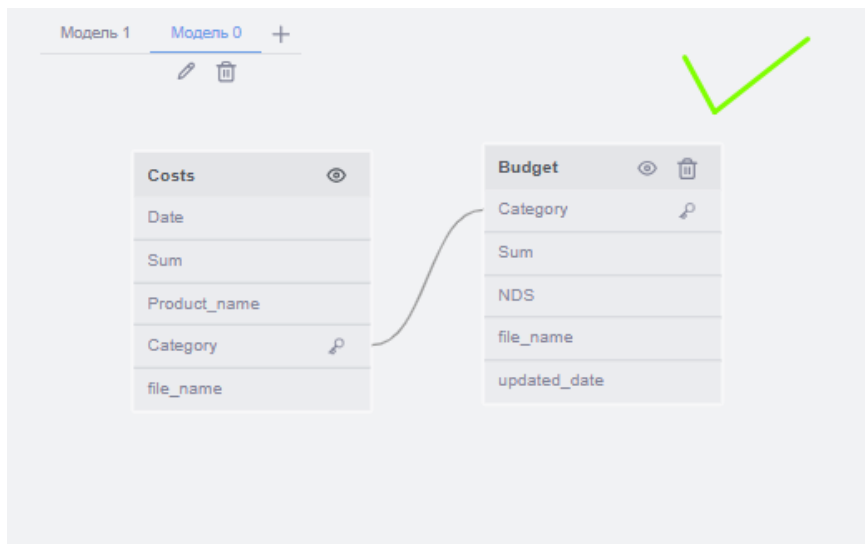
- Названия таблиц вида Table
- Названия столбцов с принадлежностью таблице вида Table.Column
- Типы данных для каждого из столбцов вида Int32, String, Decimal (2)
- Сами значения из столбца, если объём невелик, или подвыборка из столбца (н-р, как результат сэмплирования), если данных достаточно много
- Информация о полях в сортировке и первичных ключах вида Order By (список полей со всех таблиц) и Primary Key (список полей со всех таблиц)
- Разрешено ли использовать одну таблицу более чем в одной модели (булево)

Для получения наиболее корректного результата рекомендуется разбить работу модели на 4 блока: определение групп таблиц для связей, определение типа Join и условия соединения в зависимости от контекста каждой группы, классификацию столбцов внутри групп для выявления ключей соединения, формирование связей между таблицами (выход модели).

Формирование групп для соединения

Для FastBoard принципиально важно, чтобы все имеющиеся в модели данных таблицы были соединены. **Если между таблицами нет связи, то они должны быть разделены на разные группы.**





Набор входных таблиц со столбцами необходимо разбить по группам таким образом, чтобы выполнялись следующие правила:

- Ни одна таблица не должна встречаться более чем в одной группе, **если не установлена соответствующая настройка**
- Могут быть группы из одной таблицы, если это имеет логическое обоснование, причём их число должно быть минимизировано
- Должна быть как минимум одна группа из более чем одной таблицы
- Не должно быть создано группы, в которой существует хотя бы одно "кольцо" – замкнутая связь между 3 и более таблицами

Формирование групп происходит в зависимости от контекста использования таблиц. Модель должна определять этот контекст, опираясь на названия таблиц и другие входные данные (пример: связь между Income и Goods для определения результатов продаж по категориям товаров).

На этапе группировки НЕ определяются ключевые поля и НЕ создаются связи между ними.


Определение типа и условия соединения

Исходя из выбранного контекста модель должна определять тип связи между таблицами внутри группы:

- Если необходимо сохранить все строки из обеих таблиц в итоговом результате, то FULL JOIN
- Если необходимо сохранить все строки из одной таблицы в итоговом результате, то LEFT/RIGHT JOIN
- Если нет необходимости сохранить все строки, а в итоговом результате должны быть только значения из пересекающихся строк, то INNER JOIN

Необходимость сохранения строк также должна определяться контекстом соединения таблиц. Например, при соединении таблиц Income и Goods нет смысла сохранять все строки какой-то из таблиц – пользователя не интересуют продажи товаров, не описанных в таблице Goods, или товары из этой таблицы, которые не продавались и не отражены в таблице Income. А вот при соединении таблицы с данными о сотрудниках и таблицы с данными о их трудозатратах стоит выполнить Left Join по таблице сотрудников, поскольку в итоговом результате особенно важно видеть тех сотрудников, у которых отсутствует информация об их трудозатратах.

Для соединения различных таблиц в условии ON возможно использовать ТОЛЬКО оператор равенства.




[Products](#)
[Use cases](#)
[Docs](#)
[Resources](#)
[Pricing](#)
[Contact us](#)

[Getting Started](#)
[Cloud](#)
[Managing Data](#)
[Server Admin](#)
[SQL Reference](#)
[Integrations](#)
[chDB](#)
[About](#)
[Knowledge Base](#)

CTRL+ K

JOIN



Примечание
Если в условии использованы столбцы из разных таблиц, то пока поддерживается только оператор равенства (=).

Определение ключевых полей для соединения

В рамках этого блока необходимо выполнить задачу классификации для разделения полей на обычные и ключевые, сформировать пары из ключевых полей для построения связей, отсеять наименее подходящие ключевые поля и пары таких полей.

Для выполнения задачи классификации кроме метаданных (таких как названия полей и таблиц) предоставляются также данные из полей (значения ячеек). В случае, если объём исходных данных слишком велик, на вход модели может поступать лимитированная или сэмплированная часть изначальной выборки. Количество строк для каждой таблицы, поступающей на вход, не будет превышать 1 000 000.

В качестве "сигналов" о том, что поле является ключевым, может выступать следующая информация:

- В названии поля содержится сочетание букв "id", "key", "code", "ref", "num", "no", написанные отдельно, слитно с другими словами или через нижнее подчёркивание
- В поле много уникальных значений. Такое поле может быть *первичным* ключом для соединения. Чтобы проверить эту информацию, необходимо число уникальных значений разделить на число всех значений в поле. Чем ближе это значение к 1, тем выше шанс, что поле является ключевым
- В поле много повторяющихся значений. Такое поле может быть *внешним* ключом для соединения. Чтобы проверить эту информацию необходимо выполнить частотный анализ
- Поле используется в сортировке или является Primary Key. Соответствующая информация передаётся из модели данных. В ClickHouse сортировка используется в том числе и для создания первичных ключей, поэтому поля в сортировке представляют повышенный интерес при формировании связей
- В других таблицах есть поля с таким же или похожим (содержащим текст полностью или частично) названием
- В таблице, в которой находится это поле, не было найдено других ключевых полей, или итоговая вероятность присвоению другим полям класса "ключевое" ниже, чем для этого поля
- Булевы значения, списки, массивы и вещественные числовые типы (Float, Decimal) минимизируют шанс того, что поле является ключевым

Используя данную информацию и примеры построения связей в обучающих наборах необходимо сформировать список ключевых полей для каждой из таблиц. Далее необходимо перебрать все возможные пары ключевых полей **ВНУТРИ ГРУПП** и определить, возможна ли связь между ними и её приоритет по следующим правилам (если выполнены все правила, то однозначно присваиваем указанный приоритет, иначе проверяем следующий, если для него совпадений больше – устанавливаем его):

1. Высокий приоритет:

- между таблицами с такими или похожими названиями в обучающем наборе часто устанавливаются связи
- поля имеют одинаковое название
- названия полей типовые
- в одном поле 100% уникальных значений, в другом много повторяющихся
- значительная часть данных совпадает (от 50% для присоединяемой таблицы)
- поле с уникальными значениями используется в сортировке или имеет первичный ключ

2. Средний приоритет:

- между таблицами с такими или похожими названиями в обучающем наборе иногда устанавливаются связи
- поля имеют похожее название
- названия полей содержат типовые сочетания букв
- в одном поле много (более 50%) уникальных значений, в другом есть повторяющиеся
- совпадающих данных мало (10-50% для присоединяемой таблицы)
- в таблице с полем с уникальными значениями нет полей в сортировке или с первичными ключами

3. Низкий приоритет:

- между таблицами с такими или похожими названиями в обучающем наборе связи устанавливаются редко или совсем никогда
- поля отличаются по названию
- названия полей не содержат типовые сочетания букв
- в обоих полях мало (менее 50%) уникальных значений, а частота повторений не превышает 1% от общего объёма выборки для каждого из значений
- совпадающих данных нет или очень мало (до 10% для присоединяемой таблицы)
- в одной или обеих таблицах есть поле в сортировке или с первичным ключом, но это поле не из рассматриваемой пары

4. Нулевой приоритет:

- Существует по меньшей мере одно правило, полностью препятствующее созданию связи (например, из секции Важно или если запрещено повторное использование таблиц)

Важно! Если в таблице уже есть поле для связи с присоединяемой таблицы, то это **ИСКЛЮЧАЕТ** возможность создания ещё одной связи с этой таблицей. Невозможность создания связи между двумя таблицами по нескольким полям актуальна для релиза 1.4.0. В дальнейшем планируется добавить эту возможность – появится необходимость пересмотреть логику определения связи между ключевыми полями

Порядок применения правил:

- Проверяется число выполнения правил для каждого из приоритетов, начиная с высокого и заканчивая низким

- Если в одном из приоритетов наблюдается абсолютное большинство выполнений правил, то устанавливаем этот приоритет
- Если в нескольких правилах наблюдается одинаковое число выполнений правил, то устанавливаем больший приоритет из них

Порядок формирования связей:

1. Сначала расставляются связи с высоким приоритетом для полей, которые были классифицированы как "ключевые" с наибольшей вероятностью
2. После установления каждой связи происходит пересмотр всех остальных: если появилось по меньшей мере одно правило, полностью препятствующее созданию связи, то устанавливаем нулевой приоритет; если в одной из рассматриваемых таблиц появилась связь с другой таблицей – понижаем приоритет на 1 позицию, но не опускаем до нулевого
3. Далее последовательно расставляем остальные связи по мере снижения приоритета и вероятности присвоения класса "ключевое" используемым полям, возвращаясь к пункту 2 после установления каждой связи

Выход модели:

- Группы таблиц: название группы (типовое с номером) и список таблиц в группе
- Связи между таблицами вида Таблица1 INNER/LEFT/RIGHT/FULL JOIN Таблица2 ON Таблица1.поле = Таблица2.Поле
- (Дополнительно) При возможности нейросеть должна генерировать название для каждой группы одним-двумя короткими словами (будет использоваться в качестве названия модели), ChatGPT +- справился с этой задачей:

1. **Income, Goods** — "Прибыль от товаров"
2. **Income, Goods, Budget** — "Финансовый план"
3. **Budget, Costs** — "Расходный бюджет"
4. **DDS, PL** — "Движение финансов"
5. **Shops, Sales, Sellers** — "Торговая сеть"
6. **City, Street, Shops** — "Расположение магазинов"



Требования:

Время работы модели не должно превышать 3 сек.

Работа на кластере (на будущее)

Поскольку в КХ соединение JOIN для больших объёмов данных работает плохо, необходимо минимизировать их применение, соединив наибольшее возможное число таблиц с фактами в одну (например, с помощью UNION ALL).

Порядок действий:

- Нейросеть должна определить, какие из имеющихся таблиц можно отнести к справочникам (к которым будут подтягиваться данные из одной общей таблицы). Справочник можно идентифицировать по наличию пар "ключ-значение", согласно которым возможно формирование связей с таблицей фактов. Остальные таблицы относятся к фактическим.
- Фактические таблицы объединяются скриптом в одну таблицу:
`SELECT Столбцы FROM Таблица1 UNION ALL Столбцы FROM Таблица2 UNION ALL Столбцы FROM Таблица3...`
Скрипт не запускается, а используется для создания таблицы предварительного просмотра, демонстрируемой в отдельном модальном окне.
- Если пользователя устраивает структура, он нажимает кнопку "Продолжить". Иначе – "перегенерировать" (модель запускает ещё одну попытку классификации таблиц на справочники и факты).
- После подтверждения срабатывает сгенерированный скрипт как в обычной ситуации.