СОДЕРЖАНИЕ

Введение 3
1 Обзор существующих аналогов 4
1.1 Национальный статистический комитет Республики Беларусь 4
1.2 Федеральная служба государственной статистики 5
1.3 Всемирный банк 7
2 Формирование требований к базе данных 9
3 Проектирование моделей данных 12
3.1 Определение сущностей и их связей 14
3.2 Определение атрибутов сущностей 20
4 Разработка базы данных 22
4.1 Выбор технических средств 22
4.2 Разработка таблиц базы данных 24
4.3 Разработка хранимых процедур 28
4.4 Разработка триггеров 30
4.5 Разработка индексов 31
5 Тестирование базы данных 32
Заключение 35
Список использованных источников 36
Приложение А (обязательное) Листинг кода 37
Приложение Б (обязательное) Схема базы данных 48
Приложение В (обязательное) Ведомость 49

ВВЕДЕНИЕ

Статистический центр – это организация, которая собирает, анализирует и публикует статистическую информацию. Это может быть информация о экономике, населении, социальном развитии и других аспектах жизни страны или региона. Статистические центры могут быть государственными или частными. Они играют важную роль в формировании экономической и социальной политики государства.

Статистические центры играют важную роль в современном мире, поскольку они предоставляют информацию о различных аспектах жизни общества. Эта информация может использоваться для принятия решений на разных уровнях – от местного до международного.

Например, статистические данные о населении могут помочь правительству планировать инфраструктуру, образование и здравоохранение. Данные о экономике могут помочь бизнесу принимать решения о развитии и инвестициях.

Кроме того, статистические центры помогают отслеживать изменения в обществе и предсказывать будущие тенденции. Это позволяет правительствам и организациям адаптироваться к новым условиям и улучшать свою работу.

Таким образом, статистические центры являются важным инструментом для анализа и понимания данных, которые помогают принимать обоснованные решения и улучшать жизнь людей.

Целью данного курсового проекта является разработка базы данных для статистического центра.

В соответствии с поставленной целью были определены следующие задачи:

– изучить теоретические материалы по выбранной предметной области;

– выделить основные потоки данных, протекающих во время исполнения процессов выбранной предметной области;

– выделить основные сущности, с помощью которых описываются имеющиеся объекты предметной области;

– определить отношения между выделенными сущностями;

– разработать базу данных для статистического центра.

1 ОБЗОР СУЩЕСТВУЮЩИХ АНАЛОГОВ

В мире существует огромное количество статистических центров и почти у всех из них свои базы данных. В современном информационном обществе роль статистических центров становится более актуальной и важной в контексте сбора, анализа и предоставления статистической информации для эффективного управления и развития общества. Почти все страны мира уделяют значительное внимание созданию и развитию государственных статистических центров, которые являются основными источниками объективной и достоверной статистической информации.

Помимо государственных институтов, существуют также коммерческие статистические центры, предоставляющие услуги сбора и анализа данных для бизнес-сектора. Эти организации могут предложить более гибкие и индивидуальные подходы к сбору данных, а также специализированные услуги, направленные на конкретные потребности клиентов.

1.1 Национальный статистический комитет Республики Беларусь

Национальный статистический комитет Республики Беларусь (Белстат) является государственным органом, ответственным за сбор, обработку и распространение статистической информации о социально-экономическом положении страны. Белстат занимается сбором данных от предприятий, организаций и домашних хозяйств, а затем анализирует и публикует полученную информацию. Основными направлениями деятельности Белстата являются экономика, демография, социальная сфера, наука и инновации, рынок труда, уровень жизни населения и др.

Двумя основными функциями Белстата являются: формирование и представление официальной статистической информации и обеспечение ведения баз данных официальной статистической информации. У Белстата разработаны собственные базы и банки данных для хранения официальной статистической информации. А так же собственные программный продукты для формирования и представления официальной статистики.

Вместе базы данных и программные продукты для представления статистической информации формируют интерактивную информационно-аналитическую систему баз данных (ИАС БД) для распространения официальной статистической информации. Она содержит статистическую информацию в динамике за ряд лет по более чем 700 показателям, характеризующим экономическое, демографическое, социальное и экологическое положение в Республике Беларусь. Например, такие показатели как:

– число населенных пунктов на начало периода;
– число родившихся на 1000 человек населения;
– число умерших детей до 1 года на 1000 родившихся;

– валовой внутренний продукт в текущих ценах;
– оплата труда работников в текущих ценах;
– расход топливно-энергетических ресурсов организациями.

Система предоставит Вам возможность самостоятельно формировать запрос, оперативно извлекать из базы данных нужную информацию, выбирать способ ее представления (таблица, график, картограмма), получать официальную статистическую информацию из международных источников, а также сравнить показатели, характеризующие социально-экономическое развитие Республики Беларусь, с показателями других стран мира. На рисунке 1.1 представлена одна из таблиц полученных с помощью информационно-аналитической системы баз данных.

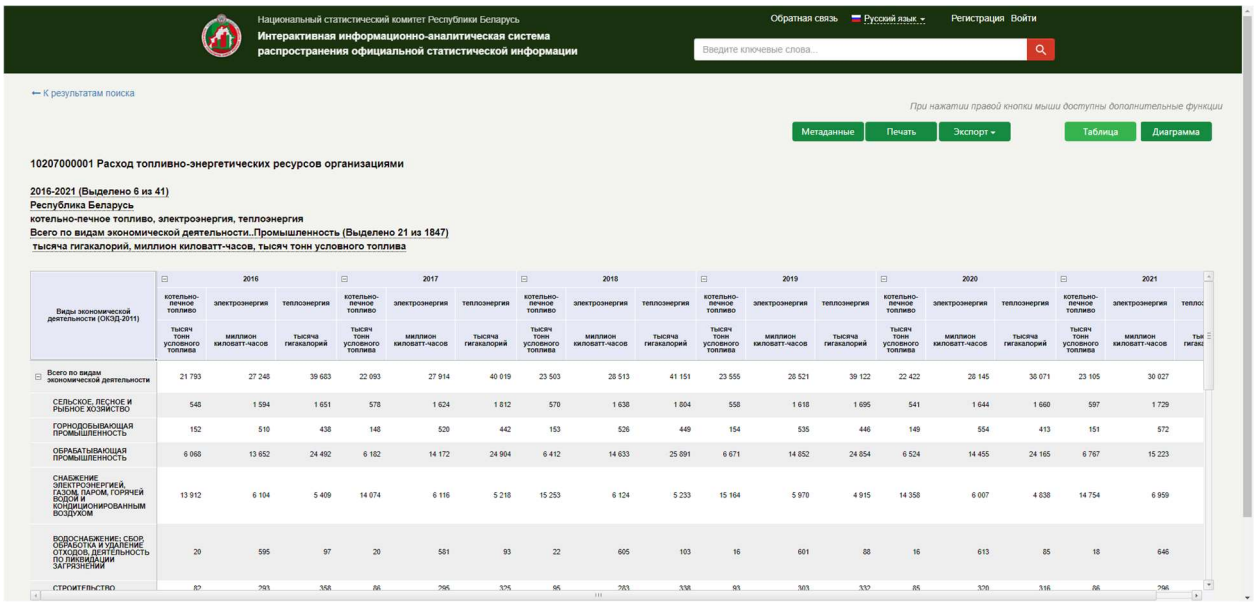


Рисунок 1.1 − Интерфейс ИАС БД

Таким образом можно выделить следующие плюсы ИАС БД:
– объективность и надежность данных;
– широкий спектр данных;
– систематичность и регулярность публикаций;
– поддержание стабильности.
Также наблюдаются следующие минусы:
– недостаток независимости;
– отсутствие свободы информации;
– отсутствие полной синхронизации с мировыми стандартами.

1.2 Федеральная служба государственной статистики

Федеральная служба государственной статистики (Росстат) является федеральным органом исполнительной власти, осуществляющим функции по

формированию официальной статистической информации о социальных, экономических, демографических, экологических и других общественных процессах в Российской Федерации, а также в порядке и случаях, установленных законодательством Российской Федерации по контролю в сфере официального статистического учета [1].

Росстат предоставляет Витрину статистических данных (ВСД) – это интерактивный ресурс, который предоставляет пользователям удобный доступ к различным статистическим данным. Витрина статистических данных Росстата позволяет осуществлять следующие действия:

1 Искать и фильтровать данные. Пользователи могут осуществлять поиск и фильтрацию данных в соответствии с интересующими их параметрами.

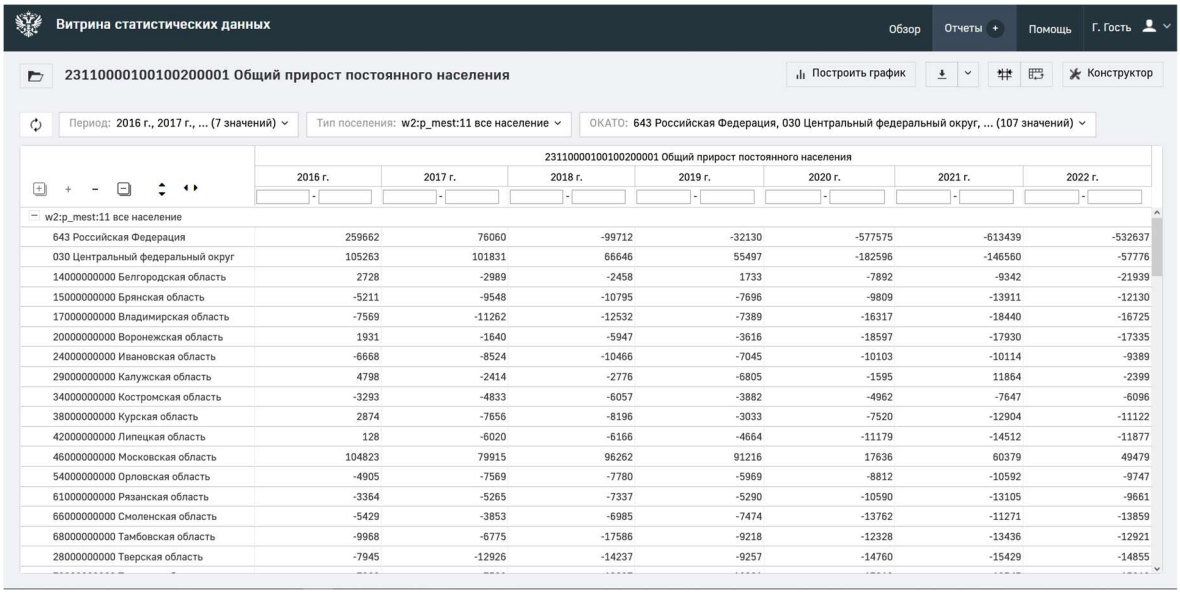
2 Визуализировать данные. Витрина статистических данных предоставляет инструменты для визуализации данных. Это может включать в себя графики, диаграммы, карты и другие визуальные элементы, делая статистическую информацию более понятной и доступной.

3. Пользователи могут сравнивать различные статистические показатели между разными регионами, периодами времени и другими параметрами. Это полезно для выявления тенденций и анализа изменений во времени.

4 Экспортировать данные. Витрина предоставляет возможность экспорта данных в различные форматы, такие как таблицы, графики или файлы для дальнейшего использования в сторонних программных средствах или для подготовки отчетов.

5. Использовать интерактивные карты. Пользователи могут исследовать статистику по регионам, используя картографические средства.

На рисунке 1.2 представлена одна из таблиц полученных с помощью ВСД.



w2:p_mest:11 все население	2016 г.	2017 г.	2018 г.	2019 г.	2020 г.	2021 г.	2022 г.
643 Российская Федерация	259662	76060	-99712	-32130	-577575	-613439	-532637
030 Центральный федеральный округ	105263	101831	66646	55497	-182596	-146560	-57776
14000000000 Белгородская область	2728	-2989	-2458	1733	-7892	-9342	-21939
15000000000 Брянская область	-5211	-9548	-10795	-7696	-9809	-13911	-12130
17000000000 Владимирская область	-7569	-11262	-12532	-7389	-16317	-18440	-16725
20000000000 Воронежская область	1931	-1640	-5947	-3616	-18597	-17930	-17335
24000000000 Ивановская область	-6668	-8524	-10466	-7045	-10103	-10114	-9389
29000000000 Калужская область	4798	-2414	-2776	-6805	-1595	11864	-2399
34000000000 Костромская область	-3293	-4833	-6057	-3882	-4962	-7647	-6096
38000000000 Курская область	2874	-7656	-8196	-3033	-7520	-12904	-11122
42000000000 Липецкая область	128	-6020	-6166	-4664	-11179	-14512	-11877
46000000000 Московская область	104823	79915	96262	91216	17636	60379	49479
54000000000 Орловская область	-4905	-7569	-7780	-5969	-8812	-10592	-9747
61000000000 Рязанская область	-3364	-5265	-7337	-5290	-10590	-13105	-9661
66000000000 Смоленская область	-5429	-3853	-6985	-7474	-13762	-11271	-13859
68000000000 Тамбовская область	-9968	-6775	-17586	-9218	-12328	-13436	-12921
28000000000 Тверская область	-7945	-12926	-14237	-9257	-14760	-15429	-14855

Рисунок 1.2 − Интерфейс ВСД

1.3 Всемирный банк

Всемирный банк – это международная финансовая организация, созданная с целью предоставления финансовой и технической помощи развивающимся странам.

Всемирный банк предоставляет различные инструменты и ресурсы для работы со статистическими данными. Такие как:

1 Данные Всемирного банка. Это онлайн-платформа, которая предоставляет доступ к разнообразным статистическим данным по странам и регионам мира. Пользователи могут проводить поиск, фильтрацию, а также скачивать данные для дальнейшего анализа.

2 Индикаторы мирового развития. Этот набор данных включает в себя ключевые показатели по мировому развитию, такие как данные о бедности, занятости, образовании, здравоохранении и другие. Индикаторы предоставляются в виде таблиц и графиков.

3 Открытые данные. Всемирный банк активно поддерживает инициативу открытых данных, предоставляя свободный доступ к широкому спектру статистических данных. Эти данные могут быть использованы для различных целей, включая исследования, образование и разработку.

4 База данных. Это веб-приложение, позволяющее пользователям исследовать, сравнивать и анализировать данные по различным показателям и временным периодам. База данных предоставляет гибкий интерфейс для работы с множеством статистических данных.

5. Библиотека микроданных. Этот ресурс предоставляет доступ к микроданным, которые могут быть использованы для более детального исследования структуры и характеристик статистических данных.

6 Всемирное интегрированное торговое решение. Этот инструмент предоставляет доступ к мировой торговой статистике. Он включает данные о торговле товарами и услугами для различных стран и регионов.

7 Интерфейс программирования приложений. Всемирный банк предоставляет интерфейс для доступа к статистическим данным, что позволяет разработчикам интегрировать данные в свои приложения и аналитические инструменты.

На рисунке 1.3 представлена таблица, полученная с помощью базы данных Всемирного банка.

Можно выделить следующие плюсы Всемирного банка:

– мировой охват данных;
– широкий спектр данных;
– доступность данных;
– международная методология.

Также наблюдаются следующие минусы:

– ограниченная детализация статистических данных;
– временная задержка результатов;

– ориентация на экономические аспекты;
– зависимость от добровольных отчетов.

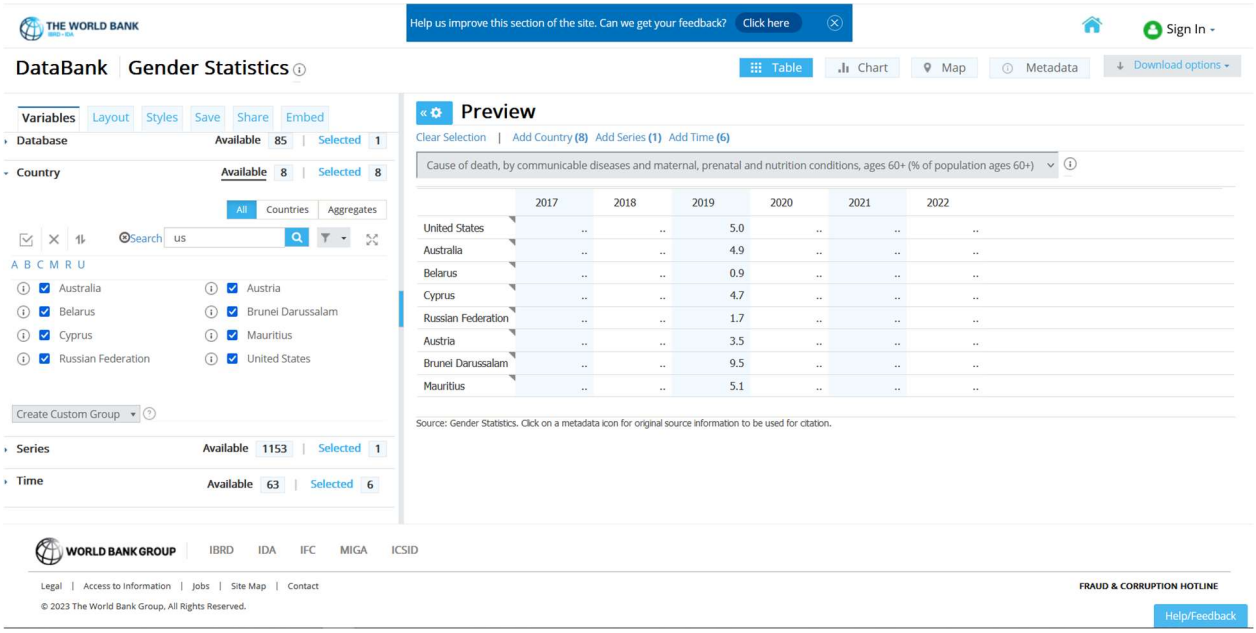


Рисунок 1.3 − Интерфейс базы данных Всемирного банка

2 ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К БАЗЕ ДАННЫХ

База данных для статистического центра должна обладать рядом важных качеств, учитывая специфику и требования обработки статистических данных.

Необходимо, чтобы база данных была спроектирована и реализована таким образом, чтобы обеспечить максимальную надежность и целостность данных. Так как обеспечение надежности и целостности данных является фундаментальным аспектом управления информацией в базах данных. Надежные и целостные данные создают основу для доверия к информации в базе данных, что крайне важно для пользователей. Также существуют законы и нормы, регулирующие обработку данных, и обеспечение целостности данных становится важным аспектом для соблюдения этих требований. Это также способствует защите от потери информации в результате сбоев, сбросов или других технических проблем. Целостные данные минимизируют ошибки и проблемы, связанные с некорректными или несогласованными данными, что, в свою очередь, упрощает процессы работы с информацией и повышает производительность пользователей. Они также способствуют поддержанию консистентности в базе данных, предотвращая противоречия и несогласованность в информации.

В контексте статистических центров, где точность и достоверность данных критичны для оценки социально-экономических показателей, обеспечение надежности и целостности данных существенно влияет на качество предоставляемого сервиса. Ненадежные или поврежденные данные могут повлиять на репутацию организации или статистического центра, в то время как качественные и достоверные данные способствуют поддержанию положительной репутации.

Обеспечение высокой производительности для базы данных статистического центра представляет собой важный аспект для реализации по нескольким ключевым причинам. Во-первых, эффективность обработки запросов играет решающую роль в оперативном анализе данных и принятии решений, удовлетворяя ожидания пользователей. Высокая производительность обеспечивает отзывчивость системы, что существенно для оперативной работы аналитиков и исследователей. Во-вторых статистические центры оперируют большим объемом разнообразных данных, включая временные ряды, социально-экономические показатели и географическую информацию. Поэтому база данных должна быть способной эффективно обрабатывать масштабные объемы информации. Производительность базы данных также оказывает влияние на оптимизацию аналитических процессов, что важно для выявления тенденций и статистических закономерностей.

Необходимо обеспечить масштабируемость для базы данных статистического центра. Обеспечение масштабируемости для базы данных статистического центра представляет собой критически важный аспект с

учетом нескольких основных причин. Прежде всего, статистические центры, занимающиеся обширными объемами данных о населении, экономике и социальных процессах, сталкиваются с постоянным ростом данных. Масштабируемость базы данных становится необходимой для эффективного управления этим ростом и обеспечения беспрепятственного функционирования.

Гибкость в обработке нагрузки также является важным аспектом, особенно учитывая временную изменчивость данных в статистических центрах, например, в периоды проведения переписи населения. Масштабируемость обеспечивает адаптацию к колебаниям в нагрузке, что поддерживает стабильность работы базы данных.

Масштабируемость также предоставляет гибкость для будущего развития системы, обеспечивая возможность легкого добавления новых ресурсов и соответствие изменяющимся требованиям. Это особенно важно в контексте статистических центров, где система должна адаптироваться к постоянно меняющейся информационной среде и оставаться устойчивой в долгосрочной перспективе.

Процесс нормализации [2] в базах данных играет ключевую роль в обеспечении эффективности и структурированности данных. В первую очередь, первая нормальная форма (1НФ) требует, чтобы каждая таблица имела уникальный первичный ключ, обеспечивая уникальную идентификацию каждой записи и исключая дублирование данных.

Вторая нормальная форма (2НФ) углубляет этот принцип, требуя полной зависимости всех неключевых атрибутов от всего первичного ключа. Это предотвращает возможные аномалии данных и способствует более структурированной базе данных.

Третья нормальная форма (3НФ) дополняет процесс, устраняя транзитивные зависимости между неключевыми атрибутами. Это обеспечивает гибкость в организации данных и снижает избыточность.

Применение этих принципов нормализации не только повышает структурированность данных, но и уменьшает вероятность ошибок в их обработке. Обеспечивая целостность и минимизацию избыточности данных, база данных становится более эффективной для выполнения запросов и управления информацией.

Данные организуются в виде таблиц, где каждая таблица представляет сущность, а строки этой таблицы представляют отдельные записи, каждая из которых уникально идентифицируется первичным ключом.

Необходимо использование внешних ключей для установления связей между таблицами. Это обеспечивает согласованность данных и позволяет эффективно связывать информацию из различных таблиц.

База данных должна быть нормализована согласно требованиям нормальных форм. До третьей нормальной формы включительно, что означает минимизацию избыточности данных и обеспечение их целостности.

Относительно количества сущностей, система должна содержать 25 сущностей. Каждая из этих сущностей представляет определенный объект или концепцию, а их атрибуты должны быть логически и связанно представлены внутри реляционной модели.

Общие требования поддержания целостности данных, избегание избыточности и обеспечение удобства обработки запросов и управления информацией должны быть обеспечены с использованием реляционной модели данных.

Разрабатываемая в настоящий момент база данных, включающая в себя вышеописанные критерии, согласно требованиям текущего проекта должна быть реализована с использованием исключительно реляционной модели данных.

3 ПРОЕКТИРОВАНИЕ МОДЕЛЕЙ ДАННЫХ

В данном разделе рассматривается проектирование моделей данных с использованием как просто описания словами, так и диаграмм. Диаграммы рассматриваются по убыванию абстрактности до физической модели данных.

Проектирование моделей данных в ходе разработки программного обеспечения представляет собой ключевой этап, нацеленный на обеспечение эффективности, надежности и удобства поддержки системы. Оно направлено на формирование структурированного и ясного представления о том, как данные будут организованы, взаимодействовать и использоваться в рамках приложения. Модели данных на этом уровне также служат основой для обеспечения целостности данных, оптимизации операций с базой данных, обеспечения согласованности в команде разработчиков, обеспечения легкости внесения изменений, обеспечения безопасности и конфиденциальности данных, а также поддержки процессов тестирования и отладки. Грамотно разработанные модели данных создают прочную основу для будущего расширения системы, обеспечивая гибкость и адаптивность к изменяющимся требованиям.

При переходе на концептуальный уровень моделирования баз данных выделяются ключевые концепции и отношения, существующие в предметной области, вне зависимости от того, как они будут физически реализованы в базе данных. Этот уровень моделирования фокусируется на описании структуры данных и их взаимосвязей, освобожденных от технических деталей и специфик систем управления базами данных. Здесь выделяются сущности, их атрибуты, отношения, ключи, ассоциации и кардинальность. Важным аспектом на данном уровне является также возможность проведения нормализации для избежания избыточности данных и обеспечения их целостности.

Перейдя на логический уровень, фокус смещается на более детализированные аспекты, связанные с конкретной реализацией базы данных. Описываются структуры таблиц, ключи, индексы и другие технические детали, которые напрямую связаны с функционированием системы управления базами данных. Логический уровень обеспечивает более конкретное представление о том, как концепции концептуального уровня будут технически воплощены в реальной базе данных.

Логический уровень моделирования баз данных является абстрактным, высокоуровневым представлением данных и их взаимосвязей, что обеспечивает лучшее понимание структуры предметной области и формирует основу для последующего проектирования и реализации физической базы данных.

Разрабатываемую базу данных можно разбить на два зависимых участка:

– часть, общая для всех статистических характеристик;

– конкретные статистические характеристики.

Участок, который отвечает за хранение информации о рождении детей представлен диаграммой IDEF1X на рисунке 3.1.

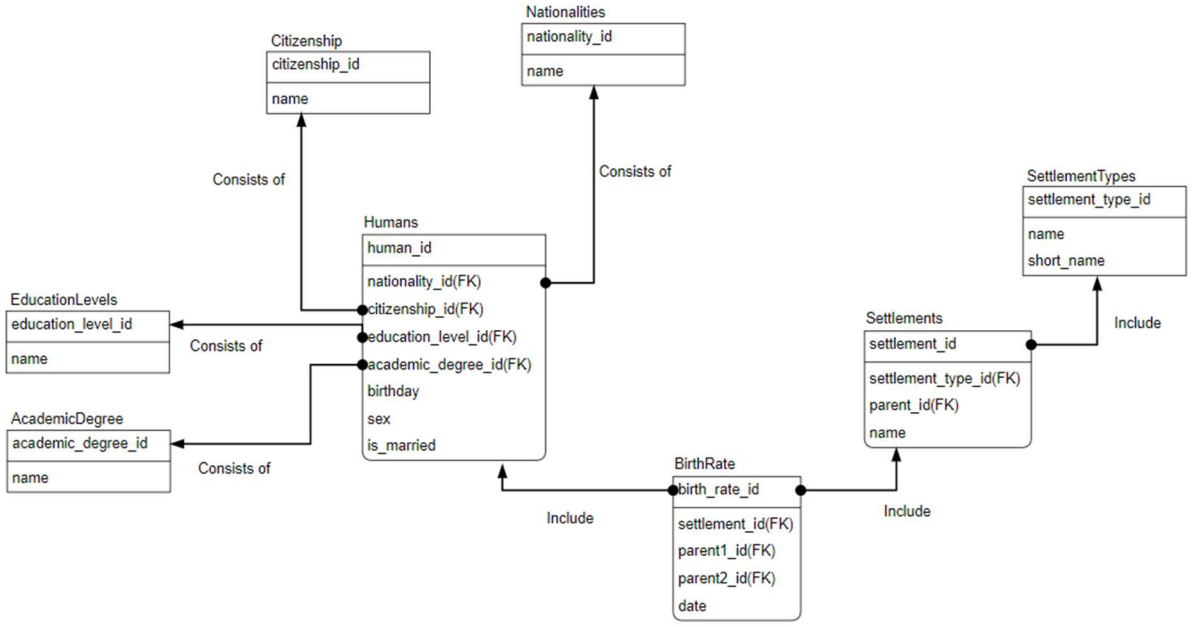


Рисунок 3.1 − Диаграмма IDEF1X показывающая хранение рождаемости

На данной диаграмме есть независимые таблицы, которые хранят название какого-то типа или какой-нибудь характеристики. Например, таблица с национальностями хранит все национальности определенные в системе. И есть зависимые таблицы, которые описывают такие сущности как человек, определенная территория и запись о рождении ребенка. Человек содержит в себе информацию интересную для статистики. Например, какой национальности родители ребенка, или какой уровень образования у родителей ребенка.

Участок, который отвечает за хранение смертей представлен диаграммой IDEF1X на рисунке 3.2.

В отличии от хранения рождения детей, для хранения смертей недостаточно сущностей территория и человек. Для этого еще необходима сущность причина смерти. Причина смерти описывается названием и кодом МКВ-10. МКБ-10 – международная классификация болезней десятого пересмотра. Представляет собой нормативный документ с общепринятой статистической классификацией медицинских диагнозов, которая используется в здравоохранении для унификации методических подходов и международной сопоставимости материалов [3].

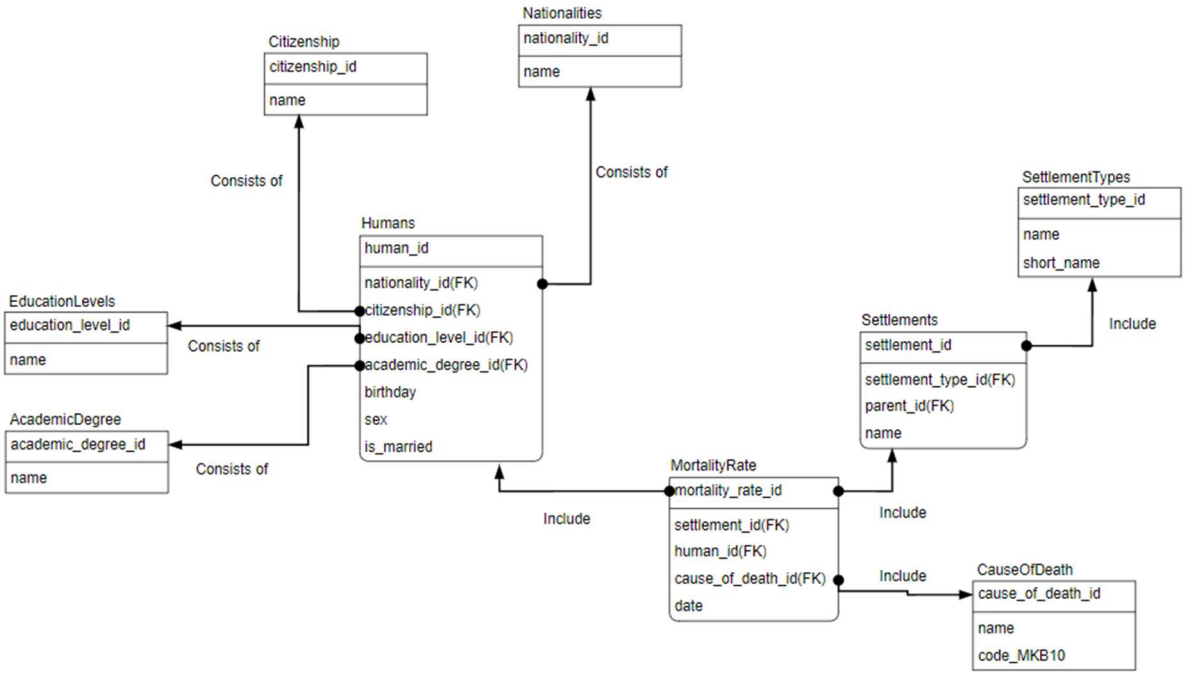


Рисунок 3.2 − Диаграмма IDEF1X показывающая хранение смертей

3.1 Определение сущностей и их связей

В результате исследования предметной области в ней были выделены сущности. Сущности и их описание перечислены в таблице 3.1.

Таблица 3.1 – Сущности системы

Сущность	Описание
Ученая степень	Степень квалификационной системы в науке, позволяющей ранжировать научных деятелей на отдельных этапах академической карьеры. Решение о присуждении учёной степени базируется на оценке только научно-исследовательского уровня соискателя. Стаж в конкретной должности, педагогические достижения и иные показатели не учитываются, в отличие от ситуации присвоения учёных званий[4].
Адрес	Уникальная структурированная иерархическая совокупность элементов, идентифицирующая местонахождение [5].
Цели прибытия	Цель прибытия на новое место. Самые частые цели прибытия: работа или учеба. Но также могут встречаться более специфичные цели.

Продолжение таблицы 3.1

Сущность	Описание
Запись о прибытии	Запись о прибытие содержит важную информацию для статистики. И включает следующие данные: характеристики человека, который прибыл, дату прибытия, откуда и куда прибыл, а также цель прибытия.
Запись о рождении	Запись о рождении содержит важную информацию для статистики. И включает следующие данные: характеристики родителей, дату и время рождения, а также место рождения.
Причина смерти	Причины смерти все болезни, патологические состояния и травмы, которые привели к смерти или способствовали ее наступлению, а также обстоятельства происшествия или акта насилия, которые привели к таким травмам. В целях демографической статистики, симптомы или стадии смерти, такие как остановка сердца или астения, не считаются причинами смерти [6].
График	Представляет определенные в системе графики. Описывается названием графика и описанием.
Гражданство	Правовая связь человека и государства, выражающаяся в совокупности их взаимных прав, обязанностей и ответственности [7]. Представляет определенные в системы гражданства.
Источник данных	Представляет источник данных, из которого были получены определенные данные.
Отдел	Представляет структурное подразделение какой-нибудь организации.
Запись о разводах	Запись о разводах содержит важную информацию для статистики. И включает следующие данные: характеристики людей, которые разводятся, дату развода, а также количество детей.
Уровень образования	Представляет собой уровень образования определенного человека. В Беларуси определены следующие уровни образования: дошкольное, общее среднее, профессионально-техническое, среднее специальное, высшее.
Должность	Представляет собой служебное положение работника, обусловленное кругом его обязанностей, должностными правами и характером ответственности [8].

Продолжение таблицы 3.1

Сущность	Описание
Цель отбытия	Цель отбытия на новое место. Важная характеристика, по которой можно анализировать, почему люди покидают определенные места. Самые частые цели отбытия: работа или учеба. Но также могут встречаться более специфичные цели.
Человек	Представляет собой наиболее важные для статистики характеристики человека. Например, такие характеристики, как: дата рождения, пол, национальность, гражданство, уровень образования, ученую степень, и в браке ли находится человек.
Запись об отбытии	Запись об отбытии содержит важную информацию для статистики. И включает следующие данные: характеристики человека, который отбыл, дату отбытия, откуда и куда отбыл, а также цель отбытия.
Запись о браке	Запись о браке содержит важную информацию для статистики. И включает следующие данные: характеристики людей, которые вступают в брак, дату вступления в брак, место вступления в брак.
Запись о смерти	Запись о смерти содержит важную информацию для статистики. И включает следующие данные: характеристики человека, дату и время, место и причину.
Национальность	Принадлежность лица к этнической группе, характеризуемая родным языком, особенностями быта, традициями, обычаями, культурой, религией, родством и другими признаками, позволяющими лицу идентифицировать себя [9].
Организация	Представляет собой основную информацию об организации, такую как: адрес, название, номер телефона и адрес электронной почты.
Тип территории	Представляет собой определенный тип территории. Например, страна, город, область, район, агрогородок, деревня.
Территория	Представляет собой конкретную территорию. Например, город Минск, Гродненская область, агрогородок Рожанка и т.д.
Пользователь	Представляет собой определенного пользователя системы.

Продолжение таблицы 3.1

Сущность	Описание
Статистический показатель	Количественно выраженное определенное свойство, качество совокупности в целом или ее частей [10].
Интервал времени	Представляет собой непустой отрезок временной шкалы.
Единица измерения	Конкретная величина, определенная и установленная по договоренности, с которой сопоставляются другие величины того же рода, для того чтобы выразить их размер по отношению к указанной величине [11].

Как видно из таблицы 3.1 часть сущностей является общей для всей базы данных статистического центра. А часть сущностей является специфическими для определенных статистических показателей.

При разработке структуры базы данных выделены отношения между сущностями. Данные отношения можно увидеть в таблица 3.2.

Таблица 3.2 – Отношения между сущностями

Название связи	Связываемые сущности	Пояснение
Содержит	Адрес – Территория	В адрес входит определенная территория. Помимо территории в адрес входит конкретная улица на территории, и конкретный дом на улице.
Содержит	Запись о прибытии – Цель прибытия	Запись о прибытии содержит в себе данные, важные для дальнейших статистических исследований. Цель прибытия является интересной для исследователей характеристикой.
Содержит	Запись о прибытии – Человека	Запись о прибытии содержит в себе данные, важные для дальнейших статистических исследований. Характеристики человека интересуют исследователей.
Содержит	Территория– Территория	Территория может содержать в себе другую, более маленькую территорию.

Продолжение таблицы 3.2

Название связи	Связываемые сущности	Пояснение
Содержит	Запись о прибытии – Территория	Запись о прибытии содержит в себе данные, важные для дальнейших статистических исследований. Откуда и куда прибыл человек являются интересными данными для исследователей.
Содержит	Запись об отбытии – Цель отбытия	Запись об отбытии содержит в себе данные, важные для дальнейших статистических исследований. Цель отбытия является интересной для исследователей характеристикой.
Содержит	Запись об отбытии – Человека	Запись об отбытии содержит в себе данные, важные для дальнейших статистических исследований. Характеристики человека интересуют исследователей.
Содержит	Запись об отбытии – Территория	Запись об отбытии содержит в себе данные, важные для дальнейших статистических исследований. Откуда и куда прибыл человек являются интересными данными для исследователей.
Содержит	Запись о разводе – Человек	Запись о разводе содержит в себе данные, важные для дальнейших статистических исследований. Характеристики человека интересуют исследователей.
Содержит	Запись о разводе – Территория	Запись о разводе содержит в себе данные, важные для дальнейших статистических исследований. Исследователей интересует где произошел развод.
Имеет	Человек– Гражданство	Определенный человек имеет гражданство. А гражданство человека может быть интересно в статистических исследованиях.

Продолжение таблицы 3.2

Название связи	Связываемые сущности	Пояснение
Содержит	Запись о браке – Территория	Запись о браке содержит в себе данные, важные для дальнейших статистических исследований. Исследователей интересует где был заключен брак.
Содержит	Запись о браке – Человек	Запись о браке содержит в себе данные, важные для дальнейших статистических исследований. Характеристики человека интересуют исследователей.
Содержит	Запись о рождении – Территория	Запись о рождении ребенка содержит в себе данные, важные для дальнейших статистических исследований. Исследователей интересует где был рожден ребенок.
Содержит	Запись о рождении – Человек	Запись о рождении ребенка содержит в себе данные, важные для дальнейших статистических исследований. Характеристики родителей ребенка интересуют исследователей.
Содержит	Запись о смерти – Территория	Запись о смерти содержит в себе данные, важные для дальнейших статистических исследований. Исследователей интересует где умер человек.
Содержит	Запись о смерти – Человек	Запись о смерти содержит в себе данные, важные для дальнейших статистических исследований. Характеристики умершего человека интересуют исследователей.
Содержит	Запись о смерти – Причина смерти	Запись о смерти содержит в себе данные, важные для дальнейших статистических исследований. Исследователей интересует причина смерти человека.

Продолжение таблицы 3.2

Название связи	Связываемые сущности	Пояснение
Имеет	Человек– Национальность	Определенный человек имеет национальность. А национальность человека интересна в статистических исследованиях.
Имеет	Человек– Гражданство	Определенный человек имеет гражданство. А гражданство человека может быть интересно в статистических исследованиях.
Имеет	Человек– Уровень образования	Определенный человек имеет определенный уровень образования. А уровень образования человека может быть интересным в статистических исследованиях.
Имеет	Человек– Ученая степень	Определенный человек имеет определенную ученую степень. А ученая степень человека может быть интересной в статистических исследованиях.
Содержит	Территория– Территория	Территория может содержать в себе другую, более маленькую территорию.
Имеет	Пользователь– Должность	Каждый пользователь имеет определенную должность.

3.2 Определение атрибутов сущностей

В данном пункте проводится анализ и описание атрибутов, характеризующих сущности, представленные в структуре базы данных. Каждая таблица базы данных представляет собой определенный тип сущности, а ее поля (атрибуты) определяют характеристики этой сущности. Данный раздел предоставляет подробное описание каждой сущности, выделяя ее ключевые атрибуты и их типы данных. Анализ атрибутов в контексте текущей базы данных позволяет лучше понять структуру данных, их взаимосвязи и обеспечивает основу для последующего проектирования запросов и взаимодействия с информацией в рамках разрабатываемого программного продукта.

Увидеть сущности и их атрибуты можно в таблице 3.3.

Таблица 3.3 – Атрибуты сущностей

Сущность	Основные атрибуты
Positions	position_id, name
Departments	department_id, name
Users	user_id, name, surname, patronymic, phone_num, city_telephone_num, email, position_id, department_id
UnitsOfMeasure	unit_of_measure_id, name, description
TimeIntervals	time_interval_id, name, interval_in_seconds
Charts	chart_id, name, description
SettlementsType	settlement_type_id, name, short_name
Settlements	settlement_id, settlements_type_id, parent_id, name
Addresses	address_id, settlement_id, street, house_num
Organizations	organization_id, address_id, name, short_name, phone_num, city_telephone_num, email
Nationalities	nationality_id, name
Citizenships	citizenship_id, name
EducationLevels	education_level_id, name
AcademicDegrees	academic_degree_id, name
Humans	human_id, birthday, sex, nationality_id, citizenship_id, education_level_id, academic_degree_id, is_married
BirthRate	birth_rate_id, date, settlement_id, parent1_id, parent2_id
CausesOfDeath	cause_of_death_id, name, code_MKB10
MortalityRate	mortality_rate_id, human_id, cause_of_death_id, settlement_id, date
Marriages	marriage_id, human1_id, human2_id, settlement_id, date
Divorces	divorce_id, human1_id, human2_id, settlement_id, date, children_count
ArrivalGoal	arrival_goal_id, goal
ArrivalRate	arrival_rate_id, human_id, date, from_id, to_id, goal_id
LeavingGoal	leaving_goal_id, goal
LeavingRate	arrival_rate_id, human_id, date, from_id, to_id, goal_id
ConceptsAndDefinitions	concept_and_definition_id, concept, definition
DataSources	data_source_id, name, reliability
StatisticalIndicators	statistical_indicator_id, name, last_update, creation_date, comment, contact_person_id

4 РАЗРАБОТКА БАЗЫ ДАННЫХ

Разработка базы данных – ключевой этап в создании информационной системы, направленной на эффективное хранение, обработку и управление данными. В данном разделе мы углубимся в процесс разработки базы данных, рассмотрим выбор технических средств, создание таблиц, разработку хранимых процедур, триггеров и индексов.

4.1 Выбор технических средств

В ходе процесса выбора системы управления базами данных (СУБД) для реализации разработанной модели данных, был проведен тщательный анализ нескольких популярных реляционных СУБД, с целью определения оптимального решения, соответствующего требованиям проекта. Рассмотрим основные аналоги, которые рассматривались в контексте принятия решения: Oracle Database, MySQL, SQLite иPostgreSQL и Microsoft SQL Server.

Oracle Database представляет собой мощную СУБД с высокой производительностью и обширными функциональными возможностями. Способна масштабироваться от небольших приложений до крупных предприятий, обрабатывая огромные объемы данных. Поддерживает транзакционную целостность данных, гарантируя надежность и согласованность в рамках транзакций. Имеет возможность интеграции с другими приложениями и обширные инструменты для управления базой данных, включая средства мониторинга и отчетности. Однако, она является проприетарным решением с высокими затратами на лицензирование, что может быть ограничивающим фактором для некоторых проектов.

MySQL – это открытая реляционная система управления базами данных, широко используемая для хранения, управления и обработки данных. MySQL, отличающаяся высокой производительностью, также она поддерживает транзакции. Способна масштабироваться от небольших веб-приложений до крупных корпоративных систем, обрабатывая различные объемы данных. Встроенные механизмы безопасности, такие как шифрование данных, аутентификация и авторизация, обеспечивают защиту информации. Обширное сообщество пользователей и разработчиков MySQL обеспечивает доступ к множеству ресурсов, включая форумы, документацию и обновления. При этом всем MySQL распространяется под лицензией General Public License (GPL), что делает ее бесплатной и доступной для широкого круга пользователей. Однако, она может иметь ограниченные функциональные возможности по сравнению с некоторыми другими СУБД.

SQLite – это встраиваемая реляционная система управления базами данных общего назначения, которая является самодостаточной и не требует отдельного сервера для своей работы. SQLite выделяется легкостью в развертывании, не требует сервера баз данных и прост в использовании.

Однако, она не предназначена для крупных и сложных проектов, имеет ограниченные возможности масштабирования.

Microsoft SQL Server — система управления реляционными базами данных, разработанная корпорацией Microsoft [12]. MS SQL Server интегрирован с другими продуктами и технологиями Microsoft, такими как .NET, Azure, Microsoft Excel, что обеспечивает совместимость и удобство использования в экосистеме Microsoft. Способен масштабироваться от небольших веб-приложений до крупных корпоративных систем, обрабатывая различные объемы данных. Включает в себя инструменты для администрирования, мониторинга и оптимизации производительности баз данных. Но лицензионные ограничения могут быть затратными.

PostgreSQL – это мощная объектно-реляционная система управления базами данных с открытым исходным кодом. PostgreSQL поддерживает множество расширенных функций SQL, включая сложные запросы, хранимые процедуры, триггеры, оконные функции и многое другое. Способен масштабироваться от небольших проектов до крупных предприятий, обрабатывая сложные и объемные базы данных. Обеспечивает надежное хранение данных, поддержку транзакций, резервное копирование и механизмы восстановления после сбоев. PostgreSQL имеет активное сообщество пользователей и разработчиков, а также обширную документацию, что облегчает использование и поддержку. Но может быть тяжелее других СУБД в настройке.

Обобщим всё вышесказанное в таблицу 4.1.

Таблица 4.1 – Сравнение СУБД

	Oracle DB	MySQL	SQLite	PostgreSQL	Microsoft SQL Server
Высокая производительность	+	+	+	+	+
Масштабируемость	+	+	-	+	+
Наличие опыта у разработчиков	+	+	+	-	-
Поддержка транзакций	+	+	+	+	+
Свободная и открытая лицензия	-	+	+	+	-
Обширное сообщество и поддержка	+	+	+	+	+

Таким образом в ходе рассмотрения различных аналогов систем управления базами данных и тщательного их анализа, в качестве СУБД для реализации данного курсового проекта была выбрана MySQL.

4.2 Разработка таблиц базы данных

В результате разработки вышеописанных моделей данных на основе технического задания получен список отношений и таблиц, соответствующих сущностям логической модели данных.

Таблица «Positions» описывает сущность должности. Здесь «position_id» служит идентификатором, а «name» хранит название должности. Использование AUTO_INCREMENT для «position_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «Departments» описывает сущность структурного подразделения. Здесь «department_id» служит идентификатором, а «name» хранит название структурного подразделения. Использование AUTO_INCREMENT для «department_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «Users» описывает сущность пользователя. Здесь «user_id» служит идентификатором, «name» хранит имя, «surname» хранит фамилию, «patronymic» хранит отчество, «phone_num» хранит номер телефона, «city_telephone_num» хранит номер городского телефона, «email» хранит адрес электронной почты, «position_id» хранит идентификатор должности, а «department_id» хранит идентификатор структурного подразделения. Использование AUTO_INCREMENT для «user_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(position_id) связывает пользователя с должностью, а FOREIGN KEY(department_id) – со структурным подразделением.

Таблица «UnitsOfMeasure» описывает сущность единицы измерения. Здесь «unit_of_measure_id» служит идентификатором, «name» хранит название, а «description» предоставляет описание данной единицы измерения. Использование AUTO_INCREMENT для «unit_of_measure_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. Текстовое поле «description» оставлено nullable, что позволяет сохранять единицы измерения без подробных описаний, если они не нужны.

Таблица «TimeIntervals» описывает сущность временного интервала. Здесь «time_interval_id» служит идентификатором, «name» хранит название интервала, а «interval_in_seconds» размер данного интервала в секундах. Использование AUTO_INCREMENT для «time_interval_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «Charts» описывает сущность графика. Здесь «chart_id» служит идентификатором, «name» хранит название, а «description» предоставляет описание. Использование AUTO_INCREMENT для «chart_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении

новых записей. Текстовое поле «description» оставлено nullable, что позволяет сохранять графики без подробных описаний, если они не нужны.

Таблица «SettlementsType» описывает сущность типа территории. Здесь «settlement_type_id» служит идентификатором, «name» хранит название, а «short_name» предоставляет сокращение. Использование AUTO_INCREMENT для «settlement_type_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. Текстовое поле «short_name» оставлено nullable, что позволяет сохранять типы территорий без подробных описаний, если они не нужны.

Таблица «Settlements» описывает сущность территории. Здесь «settlement_id» служит идентификатором, «name» хранит название, «settlement_type_id» хранит идентификатор типа территории, а «parent_id» хранит идентификатор на территорию, в которую включена данная территория. Использование AUTO_INCREMENT для «settlement_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(settlement_id) связывает территорию с родительской территорией, а FOREIGN KEY(settlement_type_id) – с типом территории.

Таблица «Addresses» описывает сущность адреса. Здесь «address_id» служит идентификатором, «settlement_id» хранит идентификатор территории, «street» хранит название улицы, а «house_num» хранит номер дома. Использование AUTO_INCREMENT для «address_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(settlement_id) связывает адрес с конкретной территорией.

Таблица «Organizations» описывает сущность организации. Здесь «organization_id» служит идентификатором, «address_id» хранит идентификатор адреса, «name» хранит название организации, «short_name» хранит краткое название, «phone_num» хранит номер телефона, «city_telephone_num» хранит номер городского телефона, а «email» хранит адрес электронной почты. Использование AUTO_INCREMENT для «organization_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(address_id) связывает организацию с конкретным адресом. Текстовые поля «short_name», «phone_num», «city_telephone_num» и «email» оставлены nullable, что позволяет сохранять организации без эти необязательных полей.

Таблица «Nationalities» описывает сущность национальности. Здесь «nationality_id» служит идентификатором, а «name» хранит название. Использование AUTO_INCREMENT для «nationality_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «Citizenships» описывает сущность гражданства. Здесь «citizenship_id» служит идентификатором, а «name» хранит название.

Использование AUTO_INCREMENT для «citizenship_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «EducationLevels» описывает сущность уровеня образования. Здесь «education_level_id» служит идентификатором, а «name» хранит название. Использование AUTO_INCREMENT для «education_level_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «AcademicDegrees» описывает сущность ученой степени. Здесь «academic_degree_id» служит идентификатором, а «name» хранит название. Использование AUTO_INCREMENT для «academic_degree_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «Humans» описывает сущность человека. Здесь «human_id» служит идентификатором, «birthday» хранит дату рождения, «sex» хранит пол, «nationality_id» хранит идентификатор национальности, «citizenship_id» хранит идентификатор гражданства, «education_level_id» хранит идентификатор уровня образования, «academic_degree_id» хранит идентификатор ученой степени, а «is_married» показвает, находится ли данный человек в браке. Использование AUTO_INCREMENT для «human_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(nationality_id) связывает человека с национальностью, FOREIGN KEY(citizenship_id) – с гражданством, FOREIGN KEY(education_level_id) – с уровнем образования, а FOREIGN KEY(academic_degree_id) – с ученой степенью.

Таблица «BirthRate» описывает сущность записи о рождении. Здесь «birth_rate_id» служит идентификатором, «date» хранит дату и время рождения, «settlement_id» хранит идентификатор территории, «parent1_id» хранит идентификатор первого родителя, «parent2_id» хранит идентификатор второго родителя. Использование AUTO_INCREMENT для «birth_rate_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(settlement_id) связывает запись о рождении с территорией, FOREIGN KEY(parent1_id) – с первым родителем, FOREIGN KEY(parent2_id) – со вторым родителем.

Таблица «CausesOfDeath» описывает сущность причины смерти. Здесь «cause_of_death_id» служит идентификатором, «name» хранит название, а «code_MKB10» хранит код причины смерти в формате МКБ-10. Использование AUTO_INCREMENT для «cause_of_death_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «MortalityRate» описывает сущность записи о смерти. Здесь «mortality_rate_id» служит идентификатором, «date» хранит дату и время смерти, «settlement_id» хранит идентификатор территории, «human_id» хранит

идентификатор человека, «cause_of_death_id» хранит идентификатор причины смерти. Использование AUTO_INCREMENT для «mortality_rate_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(settlement_id) связывает запись о смерти с территорией, FOREIGN KEY(human_id) – с человеком, FOREIGN KEY(cause_of_death_id) – с причиной смерти.

Таблица «Marriages» описывает сущность записи о браке. Здесь «marriage_id» служит идентификатором, «date» хранит дату заключения брака, «settlement_id» хранит идентификатор территории, «human1_id» хранит идентификатор первого человека, «human2_id» хранит идентификатор второго человека. Использование AUTO_INCREMENT для «marriage_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(settlement_id) связывает запись о браке с территорией, FOREIGN KEY(human1_id) – с первым человеком, FOREIGN KEY(human2_id) – со вторым человеком.

Таблица «Divorces» описывает сущность записи о разводе. Здесь «divorce_id» служит идентификатором, «date» хранит дату развода, «settlement_id» хранит идентификатор территории, «human1_id» хранит идентификатор первого человека, «human2_id» хранит идентификатор второго человека, а «children_count» хранит количество детей. Использование AUTO_INCREMENT для «divorce_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(settlement_id) связывает запись о разводе с территорией, FOREIGN KEY(human1_id) – с первым человеком, FOREIGN KEY(human2_id) – со вторым человеком.

Таблица «ArrivalGoal» описывает сущность цели прибытия. Здесь «arrival_goal_id» служит идентификатором, а «goal» хранит цель. Использование AUTO_INCREMENT для «arrival_goal_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «LeavingGoal» описывает сущность цели отбытия. Здесь «leaving_goal_id» служит идентификатором, а «goal» хранит цель. Использование AUTO_INCREMENT для «leaving_goal_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «ArrivalRate» описывает сущность записи о прибытии. Здесь «arrival_rate_id» служит идентификатором, «date» хранит дату прибытия, «from_id» хранит идентификатор территории с которой прибыл человек, «to_id» хранит идентификатор территории на которую прибыл человек, «human_id» хранит идентификатор человека, «goal_id» хранит идентификатор цели прибытия. Использование AUTO_INCREMENT для «arrival_rate_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(from_id) связывает запись о

прибытии с территорией с которой прибыл человек, FOREIGN KEY(to_id) – с территорией на которую прибыл человек, FOREIGN KEY(human_id) – с прибывшим человеком, а FOREIGN KEY(goal_id) – с целью.

Таблица «LeavingRate» описывает сущность записи об отбытии. Здесь «leaving_rate_id» служит идентификатором, «date» хранит дату отбытия, «from_id» хранит идентификатор территории с которой отбыл человек, «to_id» хранит идентификатор территории на которую отбыл человек, «human_id» хранит идентификатор человека, «goal_id» хранит идентификатор цели отбытия. Использование AUTO_INCREMENT для «leaving_rate_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(from_id) связывает запись об отбытии с территорией, с которой отбыл человек, FOREIGN KEY(to_id) – с территорией на которую отбыл человек, FOREIGN KEY(human_id) – с отбывшим человеком, а FOREIGN KEY(goal_id) – с целью.

Таблица «ConceptsAndDefinitions» описывает сущность понятия и его определения. Здесь «concept_and_definition_id» служит идентификатором, «concept» хранит понятие, а «definition» хранит определение. Использование AUTO_INCREMENT для «concept_and_definition_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «DataSources» описывает сущность источника данных. Здесь «data_source_id» служит идентификатором, «name» хранит название, а «reliability» хранит коэффициент надежности. Использование AUTO_INCREMENT для «data_source_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей.

Таблица «StatisticalIndicators» описывает сущность статистических показателей. Здесь «statistical_indicator_id» служит идентификатором, «name» хранит название, «last_update» хранит дату и время последнего обновления, «creation_date» хранит дату и время создания, «comment» хранит комментарий, а «contact_person_id» хранит идентификатор пользователя. Использование AUTO_INCREMENT для «statistical_indicator_id» обеспечивает автоматическую генерацию уникальных идентификаторов при добавлении новых записей. FOREIGN KEY(contact_person_id) связывает статистический показатель с ответственным за него пользователем. Текстовое поле «comment» оставлено nullable, что позволяет сохранять статистические показатели без комментариев, если они не нужны.

4.3 Разработка хранимых процедур

С точки зрения взаимодействия с базой данных, для сотрудников и клиентов статистического центра наиболее важными являются следующие возможности:

– добавить новую статистическую информацию;

– извлечь какие-нибудь показатели за определенный промежуток времени.

Для реализации этих наиболее важных функций базы данных статистического центра был реализован ряд хранимых процедур и функций. Далее мы опишем эти процедуры.

Процедура для добавления информации о рождении ребенка входит в категорию процедур для добавления новой статистической информации. Данная хранимая процедура принимает на вход:

– дату и время рождения ребенка;
– где родился ребенок;
– дату рождения первого родителя;
– пол первого родителя;
– национальность первого родителя;
– гражданство первого родителя;
– уровень образования первого родителя;
– ученую степень первого родителя;
– находится ли первый родитель в браке;
– дату рождения второго родителя;
– пол второго родителя;
– национальность второго родителя;
– гражданство второго родителя;
– уровень образования второго родителя;
– ученую степень второго родителя;
– находится ли второго родитель в браке.

Эта хранимая процедура создает или находит заданный тип людей для родителей, и создает запись о том, что в заданное время, в заданном месте родился ребенок с определенными родителями. В случае если указаны неизвестное место рождения, пол, национальность, гражданство, уровень образования или ученая степень, то данная процедура сообщает об ошибке.

Процедура для добавления информации о смерти человека входит в категорию процедур для добавления новой статистической информации. Данная хранимая процедура принимает на вход:

– дату и время смерти;
– населенный пункт, в котором умер человек;
– причину смерти;
– дату рождения;
– пол;
– национальность;
– гражданство;
– уровень образования;
– ученую степень;
– находится ли человек в браке;

Эта хранимая процедура создает запись о том, что в заданное время, в

заданном месте, по заданной причине умер человек с определенными выше качествами. В случае если указаны неизвестное место смерти, пол, национальность, гражданство, уровень образования или ученая степень, то данная процедура сообщает об ошибке.

Функция для получения количества рожденных детей в заданный промежуток времени на заданной территории входит в категорию функций для извлечения статистической информации.

Данная функция принимает на вход:

– дату начала интересующего промежутка, включительно;

– дату конца интересующего промежутка, исключительно;

– территорию, на которой интересен данный показатель.

Эта функция возвращает количество рожденных детей в заданный промежуток времени на заданной территории. В случае если указана неизвестная территория, то данная функция сообщает об ошибке.

4.4 Разработка триггеров

В рамках разработки базы данных для статистического центра были разработаны ряд триггеров. Данные триггеры можно разделить на несколько групп:

– исправляющие входные данные;

– проверяющие входные данные;

– поддерживающие данные в консистентном состоянии.

В первую группу входят триггеры, которые в входной строке в начале и конце удаляют пробельные символы. Они защищают пользователя от некорректного ввода. Данные триггеры реализуют полезную способность базы данных очищать входные данные от ненужных символов, тем самым поддерживая строки в одном формате.

Во вторую категорию входят триггеры проверяющие входные данные. К ним относятся:

1 Триггер, который проверяет корректность записи о рождении ребенка. А именно, что один родитель является мужчиной, а другой женщиной.

2 Триггер, который проверяет корректность записи о браке. А именно, что один супруг является мужчиной, а другой женщиной.

3 Триггер, который проверяет корректность записи о браке. А именно, что люди, которые развелись, были до этого в браке.

В третью группу входят триггеры, которые поддерживают данные в консистентном состоянии. Сюда входят триггеры, которые изменяются время обновления статистического показателя, при изменении данных, которые формируют данный статистический показатель. Это является важной функцией, так как пользователям необходимо знать, насколько свежая информация присутствует в статистическом центре.

4.5 Разработка индексов

Индекс – объект базы данных, создаваемый с целью повышения производительности поиска данных [13]. Таблицы в базе данных могут иметь большое количество строк, которые хранятся в произвольном порядке, и их поиск по заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени. Индекс формируется из значений одного или нескольких столбцов таблицы и указателей на соответствующие строки таблицы и, таким образом, позволяет искать строки, удовлетворяющие критерию поиска. Ускорение работы с использованием индексов достигается в первую очередь за счёт того, что индекс имеет структуру, оптимизированную под поиск – например, сбалансированного дерева.

Необходимость создания индексов может вызвать медленная работа запросов при выборке данных. Показать это может только тестирование на больших объёмах и показания при использовании реального приложения с реальными данными.

5 ТЕСТИРОВАНИЕ БАЗЫ ДАННЫХ

Протестируем триггеры, которые убирают пробельные символы в начале и конце строк. На рисунке 5.1 показано изначальное состояние таблицы.

	citizenship_id	name
1	1	Соединенные Штаты Америки
2	2	России
3	3	Беларуси
4	4	Китая
5	5	Германии
6	6	Великобритании

Рисунок 5.1 − Начальное состояние таблицы с гражданствами

На рисунке 5.2 продемонстрированы тестовые запросы на вставку в базу данных.

```
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (7, '  Канады  ');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (8, '  Франции');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (9, 'Японии   ');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (10, '    Австралии   ');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (11, ' Индии ');
```

Рисунок 5.2 − Запросы на вставку в таблицу с гражданствами

В результате на рисунке 5.3 мы видим, что триггеры отработали верно и убрали все пробельные символы в начале и конце строки.

	citizenship_id	name
1	1	Соединенные Штаты Америки
2	2	России
3	3	Беларуси
4	4	Китая
5	5	Германии
6	6	Великобритании
7	7	Канады
8	8	Франции
9	9	Японии
10	10	Австралии
11	11	Индии

Рисунок 5.3 − Результат вставки в таблицу с гражданствами

На рисунке 5.4 показано изначальное состояние таблицы национальностями.

	nationality_id	name
1	1	белорусы
2	2	русские
3	3	поляки

Рисунок 5.4 − Начальное состояние таблицы с национальностями

На рисунке 5.5 продемонстрированы тестовые запросы на вставку в базу данных.

```
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (4, ' украинцы');
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (5, 'евреи ');
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (6, '  другие национальности  ');
```

Рисунок 5.5 − Запросы на вставку в таблицу с национальностями

В результате на рисунке 5.6 мы видим, что триггеры отработали верно и убрали все пробельные символы в начале и конце строки.

	nationality_id	name
1	1	белорусы
2	2	русские
3	3	поляки
4	4	украинцы
5	5	евреи
6	6	другие национальности

Рисунок 5.6 − Результат вставки в таблицу с национальностями

Далее рассмотри триггеры, который изменяют дату и время обновления статистического показателя. Для этого надо вставить новую запись о рождении. Ожидаем, что у статистических показателей, которые связаны с рождением детей изменится время последнего обновления.

На рисунке 5.7 представлено стартовое состояние таблицы со статистическими показателями.

	statistical_indicator_id	name	last_update	creation_date	comment
1	1	Число родившихся	2023-12-17 18:08:58	2023-12-17 18:08:58	Методики по формированию и расчету статистических показателей приведены …
2	2	Число умерших детей до 1 года	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
3	3	Число умерших	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
4	4	Число браков	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
5	5	Число разводов	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
6	6	Число прибывших лиц	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
7	7	Число выбывших лиц	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>

Рисунок 5.7 − Начальное состояние таблицы с показателями

На рисунке 5.8 продемонстрированы тестовые запросы на вставку в базу данных.

```
INSERT INTO statisticalcenter.birthrate (birth_rate_id, date, parent1_id, parent2_id, settlement_id) VALUES (1, '2022-12-17 17:55:12', 1, 2, 11);
INSERT INTO statisticalcenter.birthrate (birth_rate_id, date, parent1_id, parent2_id, settlement_id) VALUES (2, '2022-12-16 17:56:18', 5, 6, 11);
```

Рисунок 5.8 − Запросы на вставку в таблицу с записями о рождении

В результате на рисунке 5.9 мы видим, что триггеры отработали верно и обновили дату и время последнего обновления.

	statistical_indicator_id	name	last_update	creation_date	comment
1	1	Число родившихся	2023-12-18 15:30:58	2023-12-17 18:08:58	Методики по формированию и расчету статистических показателей приведены …
2	2	Число умерших детей до 1 года	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
3	3	Число умерших	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
4	4	Число браков	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
5	5	Число разводов	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
6	6	Число прибывших лиц	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>
7	7	Число выбывших лиц	2023-12-17 18:10:55	2023-12-17 18:10:55	<null>

Рисунок 5.9 − Результат работы триггера на обновление даты изменения

ЗАКЛЮЧЕНИЕ

Работа по проектированию и разработке базы данных для статистического центра представляет собой важный этап в создании статистического центра. В соответствии с требованиями технического задания (ТЗ) и целями проекта было принято ряд ключевых решений, оформленных в виде структурированных таблиц, хранимых процедур и триггеров.

Выбор системы управления базами данных (СУБД) играет фундаментальную роль в обеспечении эффективного и надежного хранения данных. После внимательного анализа и рассмотрения различных вариантов, была выбрана MySQL. Эта СУБД предоставляет широкий функционал, является популярным решением в области реляционных баз данных и обладает открытым исходным кодом, что соответствует принципам проекта.

Модель данных была создана с учетом особенностей использования данных в статистических центрах. Таблицы были разработаны для хранения информации о пользователях, статистических показателях, графиках, территориях, организациях и характеристиках людей.

Хранимые процедуры и триггеры были реализованы с целью автоматизации часто встречающихся операций и поддержания целостности данных. Процедуры реализуют удобный и эффективный способ добавления новых данных в базу данных. Триггеры, в свою очередь, поддерживают целостность данных в базе данных.

Работа по разработке базы данных для статистического центра – это сложный, но важный этап проекта. Она обеспечивает эффективное хранение данных, обеспечивает их целостность и предоставляет удобные средства для взаимодействия с информацией. Правильно спроектированная и поддерживаемая база данных является ключевым элементом успешного функционирования статистического центра.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Что такое Росстат [Электронный ресурс]. – Режим доступа : https://rosstat.gov.ru/mission/.

[2] Нормализация баз данных [Электронный ресурс]. – Режим доступа: https://www.normalization.com/.

[3] Международная классификация болезней МКБ-10 [Электронный ресурс]. – Режим доступа : https://rosoncoweb.ru/standarts/RUSSCO/2018/2018-mkb10.pdf.

[4] Указы президента Республики Беларусь [Электронный реусур]. – Режим доступа : https://pravo.by/document/?guid=3871&p0=p30400560/.

[5] Адресация объектов недвижимости [Электронный ресурс]. – Режим доступа : https://www.vagr.by/address-kadastr/.

[6] Показатель смертности по полу [Электронный ресурс]. – Режим доступа : https://w3.unece.org/PXWeb2015/pxweb/ru/STAT/STAT__30-GE__06-Health/007_ru_GEHECOD_r.px/.

[7] Гражданство [Электронный ресурс]. – Режим доступа : http://www.eycb.coe.int/compass/ru/chapter_5/5_2.html.

[8] Должность [Электронный ресурс]. – Режим доступа : https://kodeksy-bel.com/dictionary/d/dolzhnost.htm.

[9] Право определять и указывать свою национальную принадлежность [Электронный ресурс]. – Режим доступа : https://01.xn--b1aew.xn--p1ai/document/201357.

[10] Теория статистического наблюдения [Электронный ресурс]. – Режим доступа : https://www.hse.ru/data/353/323/1234/lect4-5ec2005.pdf.

[11] Экономический и социальный совет [Электронный ресурс]. – Режим доступа : https://unece.org/sites/default/files/2023-10/rec20_rev5R.pdf.

[12] Microsoft SQL Server [Электронный ресурс]. – Режим доступа : https://www.tadviser.ru/index.php/%D0%9F%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82:Microsoft_SQL_Server.

[13] Индексы базы данных [Электронный ресурс]. – Режим доступа: https://www.tutorialspoint.com/postgresql/postgresql_indexes.htm.

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг кода

```
DROP DATABASE StatisticalCenter;

CREATE DATABASE StatisticalCenter;
use StatisticalCenter;

-- Должность User
CREATE TABLE Positions
(
    position_id INTEGER      NOT NULL AUTO_INCREMENT,
    name        VARCHAR(100) NOT NULL,
    PRIMARY KEY (position_id)
);

-- Отдел, в котором работает user
CREATE TABLE Departments
(
    department_id INTEGER      NOT NULL AUTO_INCREMENT,
    name          VARCHAR(200) NOT NULL,
    PRIMARY KEY (department_id)
);

CREATE TABLE IF NOT EXISTS Users
(
    user_id            INTEGER     NOT NULL AUTO_INCREMENT,
    name               VARCHAR(45) NOT NULL,
    surname            VARCHAR(45) NOT NULL,
    patronymic         VARCHAR(45),
    phone_num          VARCHAR(13),
    city_telephone_num VARCHAR(7),
    email              VARCHAR(45),
    position_id        INTEGER,
    department_id      INTEGER,
    PRIMARY KEY (user_id),
    FOREIGN KEY (position_id) REFERENCES Positions (position_id),
    FOREIGN KEY (department_id) REFERENCES Departments (department_id)
);

CREATE TABLE IF NOT EXISTS UnitsOfMeasure
(
    unit_of_measure_id INTEGER     NOT NULL AUTO_INCREMENT,
    name               VARCHAR(45) NOT NULL,
    description        VARCHAR(100),
    PRIMARY KEY (unit_of_measure_id)
);

CREATE TABLE IF NOT EXISTS TimeIntervals
(
    time_interval_id    INTEGER     NOT NULL AUTO_INCREMENT,
    name                VARCHAR(45) NOT NULL,
    interval_in_seconds INTEGER,
    PRIMARY KEY (time_interval_id)
);

-- Графики
CREATE TABLE IF NOT EXISTS Charts
(
```

```
    chart_id    INTEGER     NOT NULL AUTO_INCREMENT,
    name        VARCHAR(45) NOT NULL,
    description VARCHAR(300),
    PRIMARY KEY (chart_id)
);

-- Тип населенного пункта(а.г., город, район, область и т.д.)
CREATE TABLE IF NOT EXISTS SettlementsType
(
    settlement_type_id INTEGER     NOT NULL AUTO_INCREMENT,
    name               VARCHAR(50) NOT NULL,
    short_name         VARCHAR(10),
    PRIMARY KEY (settlement_type_id)
);

-- Населенные пунты
CREATE TABLE IF NOT EXISTS Settlements
(
    settlement_id       INTEGER      NOT NULL AUTO_INCREMENT,
    settlements_type_id INTEGER      NOT NULL,
    parent_id           INTEGER DEFAULT NULL,
    name                VARCHAR(100) NOT NULL,
    PRIMARY KEY (settlement_id),
    FOREIGN KEY (settlements_type_id) REFERENCES SettlementsType
(settlement_type_id),
    FOREIGN KEY (parent_id) REFERENCES Settlements (settlement_id)
);


CREATE TABLE IF NOT EXISTS Addresses
(
    address_id    INTEGER NOT NULL AUTO_INCREMENT,
    settlement_id INTEGER NOT NULL,
    street        VARCHAR(100),
    house_num     VARCHAR(10),
    PRIMARY KEY (address_id),
    FOREIGN KEY (settlement_id) REFERENCES Settlements (settlement_id)
);

CREATE TABLE IF NOT EXISTS Organizations
(
    organization_id    INTEGER      NOT NULL AUTO_INCREMENT,
    address_id         INTEGER      NOT NULL,
    name               VARCHAR(100) NOT NULL,
    short_name         VARCHAR(20),
    phone_num          VARCHAR(13),
    city_telephone_num VARCHAR(7),
    email              VARCHAR(45),
    PRIMARY KEY (organization_id),
    FOREIGN KEY (address_id) REFERENCES Addresses (address_id)
);

-- Национальности
CREATE TABLE IF NOT EXISTS Nationalities
(
    nationality_id INTEGER NOT NULL AUTO_INCREMENT,
    name           VARCHAR(100) NOT NULL,
    PRIMARY KEY (nationality_id)
);

-- Гражданства
CREATE TABLE IF NOT EXISTS Citizenships
```

```
(
    citizenship_id INTEGER NOT NULL AUTO_INCREMENT,
    name           VARCHAR(100) NOT NULL,
    PRIMARY KEY (citizenship_id)
);

CREATE TABLE IF NOT EXISTS EducationLevels
(
    education_level_id INTEGER NOT NULL AUTO_INCREMENT,
    name               VARCHAR(100) NOT NULL,
    PRIMARY KEY (education_level_id)
);

CREATE TABLE IF NOT EXISTS AcademicDegrees
(
    academic_degree_id INTEGER NOT NULL AUTO_INCREMENT,
    name               VARCHAR(100) NOT NULL,
    PRIMARY KEY (academic_degree_id)
);

CREATE TABLE IF NOT EXISTS Humans
(
    human_id           INTEGER                 NOT NULL AUTO_INCREMENT,
    birthday           DATE    DEFAULT NULL,
    sex                ENUM ('Male', 'Female') NOT NULL,
    nationality_id     INTEGER DEFAULT NULL,
    citizenship_id     INTEGER DEFAULT NULL,
    education_level_id INTEGER DEFAULT NULL,
    academic_degree_id INTEGER DEFAULT NULL,
    is_married         BIT(1)  DEFAULT 0,

    PRIMARY KEY (human_id),
    FOREIGN KEY (nationality_id) REFERENCES Nationalities (nationality_id),
    FOREIGN KEY (citizenship_id) REFERENCES Citizenships (citizenship_id),
    FOREIGN KEY (education_level_id) REFERENCES EducationLevels
(education_level_id),
    FOREIGN KEY (academic_degree_id) REFERENCES AcademicDegrees
(academic_degree_id)
);

CREATE TABLE IF NOT EXISTS BirthRate
(
    birth_rate_id INTEGER  NOT NULL AUTO_INCREMENT,
    date          DATETIME NOT NULL,
    settlement_id INTEGER  NOT NULL,
    parent1_id    INTEGER,
    parent2_id    INTEGER,
    PRIMARY KEY (birth_rate_id),
    FOREIGN KEY (settlement_id) REFERENCES Settlements (settlement_id),
    FOREIGN KEY (parent1_id) REFERENCES Humans (human_id),
    FOREIGN KEY (parent2_id) REFERENCES Humans (human_id)
);

CREATE TABLE IF NOT EXISTS CausesOfDeath
(
    cause_of_death_id INTEGER NOT NULL AUTO_INCREMENT,
    name              VARCHAR(100),
    code_MKB10        VARCHAR(30),
    PRIMARY KEY (cause_of_death_id)
);

CREATE TABLE IF NOT EXISTS MortalityRate
```

```
(
    mortality_rate_id INTEGER  NOT NULL AUTO_INCREMENT,
    human_id          INTEGER  NOT NULL,
    cause_of_death_id INTEGER  NOT NULL,
    settlement_id     INTEGER  NOT NULL,
    date              DATETIME NOT NULL,
    PRIMARY KEY (mortality_rate_id),
    FOREIGN KEY (human_id) REFERENCES Humans (human_id),
    FOREIGN KEY (cause_of_death_id) REFERENCES CausesOfDeath
(cause_of_death_id),
    FOREIGN KEY (settlement_id) REFERENCES Settlements (settlement_id)
);

-- Браки
CREATE TABLE IF NOT EXISTS Marriages
(
    marriage_id   INTEGER NOT NULL AUTO_INCREMENT,
    human1_id     INTEGER NOT NULL,
    human2_id     INTEGER NOT NULL,
    settlement_id INTEGER NOT NULL,
    date          DATE    NOT NULL,
    PRIMARY KEY (marriage_id),
    FOREIGN KEY (human1_id) REFERENCES Humans (human_id),
    FOREIGN KEY (human2_id) REFERENCES Humans (human_id),
    FOREIGN KEY (settlement_id) REFERENCES Settlements (settlement_id)
);

-- Разводы
CREATE TABLE IF NOT EXISTS Divorces
(
    divorce_id     INTEGER NOT NULL AUTO_INCREMENT,
    human1_id      INTEGER NOT NULL,
    human2_id      INTEGER NOT NULL,
    settlement_id  INTEGER NOT NULL,
    date           DATE    NOT NULL,
    children_count INTEGER DEFAULT 0, -- TODO: Maybe delete
    PRIMARY KEY (divorce_id),
    FOREIGN KEY (human1_id) REFERENCES Humans (human_id),
    FOREIGN KEY (human2_id) REFERENCES Humans (human_id),
    FOREIGN KEY (settlement_id) REFERENCES Settlements (settlement_id)
);

CREATE TABLE IF NOT EXISTS ArrivalGoal
(
    arrival_goal_id INTEGER NOT NULL AUTO_INCREMENT,
    goal            VARCHAR(100) NOT NULL,
    PRIMARY KEY (arrival_goal_id)
);

CREATE TABLE IF NOT EXISTS ArrivalRate
(
    arrival_rate_id INTEGER NOT NULL AUTO_INCREMENT,
    human_id        INTEGER NOT NULL,
    date            DATE    NOT NULL,
    from_id         INTEGER NOT NULL,
    to_id           INTEGER NOT NULL,
    goal_id         INTEGER NOT NULL,

    PRIMARY KEY (arrival_rate_id),
    FOREIGN KEY (human_id) REFERENCES Humans (human_id),
    FOREIGN KEY (from_id) REFERENCES Settlements (settlement_id),
    FOREIGN KEY (to_id) REFERENCES Settlements (settlement_id),
```

```
    FOREIGN KEY (goal_id) REFERENCES ArrivalGoal (arrival_goal_id)
);

CREATE TABLE IF NOT EXISTS LeavingGoal
(
    leaving_goal_id INTEGER NOT NULL AUTO_INCREMENT,
    goal            VARCHAR(100) NOT NULL,
    PRIMARY KEY (leaving_goal_id)
);

CREATE TABLE IF NOT EXISTS LeavingRate
(
    leaving_rate_id INTEGER NOT NULL AUTO_INCREMENT,
    human_id        INTEGER NOT NULL,
    date            DATE    NOT NULL,
    from_id         INTEGER NOT NULL,
    to_id           INTEGER NOT NULL,
    goal_id         INTEGER NOT NULL,
    PRIMARY KEY (leaving_rate_id),
    FOREIGN KEY (human_id) REFERENCES Humans (human_id),
    FOREIGN KEY (from_id) REFERENCES Settlements (settlement_id),
    FOREIGN KEY (to_id) REFERENCES Settlements (settlement_id),
    FOREIGN KEY (goal_id) REFERENCES LeavingGoal (leaving_goal_id)
);

CREATE TABLE IF NOT EXISTS ConceptsAndDefinitions
(
    concept_and_definition_id INTEGER NOT NULL AUTO_INCREMENT,
    concept                   VARCHAR(100) NOT NULL,
    definition                VARCHAR(1000) NOT NULL,
    PRIMARY KEY (concept_and_definition_id)
);

CREATE TABLE IF NOT EXISTS DataSources
(
    data_source_id INTEGER NOT NULL AUTO_INCREMENT,
    name           VARCHAR(300) NOT NULL,
    reliability    DOUBLE  NOT NULL DEFAULT 1,
    PRIMARY KEY (data_source_id),
    CONSTRAINT DataSources_reliability_chk CHECK (reliability BETWEEN 0 AND
1)
);

CREATE TABLE IF NOT EXISTS StatisticalIndicators
(
    statistical_indicator_id INTEGER  NOT NULL AUTO_INCREMENT,
    name                     VARCHAR(100) NOT NULL,
    -- TODO: Тут можно ввести тригеры на обновление last_update
    last_update              DATETIME NOT NULL DEFAULT NOW(),
    creation_date            DATETIME NOT NULL DEFAULT NOW(),
    comment                  VARCHAR(1000),
    contact_person_id        INTEGER  NOT NULL,
    PRIMARY KEY (statistical_indicator_id),
    FOREIGN KEY (contact_person_id) REFERENCES Users (user_id)
);

CREATE TABLE IF NOT EXISTS TimeIntervals_StatisticalIndicators
(
    time_interval_id         INTEGER NOT NULL,
    statistical_indicator_id INTEGER NOT NULL,
    PRIMARY KEY (time_interval_id, statistical_indicator_id),
    FOREIGN KEY (time_interval_id) REFERENCES TimeIntervals
```

```
(time_interval_id),
    FOREIGN KEY (statistical_indicator_id) REFERENCES StatisticalIndicators
(statistical_indicator_id)
);

CREATE TABLE IF NOT EXISTS UnitsOfMeasure_StatisticalIndicators
(
    unit_of_measure_id       INTEGER NOT NULL,
    statistical_indicator_id INTEGER NOT NULL,
    PRIMARY KEY (unit_of_measure_id, statistical_indicator_id),
    FOREIGN KEY (unit_of_measure_id) REFERENCES UnitsOfMeasure
(unit_of_measure_id),
    FOREIGN KEY (statistical_indicator_id) REFERENCES StatisticalIndicators
(statistical_indicator_id)
);

CREATE TABLE IF NOT EXISTS ConceptsAndDefinitions_StatisticalIndicators
(
    concept_and_definition_id INTEGER NOT NULL,
    statistical_indicator_id  INTEGER NOT NULL,
    PRIMARY KEY (concept_and_definition_id, statistical_indicator_id),
    FOREIGN KEY (concept_and_definition_id) REFERENCES ConceptsAndDefinitions
(concept_and_definition_id),
    FOREIGN KEY (statistical_indicator_id) REFERENCES StatisticalIndicators
(statistical_indicator_id)
);

CREATE TABLE IF NOT EXISTS DataSources_StatisticalIndicators
(
    data_source_id           INTEGER NOT NULL,
    statistical_indicator_id INTEGER NOT NULL,
    PRIMARY KEY (data_source_id, statistical_indicator_id),
    FOREIGN KEY (data_source_id) REFERENCES DataSources (data_source_id),
    FOREIGN KEY (statistical_indicator_id) REFERENCES StatisticalIndicators
(statistical_indicator_id)
);

CREATE VIEW HumansInfo AS
SELECT H.human_id,
       H.birthday,
       H.sex,
       N.name  AS nationality,
       C.name  AS citizenship,
       EL.name AS education_level,
       AD.name AS academic_degree,
       H.is_married
FROM Humans H
         LEFT JOIN Nationalities N on N.nationality_id = H.nationality_id
         LEFT JOIN Citizenships C on H.citizenship_id = C.citizenship_id
         LEFT JOIN EducationLevels EL on H.education_level_id =
EL.education_level_id
         LEFT JOIN AcademicDegrees AD on H.academic_degree_id =
AD.academic_degree_id;

show tables;

-- Create AcademicDegrees
INSERT INTO statisticalcenter.academicdegrees (academic_degree_id, name)
VALUES (1, 'Магистр');
INSERT INTO statisticalcenter.academicdegrees (academic_degree_id, name)
VALUES (2, 'Кандидат наук');
```

```
INSERT INTO statisticalcenter.academicdegrees (academic_degree_id, name)
VALUES (3, 'Доктор наук');
INSERT INTO statisticalcenter.academicdegrees (academic_degree_id, name)
VALUES (4, 'Бакалавр');

-- Create ArrivalGoal
INSERT INTO statisticalcenter.arrivalgoal (arrival_goal_id, goal) VALUES (1,
'Учеба');
INSERT INTO statisticalcenter.arrivalgoal (arrival_goal_id, goal) VALUES (2,
'Работа');
INSERT INTO statisticalcenter.arrivalgoal (arrival_goal_id, goal) VALUES (3,
'Семейные обстоятельства');

-- CreateLeavingGoal
INSERT INTO statisticalcenter.leavinggoal (leaving_goal_id, goal) VALUES (1,
'Учеба');
INSERT INTO statisticalcenter.leavinggoal (leaving_goal_id, goal) VALUES (2,
'Работа');
INSERT INTO statisticalcenter.leavinggoal (leaving_goal_id, goal) VALUES (3,
'Семейные обстоятельства');


-- Create Charts
INSERT INTO statisticalcenter.charts (chart_id, name, description) VALUES (1,
'Линейный график', 'Линейный график представляет собой линию, соединяющую
точки данных на плоскости. Используется для отображения изменения значений
величины во времени или при изменении другого параметра.');
INSERT INTO statisticalcenter.charts (chart_id, name, description) VALUES (2,
'Столбчатая диаграмма', 'Столбчатая диаграмма использует прямоугольные
столбцы для представления данных. Чаще всего используется для сравнения
значений различных категорий или отслеживания изменений величин.');
INSERT INTO statisticalcenter.charts (chart_id, name, description) VALUES (3,
'Круговая диаграмма', 'Круговая диаграмма представляет собой круг,
разделенный на секторы, пропорциональные относительным размерам частей
целого. Подходит для отображения долей в общей сумме.');
INSERT INTO statisticalcenter.charts (chart_id, name, description) VALUES (4,
'Гистограмма', ' Гистограмма используется для представления распределения
данных по интервалам или группам. Каждый столбец представляет собой частоту
встречаемости значений в определенном диапазоне.');
INSERT INTO statisticalcenter.charts (chart_id, name, description) VALUES (5,
'Точечная диаграмма', 'Точечная диаграмма представляет собой набор точек,
каждая из которых представляет сочетание значений двух переменных.
Используется для выявления взаимосвязи или распределения данных.');

-- Create Citizenships
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (1,
'Соединенные Штаты Америки');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (2,
'России');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (3,
'Беларуси');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (4,
'Китая');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (5,
'Германии');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (6,
'Великобритании');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (7,
'Канады');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (8,
'Франции');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (9,
```

```
'Японии');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (10,
'Австралии');
INSERT INTO statisticalcenter.citizenships (citizenship_id, name) VALUES (11,
'Индии');

-- Create Natinalities
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (1,
'белорусы');
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (2,
'русские');
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (3,
'поляки');
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (4,
'украинцы');
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (5,
'евреи');
INSERT INTO statisticalcenter.nationalities (nationality_id, name) VALUES (6,
'другие национальности');


-- Create DataSources
INSERT INTO statisticalcenter.datasources (data_source_id, name, reliability)
VALUES (1, 'Выборочное обследование домашних хозяйств в целях изучения
проблем занятости населения', 1);
INSERT INTO statisticalcenter.datasources (data_source_id, name, reliability)
VALUES (2, 'Тестовый источник данных', 1);

-- Create CouseOfDeath
INSERT INTO statisticalcenter.causesofdeath (cause_of_death_id, name,
code_MKB10) VALUES (1, 'Злокачественные новообразования пищевода ', 'C15');
INSERT INTO statisticalcenter.causesofdeath (cause_of_death_id, name,
code_MKB10) VALUES (2, 'Дифтерия ', 'A36');
INSERT INTO statisticalcenter.causesofdeath (cause_of_death_id, name,
code_MKB10) VALUES (3, 'Другие формы столбняка', 'A35 ');
INSERT INTO statisticalcenter.causesofdeath (cause_of_death_id, name,
code_MKB10) VALUES (4, 'Коклюш', 'A37');
INSERT INTO statisticalcenter.causesofdeath (cause_of_death_id, name,
code_MKB10) VALUES (5, 'Скарлатина', 'A38');
INSERT INTO statisticalcenter.causesofdeath (cause_of_death_id, name,
code_MKB10) VALUES (6, 'Менингококковая инфекция', 'A39');

-- Create Departments
INSERT INTO statisticalcenter.departments (department_id, name) VALUES (1,
'управление статистики занятости населения Главного управления статистики
труда');
INSERT INTO statisticalcenter.departments (department_id, name) VALUES (2,
'Тестовый отдел');

-- Create EducationLevel
INSERT INTO statisticalcenter.educationlevels (education_level_id, name)
VALUES (1, 'Дошкольное');
INSERT INTO statisticalcenter.educationlevels (education_level_id, name)
VALUES (2, 'Общее базовое');
INSERT INTO statisticalcenter.educationlevels (education_level_id, name)
VALUES (3, 'Общее среднее');
INSERT INTO statisticalcenter.educationlevels (education_level_id, name)
VALUES (4, 'Профессионально-техническое');
INSERT INTO statisticalcenter.educationlevels (education_level_id, name)
VALUES (5, 'Среднее специальное');
INSERT INTO statisticalcenter.educationlevels (education_level_id, name)
VALUES (6, 'Высшее профессиональное');
```

```
-- Create SettlementsType
INSERT INTO statisticalcenter.settlementstype (settlement_type_id, name,
short_name) VALUES (1, 'город', 'г.');
INSERT INTO statisticalcenter.settlementstype (settlement_type_id, name,
short_name) VALUES (2, 'планета', null);
INSERT INTO statisticalcenter.settlementstype (settlement_type_id, name,
short_name) VALUES (3, 'страна', null);
INSERT INTO statisticalcenter.settlementstype (settlement_type_id, name,
short_name) VALUES (4, 'область', 'обл.');
INSERT INTO statisticalcenter.settlementstype (settlement_type_id, name,
short_name) VALUES (5, 'район', 'р-н.');
INSERT INTO statisticalcenter.settlementstype (settlement_type_id, name,
short_name) VALUES (6, 'агрогородок', 'аг.');
INSERT INTO statisticalcenter.settlementstype (settlement_type_id, name,
short_name) VALUES (7, 'деревня', 'д.');

-- Create Settlements
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (1, 2, null, 'Земля');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (2, 3, 1, 'Республика
Беларусь');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (3, 1, 2, 'Минск');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (4, 4, 2, 'Гродненская');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (5, 4, 2, 'Брестская');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (6, 4, 2, 'Витебская');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (7, 4, 2, 'Гомельская');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (8, 4, 2, 'Минская');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (9, 4, 2, 'Могилевская');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (10, 5, 4, 'Щучинский');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (11, 1, 10, 'Щучин');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (12, 6, 10, 'Рожанка');
INSERT INTO statisticalcenter.settlements (settlement_id,
settlements_type_id, parent_id, name) VALUES (13, 7, 10, 'Потока');

-- Create unitsOfMeasure
INSERT INTO statisticalcenter.unitsofmeasure (unit_of_measure_id, name,
description) VALUES (1, 'проценты', '%');
INSERT INTO statisticalcenter.unitsofmeasure (unit_of_measure_id, name,
description) VALUES (2, 'человек', null);
INSERT INTO statisticalcenter.unitsofmeasure (unit_of_measure_id, name,
description) VALUES (3, 'единиц', null);

-- Create Addresses
INSERT INTO statisticalcenter.addresses (address_id, settlement_id, street,
house_num) VALUES (1, 3, 'Якуба Коласа', '28');
INSERT INTO statisticalcenter.addresses (address_id, settlement_id, street,
house_num) VALUES (2, 11, 'Строителей', '8');

-- Create Organizations
INSERT INTO statisticalcenter.organizations (organization_id, address_id,
```

```
name, short_name, phone_num, city_telephone_num, email) VALUES (1, 1,
'Статистический центр', null, '+375295941978', null, 'stat@gmail.com');
INSERT INTO statisticalcenter.organizations (organization_id, address_id,
name, short_name, phone_num, city_telephone_num, email) VALUES (2, 2,
'Щучинский статистический центр', null, '+375752517856', null,
'shstat@gmail.com');

-- Create TimeIntervals
INSERT INTO statisticalcenter.timeintervals (time_interval_id, name,
interval_in_seconds) VALUES (1, 'месяц', 2592000);
INSERT INTO statisticalcenter.timeintervals (time_interval_id, name,
interval_in_seconds) VALUES (2, 'квартал', 7862400);
INSERT INTO statisticalcenter.timeintervals (time_interval_id, name,
interval_in_seconds) VALUES (3, 'год', 31536000);

-- Create Humans
INSERT INTO statisticalcenter.humans (human_id, birthday, sex,
nationality_id, citizenship_id, education_level_id, academic_degree_id,
is_married) VALUES (1, '2000-12-14', 'Male', 1, 1, 6, 4, true);
INSERT INTO statisticalcenter.humans (human_id, birthday, sex,
nationality_id, citizenship_id, education_level_id, academic_degree_id,
is_married) VALUES (2, '2021-06-02', 'Female', 1, 1, 6, 4, true);
INSERT INTO statisticalcenter.humans (human_id, birthday, sex,
nationality_id, citizenship_id, education_level_id, academic_degree_id,
is_married) VALUES (3, '1976-12-18', 'Male', 2, 2, 5, null, false);
INSERT INTO statisticalcenter.humans (human_id, birthday, sex,
nationality_id, citizenship_id, education_level_id, academic_degree_id,
is_married) VALUES (4, '1992-06-17', 'Female', 1, 1, 4, null, false);
INSERT INTO statisticalcenter.humans (human_id, birthday, sex,
nationality_id, citizenship_id, education_level_id, academic_degree_id,
is_married) VALUES (5, '1991-12-13', 'Male', 2, 2, 3, null, true);
INSERT INTO statisticalcenter.humans (human_id, birthday, sex,
nationality_id, citizenship_id, education_level_id, academic_degree_id,
is_married) VALUES (6, '1987-10-17', 'Female', 3, 3, 2, null, true);

-- Create ArrivalRate
INSERT INTO statisticalcenter.arrivalrate (arrival_rate_id, human_id, date,
from_id, to_id, goal_id) VALUES (1, 6, '2022-12-02', 3, 11, 1);
INSERT INTO statisticalcenter.arrivalrate (arrival_rate_id, human_id, date,
from_id, to_id, goal_id) VALUES (2, 5, '2022-12-03', 3, 11, 2);

-- Create Birthrate
INSERT INTO statisticalcenter.birthrate (birth_rate_id, date, parent1_id,
parent2_id, settlement_id) VALUES (1, '2022-12-17 17:55:12', 1, 2, 11);
INSERT INTO statisticalcenter.birthrate (birth_rate_id, date, parent1_id,
parent2_id, settlement_id) VALUES (2, '2022-12-16 17:56:18', 5, 6, 11);

-- Create divorces
INSERT INTO statisticalcenter.divorces (divorce_id, human1_id, human2_id,
date, children_count, settlement_id) VALUES (1, 1, 2, '2023-12-16', 1, 11);
INSERT INTO statisticalcenter.divorces (divorce_id, human1_id, human2_id,
date, children_count, settlement_id) VALUES (2, 5, 6, '2023-12-15', 1, 11);

-- Create mortalityRate
INSERT INTO statisticalcenter.mortalityrate (mortality_rate_id, human_id,
cause_of_death_id, date, settlement_id) VALUES (1, 3, 1, '2022-07-22
18:02:03', 11);
INSERT INTO statisticalcenter.mortalityrate (mortality_rate_id, human_id,
cause_of_death_id, date, settlement_id) VALUES (2, 4, 2, '2022-07-17
18:02:34', 11);

-- Create Positions
```

```
INSERT INTO statisticalcenter.positions (position_id, name) VALUES (1,
'Начальник отдела');
INSERT INTO statisticalcenter.positions (position_id, name) VALUES (2,
'Специалист');
INSERT INTO statisticalcenter.positions (position_id, name) VALUES (3,
'Консультант');

-- Create Users
INSERT INTO statisticalcenter.users (user_id, name, surname, patronymic,
phone_num, city_telephone_num, email, position_id, department_id) VALUES (1,
'Александр', 'Александров', 'Александрович', '+375295752319', null,
'alex@gmail.com', 3, 1);
INSERT INTO statisticalcenter.users (user_id, name, surname, patronymic,
phone_num, city_telephone_num, email, position_id, department_id) VALUES (2,
'Татьяна', 'Савченко', 'Ивановна', '+375173785839', null,
'savchenko@gmail.com', 2, 2);

-- Create StatisticalIndicators
INSERT INTO statisticalcenter.statisticalindicators
(statistical_indicator_id, name, last_update, creation_date, comment,
contact_person_id) VALUES (1, 'Число родившихся', '2023-12-17 18:08:58',
'2023-12-17 18:08:58', 'Методики по формированию и расчету статистических
показателей приведены на официальном Интернет-сайте', 1);
INSERT INTO statisticalcenter.statisticalindicators
(statistical_indicator_id, name, last_update, creation_date, comment,
contact_person_id) VALUES (2, 'Число умерших детей до 1 года', '2023-12-17
18:10:55', '2023-12-17 18:10:55', null, 1);
INSERT INTO statisticalcenter.statisticalindicators
(statistical_indicator_id, name, last_update, creation_date, comment,
contact_person_id) VALUES (3, 'Число умерших', '2023-12-17 18:10:55', '2023-
12-17 18:10:55', null, 1);
INSERT INTO statisticalcenter.statisticalindicators
(statistical_indicator_id, name, last_update, creation_date, comment,
contact_person_id) VALUES (4, 'Число браков', '2023-12-17 18:10:55', '2023-
12-17 18:10:55', null, 1);
INSERT INTO statisticalcenter.statisticalindicators
(statistical_indicator_id, name, last_update, creation_date, comment,
contact_person_id) VALUES (5, 'Число разводов', '2023-12-17 18:10:55', '2023-
12-17 18:10:55', null, 2);
INSERT INTO statisticalcenter.statisticalindicators
(statistical_indicator_id, name, last_update, creation_date, comment,
contact_person_id) VALUES (6, 'Число прибывших лиц', '2023-12-17 18:10:55',
'2023-12-17 18:10:55', null, 2);
INSERT INTO statisticalcenter.statisticalindicators
(statistical_indicator_id, name, last_update, creation_date, comment,
contact_person_id) VALUES (7, 'Число выбывших лиц', '2023-12-17 18:10:55',
'2023-12-17 18:10:55', null, 2);
```

ПРИЛОЖЕНИЕ Б
(обязательное)
Схема базы данных

ПРИЛОЖЕНИЕ В
(ОБЯЗАТЕЛЬНОЕ)
ВЕДОМОСТЬ