**Модель базы данных** - тип модели данных, которая определяет логическую структуру базы данных и в корне определяет, каким образом данные могут храниться, организовываться и обрабатываться. Самым популярным примером модели базы данных является [реляционная модель](https://ru.bmstu.wiki/RDBMS_(Relational_Database_Management_System)), которая использует табличный формат.

**Модель данных** — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Эти объекты позволяют моделировать структуру данных, а операторы — поведение данных[[1]](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85#cite_note-Date-1).

**Каждая**[**БД**](https://ru.wikipedia.org/wiki/%D0%91%D0%94)**и**[**СУБД**](https://ru.wikipedia.org/wiki/%D0%A1%D0%A3%D0%91%D0%94) строится на основе некоторой явной или неявной модели данных. Все СУБД, построенные на одной и той же модели данных, относят к одному типу. Например, основой реляционных СУБД является [реляционная модель данных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), сетевых СУБД — [сетевая модель данных](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), иерархических СУБД — [иерархическая модель данных](https://ru.wikipedia.org/wiki/%D0%98%D0%B5%D1%80%D0%B0%D1%80%D1%85%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

**Общие логические модели данных для баз данных:**

[Иерархическая модель данных](https://ru.bmstu.wiki/%D0%98%D0%B5%D1%80%D0%B0%D1%80%D1%85%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85); [Сетевая модель](https://ru.bmstu.wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85); Реляционная модель; [ER-модель](https://ru.bmstu.wiki/index.php?title=ER-%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C&action=edit&redlink=1); Расширенная модель сущностных отношений; [Объектная модель](https://ru.bmstu.wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85); Документоориентированная модель; Модель объекта-атрибута; Схема звезды; Объектно-реляционная база данных объединяет две связанные структуры.

**Модели физических данных включают:**

Инвертированный индекс;[Табличная модель](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0)Плоский файл.

**Другие модели:**

Ассоциативная модель; Корреляционная модель; Многомерная модель; Многозначная модель; Семантическая модель; База данных [XML](https://ru.bmstu.wiki/index.php?title=XML&action=edit&redlink=1); Именованный граф; [Triplestore](https://ru.bmstu.wiki/index.php?title=Triplestore&action=edit&redlink=1).

**Три основные модели базы данных** – это иерархическая, сетевая и реляционная. Эти модели отличаются между собой по способу установления связей между данными.

**Иерархический подход к организации баз данных.** Иерархические базы данных имеют форму деревьев с дугами-связями и узлами-элементами данных. предполагала неравноправие между данными – одни подчинены другим.

**Сетевая модель данных.** В сетевых БД наряду с вертикальными реализованы и горизонтальные связи. Однако унаследованы многие недостатки иерархической и главный из них, необходимость четко определять на физическом уровне связи данных и столь же четко следовать этой структуре связей при запросах к базе.

**Реляционная модель данных.**Реляционная модель появилась вследствие стремления сделать базу данных как можно более гибкой. Данная модель предоставила простой и эффективный механизм поддержания связей данных.

**Объектно-ориентированная модель.**  способности хранить и обрабатывать новые объекты – текст, аудио- и видеоинформацию, а также документы. Основные трудности объектно-ориентированного моделирования данных проистекают из того, что такого развитого математического аппарата, на который могла бы опираться общая объектно-ориентированная модель данных, не существует. В большой степени, поэтому до сих пор нет базовой объектно-ориентированной модели. Несмотря на преимущества объектно-ориентированных систем – реализация сложных типов данных, связь с языками программирования и т.п. – на ближайшее время превосходство реляционных [СУБД](https://ru.bmstu.wiki/%D0%A1%D0%A3%D0%91%D0%94) гарантировано.

Во-первых, все данные в модели представляются в виде таблиц и только таблиц**. Реляционная** модель – **единственная из всех обеспечивает единообразие представления данных**. И сущности, и связи этих самых сущностей представляются в модели совершенно одинаково – таблицами.

Избежать трудностей манипулирования позволяет второй элемент модели – **реляционно-полный язык** (отметим, что язык является неотъемлемой частью любой модели данных, без него модель не существует). Полнота языка в приложении к реляционной модели означает, что он должен выполнять любую операцию реляционной алгебры или реляционного исчисления. Более того, язык должен описывать любой запрос в виде операций с таблицами, а не с их строками. Одним из таких языков является [SQL](https://ru.bmstu.wiki/SQL).

Третий элемент реляционной модели требует от реляционной модели **поддержания некоторых ограничений целостности.** Одно из таких ограничений утверждает, что каждая строка в таблице должна иметь некий уникальный идентификатор, называемый первичным ключом. Второе ограничение накладывается на целостность ссылок между таблицами. Оно утверждает, что атрибуты таблицы, ссылающиеся на первичные ключи других таблиц, должны иметь одно из значений этих первичных ключей.

**Объектно-реляционные субд**

Разница между объектно-реляционными и объектными СУБД: первые являют собой надстройку над реляционной схемой, вторые же изначально объектно-ориентированы. Главная особенность и отличие объектно-реляционных, как и объектных, СУБД от реляционных заключается в том, что О(Р)СУБД интегрированы с Объектно-Ориентированным (OO) языком программирования, внутренним или внешним как C++, Java. Характерные свойства OРСУБД:

комплексные данные,

наследование типа,

объектное поведение.

Комплексные данные могут быть реализованы через постоянно-хранимые объекты (persistent objects). Создание комплексных данных в большинстве существующих ОРСУБД основано на предварительном определении схемы через определяемый пользователем тип ([UDT](https://ru.bmstu.wiki/UDP_(User_Datagram_Protocol)) - user-defined type). Используются также встроенные конструкторы составных типов, например массив (ARRAY).

Иерархия структурных комплексных данных предлагает дополнительное свойство, наследование типа. То есть структурный тип может иметь подтипы, которые используют все его атрибуты и содержат дополнительные атрибуты, специфицированные в подтипе.

Объектное поведение закладывается через описание программных объектов. Такие объекты должны быть сохраняемыми и переносимыми для обработки в базе данных, поэтому они называются обычно как постоянные объекты. Внутри базы данных все отношения с постоянным программным объектом есть отношения с его объектным идентификатором (OID).

[Реляционная модель](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) введена в 1970 году как способ сделать системы управления базами данных более независимыми от какого-либо конкретного приложения.

Три ключевых термина широко используются в моделях реляционных баз данных: отношения, атрибуты и домены. Отношение - это таблица со столбцами и строками. Именованные столбцы отношения называются атрибутами, а домен - это набор значений, которые могут принимать атрибуты.

Структурой данных реляционной модели является таблица, в которой информация о конкретном объекте представлена в строках и столбцах. Таким образом, «отношение» в «реляционной базе данных» относится к различным таблицам в базе данных; Отношение представляет собой набор кортежей. В столбцах перечислены различные атрибуты объекта (например, имя сотрудника, адрес или номер телефона), а строка является фактическим экземпляром объекта (конкретного сотрудника), который представлен отношением. В результате каждый кортеж таблицы employee представляет различные атрибуты одного сотрудника.

Все отношения в реляционной базе данных должны придерживаться некоторых основных правил, которые можно квалифицировать как отношения. Во-первых, упорядочение столбцов несущественно в таблице. Во-вторых, не может быть одинаковых кортежей или строк в таблице. И, в-третьих, каждый кортеж будет содержать одно значение для каждого из его атрибутов.

Реляционная база данных содержит несколько таблиц, каждая из которых похожа на таковую в «плоской» модели базы данных. Одна из сильных сторон реляционной модели заключается в том, что в принципе любое значение, имеющее место в двух разных записях (принадлежащих к одной таблице или к разным таблицам), подразумевает взаимосвязь между этими двумя записями. Тем не менее, для обеспечения явных ограничений целостности отношения между записями в таблицах также могут быть определены явно, путем идентификации отношений родитель-ребенок, характеризующихся присвоением мощности (1: 1, (0) 1: M, M: M ). Таблицы также могут иметь назначенный единственный атрибут или набор атрибутов, которые могут действовать как «ключ», которые могут использоваться для однозначной идентификации каждого кортежа в таблице.

Ключ, который может использоваться для однозначной идентификации строки в таблице, называется первичным ключом. Ключи обычно используются для объединения или объединения данных из двух или более таблиц. Например, таблица Employee может содержать столбец с именем Location, который содержит значение, соответствующее ключу таблицы Location. Ключи также важны при создании индексов, которые облегчают быстрый поиск данных из больших таблиц. Любой столбец может быть ключом, или несколько столбцов могут быть сгруппированы вместе в составной ключ. Нет необходимости заранее определять все ключи; Столбец можно использовать в качестве ключа, даже если он изначально не был предназначен.

Ключ, который имеет внешний, реальный смысл (например, имя человека, ISBN книги или серийный номер автомобиля), иногда называют «естественным» ключом. Если никакой естественный ключ не подходит (подумайте о многих людей по имени Браун), может быть назначен произвольный или суррогатный ключ (например, указывая идентификационные номера сотрудников). На практике большинство баз данных имеют как сгенерированные, так и естественные ключи, поскольку сгенерированные ключи могут использоваться внутри, чтобы создавать связи между строками, которые не могут сломаться, в то время как естественные ключи можно использовать, менее надежно, для поиска и для интеграции с другими базами данных. (Например, записи в двух независимо разработанных базах данных могут быть сопоставлены номером социального страхования, за исключением случаев, когда номера социального страхования неверны, отсутствуют или изменены).

Наиболее распространенным языком запросов, используемым с реляционной моделью, является язык структурированных запросов ([SQL](https://ru.wikipedia.org/wiki/SQL)).