КАРАГАНДИНСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ АБЫЛКАСА САГИНОВА
Кафедра ИВС

Дисциплина «Проектирование и разработка приложений БД»

Тема: «Трёхуровневая модель ANSI/SPARC: концептуальный, внешний, внутренний уровни»

ОП «Информационные системы», «IT - медицина»

Авторы:
ст.преп. каф. ИВС Саданова Б.М.
ст.преп. каф. ИВС Олейникова А.В.

2026

Архитектура ANSI/SPARC

Три уровня абстракции данных в системах управления базами данных.

ВНЕШНИЙ УРОВЕНЬ (External Level)
Пользовательские представления данных

External/Conceptual Mapping

КОНЦЕПТУАЛЬНЫЙ УРОВЕНЬ (Conceptual Level)
Общая логическая модель данных

Conceptual/Internal Mapping

ВНУТРЕННИЙ УРОВЕНЬ (Internal Level)
Физическое хранение данных

Концептуальный уровень

Общая логическая структура базы данных для сообщества пользователей.

Характеристики

- Единое представление всех данных предметной области
- Описывает, ЧТО хранится, а не КАК хранится
- Независим от физических деталей хранения
- Определяет ограничения целостности (constraints)
- Устанавливает связи между сущностями (1:1, 1:N, N:M)
- Описывает семантику данных — смысл и бизнес-правила

Содержимое концептуальной схемы

- Все сущности (таблицы, объекты, классы)
- Атрибуты и их типы данных (домены)
- Отношения между сущностями
- Ограничения целостности (PK, FK, CHECK, UNIQUE)
- Бизнес-правила и деривативные правила
- Авторизация доступа на уровне сущностей

Внешний уровень

Часть базы данных, интересующая конкретную группу пользователей.

Характеристики

- Множественность представлений для разных пользователей
- Скрывает сложность и объём полной концептуальной схемы
- Обеспечивает безопасность через ограничение видимости данных
- Может содержать вычисляемые (derived) поля
- Поддерживает логическую независимость
- Адаптирован под потребности конкретного пользователя

Типы пользователей

- Прикладные программисты — работают через API и ORM
- Конечные пользователи — используют готовые приложения
- Аналитики — выполняют сложные ad-hoc запросы
- Администраторы БД — управляют метаданными и правами
- Случайные пользователи — нерегулярные запросы к БД

Внутренний уровень

Физическая организация данных в памяти компьютера.

Физические аспекты

- Формат записей на диске (tuple layout, row format)
- Размер страниц и блоков данных (page size: 4KB, 8KB, 16KB)
- Методы размещения: heap, clustered, partitioned
- Сжатие данных (compression: Snappy, Zstd, LZ4)
- Шифрование на уровне хранения (TDE, AES-256)
- Разбиение на разделы (partitioning: range, hash, list)

Структуры доступа

- B-деревья и B+ деревья — O(log n) поиск, диапазоны
- Хеш-индексы — O(1) точный поиск по равенству
- Bitmap-индексы — аналитические запросы, low cardinality
- Инвертированные списки — полнотекстовый поиск
- GiST / GIN индексы — PostgreSQL, составные типы
- Постраничная организация (slotted page structure)

Пример: университетская база данных

Как одна предметная область выглядит на трёх уровнях.

КОНЦЕПТУАЛЬНЫЙ УРОВЕНЬ:
СТУДЕНТ (id, ФИО, дата_рождения, факультет_id)
КУРС (id, название, кредиты, преподаватель_id)
СВЯЗЬ: СТУДЕНТ — ЗАПИСЬ — КУРС (N:M)

ВНЕШНИЙ УРОВЕНЬ (представление «Студент»):
SELECT курс.название, оценка.балл
FROM оценка JOIN курс ON оценка.курс_id = курс.id
WHERE оценка.студент_id = CURRENT_USER_ID
→ Студент видит только СВОИ курсы и оценки

ВНУТРЕННИЙ УРОВЕНЬ (PostgreSQL):
Файл: base/16384/16385 | Страница: 8KB
B-tree PK на id → файл 16386
B-tree INDEX на фамилия → файл 16387
TOAST для полей > 2KB | WAL-журнал

Понятие схемы в ANSI/SPARC

Схема — это формализованное описание структуры данных на конкретном уровне абстракции.

Определение схемы

- Схема (Schema) — логическая структура базы данных
- Описывает объекты, их атрибуты и связи
- Является метаданными — «данными о данных»
- Хранится в системном каталоге СУБД
- Изменяется через DDL-команды (CREATE, ALTER, DROP)
- Каждый уровень ANSI/SPARC имеет свою схему

Три типа схем

- Концептуальная схема — единая логическая модель
- Внешние схемы (подсхемы) — множество представлений
- Внутренняя схема — физическая организация
- Соотношение: 1 концептуальная = N внешних = 1 внутренней
- Схемы связаны через отображения (mappings)
- Изменение одной схемы не обязательно требует изменения других

Концептуальная схема

Единая логическая модель всей базы данных.

Назначение

- Обеспечивает общее понимание данных для всех пользователей
- Служит основой для создания внешних схем
- Определяет ограничения целостности
- Является отправной точкой физического проектирования
- Поддерживает документирование предметной области
- Обеспечивает интеграцию данных из разных источников

Компоненты

- Сущности и их атрибуты с типами данных
- Первичные и внешние ключи
- Связи: один-к-одному, один-ко-многим, многие-ко-многим
- Ограничения: NOT NULL, UNIQUE, CHECK, DEFAULT
- Домены (domain) — допустимые значения атрибутов
- Триггеры и хранимые процедуры (бизнес-логика)

Внешние схемы (подсхемы)

Представления данных для конкретных групп пользователей.

Определение и свойства

- Подсхема — подмножество концептуальной схемы
- Может объединять данные из нескольких сущностей
- Скрывает конфиденциальные атрибуты
- Может содержать вычисляемые поля (derived)
- Реализуется через SQL VIEW или API endpoints
- Каждый пользователь работает со своей подсхемой

Примеры подсхем

- Студент: видит только свои курсы и оценки
- Преподаватель: видит группы, может выставлять оценки
- Декан: видит статистику по факультету
- Бухгалтер: видит стипендии, оплату, штрафы
- Администратор БД: имеет доступ ко всем уровням
- Аналитик: агрегированные данные без персоналий

Внутренняя схема

Физическое представление данных на уровне хранения.

Содержание

- Физические файлы данных на диске
- Структуры хранения: heap, clustered, IOT
- Индексы: B-tree, hash, bitmap, GiST
- Параметры хранения: page size, fillfactor
- Tablespaces и файловые группы
- Сжатие, шифрование, репликация

Отличие от концептуальной

- Концептуальная: ЧТО хранить (сущности, связи)
- Внутренняя: КАК хранить (файлы, блоки, индексы)
- Концептуальная: не зависит от СУБД
- Внутренняя: специфична для конкретной СУБД
- Концептуальная: понятна аналитикам и бизнесу
- Внутренняя: управляется администратором БД

Межуровневые отображения

Mappings — механизмы преобразования между уровнями.

Внешний уровень | **Концептуальный** | **Внутренний**

External/Conceptual Mapping

- Преобразует запросы пользователя в термины концептуальной схемы
- Реализуется через SQL VIEW и механизм авторизации
- Обеспечивает логическую независимость
- Пример: VIEW скрывает колонку «зарплата» от студентов
- При изменении концептуальной схемы — адаптирует VIEW

Conceptual/Internal Mapping

- Преобразует логические операции в физические команды
- Реализуется через оптимизатор запросов СУБД
- Обеспечивает физическую независимость
- Пример: SELECT → чтение страниц с диска через B-tree
- При добавлении индекса — автоматически использует его

Понятие независимости данных

Способность системы изменять один уровень без влияния на другие.

Определение

- Независимость данных — ключевое свойство ANSI/SPARC
- Изменение на одном уровне не требует изменений на других
- Обеспечивается механизмом межуровневых отображений
- Снижает стоимость сопровождения (TCO)
- Упрощает эволюцию и масштабирование БД
- Позволяет адаптировать БД под меняющиеся требования

Два типа независимости

- Логическая независимость — изменение концептуальной схемы
- Физическая независимость — изменение внутренней схемы
- Логическая защищает приложения от изменений структуры
- Физическая защищает логику от изменений хранения
- Обе обеспечиваются через External/Conceptual и Conceptual/Internal mappings
- Современные СУБД реализуют оба типа автоматически

Логическая независимость

Изменение концептуальной схемы не требует изменения внешних схем.

Определение

- Способность изменять концептуальную схему без модификации приложений
- External/Conceptual Mapping адаптирует подсхемы автоматически
- Приложения работают через VIEW — «стабильный интерфейс»
- Снижает затраты на сопровождение прикладного ПО
- Позволяет эволюционировать БД поэтапно
- Критична для долгоживущих enterprise-систем

Примеры изменений

- Добавление нового поля в таблицу (email, телефон)
- Разделение таблицы на две (нормализация)
- Объединение таблиц (денормализация для производительности)
- Добавление ограничения целостности (CHECK, FOREIGN KEY)
- Изменение типа данных (INT → BIGINT для больших значений)
- Добавление вычисляемого столбца (возраст из даты рождения)

Физическая независимость

Изменение внутренней схемы не требует изменения концептуальной.

Определение

- Способность изменять физическую организацию без модификации логики
- Conceptual/Internal Mapping обеспечивает прозрачность доступа
- Оптимизатор запросов автоматически использует новые структуры
- Приложения не знают о физическом расположении данных
- Позволяет оптимизировать производительность «на лету»
- Критична для масштабирования и облачных миграций

Примеры изменений

- Добавление индекса для ускорения запросов
- Перенос таблицы на SSD или в облачное хранилище
- Изменение размера страницы (4KB → 8KB → 16KB)
- Включение сжатия данных (TDE, columnar compression)
- Партиционирование большой таблицы по диапазону дат
- Репликация данных на другой сервер (master → slave)

Понятие СУБД

Система управления базами данных — программный комплекс для создания, хранения и управления данными.

Определение СУБД

- СУБД — программный комплекс для управления данными
- Обеспечивает создание, хранение, изменение и удаление данных
- Поддерживает многопользовательский доступ
- Гарантирует целостность и безопасность данных
- Реализует механизмы восстановления после сбоев
- Предоставляет язык запросов (SQL) для манипуляции данными

Место в ИС

- СУБД — центральный компонент информационной системы
- Связывает прикладное ПО с физическим хранилищем
- Изолирует приложения от деталей хранения данных
- Поддерживает интеграцию данных из разных источников
- Обеспечивает конкурентный доступ множества пользователей
- Является «мозгом» системы управления данными

Компоненты СУБД: Ядро и Процессор

Центральные модули обработки данных.

Ядро СУБД (Database Engine)

- Главный исполнительный модуль СУБД
- Управляет доступом к данным и их обработкой
- Координирует работу всех остальных компонентов
- Обеспечивает ACID-свойства транзакций
- Реализует механизмы блокировок и MVCC
- Управляет кэшем и буферным пулом в памяти

Процессор языка (DML/DDL)

- Компилятор SQL-запросов в машинные команды
- Разбор (parsing) и синтаксический анализ запросов
- Семантический анализ: проверка существования объектов
- Генерация дерева разбора (parse tree)
- Преобразование в реляционную алгебру
- Подготовка к оптимизации (query plan)

Интерфейсы СУБД

Способы взаимодействия приложений и пользователей с базой данных.

Внешние интерфейсы

- SQL/CLI — стандартный язык запросов (ISO SQL:2016)
- ODBC (Open Database Connectivity) — универсальный драйвер
- JDBC (Java Database Connectivity) — для Java-приложений
- ORM: Hibernate, SQLAlchemy, Django ORM, Prisma
- REST API / GraphQL — через middleware слои
- gRPC — высокопроизводительные RPC-вызовы

Внутренние интерфейсы

- Query Executor ↔ Buffer Manager (чтение страниц)
- Buffer Manager ↔ Storage Manager (дисковый I/O)
- Transaction Manager ↔ Lock Manager (блокировки)
- Storage Manager ↔ OS File System (системные вызовы)
- Catalog Manager ↔ System Tables (метаданные)
- WAL Writer ↔ Disk (журналирование изменений)

🕮 Понятие метаданных

Данные о данных — нервная система любой информационной системы.

Определение

- Метаданные — данные, описывающие другие данные
- Содержат структуру, типы, ограничения, связи
- Хранятся в системном каталоге СУБД
- Используются оптимизатором для построения планов
- Необходимы для авторизации и аудита
- Поддерживают Data Governance и Data Quality

Классификация метаданных

- Технические: типы данных, размеры, индексы, ограничения
- Операционные: даты создания, владельцы, права доступа
- Бизнес-метаданные: описания, теги, глоссарии терминов
- Происхождение (provenance): откуда данные пришли
- Использование (lineage): кто и как использует данные
- Качество (profiling): статистики, аномалии, скоринг

Системный каталог СУБД

Внутренняя база данных СУБД, хранящая метаданные обо всех объектах.

Назначение

- Хранит метаданные о всех объектах БД
- Таблицы, столбцы, индексы, ограничения, триггеры
- Пользователи, роли, привилегии доступа
- Статистика распределения данных для оптимизатора
- Информация о tablespaces, файлах, репликации
- Журналы аудита и системные события

Особенности

- Реализован как набор системных таблиц
- Доступен через специальные VIEW и INFORMATION_SCHEMA
- Обновляется автоматически при DDL-операциях
- Защищен от прямого изменения пользователями
- Используется СУБД для парсинга и оптимизации
- Является основой для инструментов администрирования

Словарь данных

Бизнес-ориентированное описание данных, выходящее за рамки технического каталога.

Отличие от системного каталога

- Системный каталог = технические метаданные (для СУБД)
- Словарь данных = бизнес-метаданные (для аналитиков)
- Каталог: column type VARCHAR(255)
- Словарь: ФИО клиента — полное имя физического лица
- Каталог: foreign key constraints
- Словарь: бизнес-правила, владельцы данных, SLA

Содержимое словаря

- Бизнес-описания сущностей и атрибутов
- ER-диаграммы и модели предметной области
- Глоссарии терминов и определений
- Data lineage — происхождение и трансформации данных
- Политики качества данных (Data Quality Rules)
- Владельцы данных (Data Stewards) и контакты

Каталог в архитектуре ANSI/SPARC

Метаданные всех трех уровней в единой системе.

Метаданные уровней

- Внешний уровень: VIEW definitions, column aliases
- Концептуальный уровень: tables, columns, keys, constraints
- Внутренний уровень: files, indexes, page sizes, tablespaces
- Отображения: mapping rules between levels
- Безопасность: grants, roles, audit trails
- Статистика: row counts, histograms, correlations

Роль каталога

- Парсер использует каталог для проверки имен объектов
- Оптимизатор читает статистику для выбора плана
- Executor проверяет права доступа через каталог
- DDL-команды автоматически обновляют каталог
- Администраторы запрашивают каталог для диагностики
- Инструменты BI используют каталог для построения отчетов