**Базы данных**

**Реляционная модель данных**

**Реляционная алгебра**

Реляционная модель состоит из 3 частей

1. Структурная часть – описывает какие объекты рассматриваются реляционной моделью.
2. Целостная часть
3. Манипуляционная часть

Домен – семантическое понятие подмножество значений некоторого типа данных имеющих определенный смысл.

Ключевое поле – основное поле таблицы базы данных

Селекция – выборка по строкам

Проекция – выборка по столбцам

Объединение А\*В | Select\*from A Union Select\*from B входят все строки кроме одинаковых

Декартовое произведение – произведение каждого с каждым (умножение матриц) A TIMES B

Пересечение - входят только одинаковые строки

Разность

Естественное соединение

**Выборка (ограничение, селекция)**

* Селекция: Select \* from A Where Зарплата < 3000
* Проекция

Соединения:

1. Общая операция соединения - соединение по некоторому условию (Декартовое произведение)
2. Тэта-соединения – частный случай ООС (A=B, A<=B, A>B) Сравниваются значения в столбцах таблицы
3. Экви-соединение – частный случай Тета-соединения A = B
4. Естественное соединения – частный случай Экви соединения. Какие то значения из таблицы B должны совпадать со значениями из таблицы А.

Ассоциативная таблица – таблица связывающая 2 другие таблицы

* Деление – из таблицы соответствия выделяется нужное значение, и выводятся все значения соответствующие выбранному значению

**Реляционные СУБД**

Первичный ключ – обязательный атрибут таблицы

Внешний ключ – служит для связи таблиц

2 или более внешних ключей могут составлять первичный ключ.

**Степени связи:**

1. 1 к 1 – встречается редко.
2. 1 к многим – одной записи первой таблицы соответствует несколько записей другой таблицы
3. Многое ко многим – 1 к многим в обе стороны. Пример: студенты и предметы.

Сущность – отношение – таблица

Кортеж = запись = строка

Атрибут = поле = стробец

12 правил кодда:

1. Реляционная СУБД способна полностью управлять базой данных
2. Вся информация в бд должна быть строго в таблицах
3. Гарантированный доступ – любое значение должно быть доступно через комбинацию имени таблицы значение
4. СУБД должна работать с пустыми значениями
5. Описание БД и ее содержание можно доставать с помощью запросов
6. Должен быть четко определенный синтаксис. Поддерживать – описание структуры данных и манипулирование данных, правила целостности, авторизацию и транзакцию
7. Представление является конечным
8. Обязательно – Вставка, обновление и удаление
9. Информация не влияет на программу СУБД
10. На программы не влияют изменения структур таблиц
11. Должен быть способен определять правила целостности
12. Независимо первый раз используется данные или повторно
13. Невозможность обойти правила целостности

**Язык ДДЛ**

Создать:

CREATE TABLE(INDEX VIEW)

Удалить:

DROP TABLE(INDEX VIEW)

TRUNCATE TABLE

ALTER TABLE

**DML язык**

INSERT – добавить запись

DELETE – удалить запись

UPDATE – обновить значение записи

SELECT – выбрать

***Все команды с прописной буквы, все остальное с маленькой***

PRIMARY KEY(id) – обозначение первичного ключа

CHECK (….) проверка значения

ON DELETE NO ACNION – ПРИ УДАЛЕНИИ ПЕРВИЧНОГО КЛЮЧА НЕ УДАЛЯЮТСЯ ДАННЫЕ СВЯЗАННЫХ ДАННЫХ

Индексирование:

CREATE INDEX index\_name ON table\_name (column\_name) проще поиск данных и сортировка

ALTER TABLE table\_name ADD column\_name datatype – добавить столбец в таблицу

ALTER TABLE table\_name DROP COLUMN – УДАЛЯЕТ

INSERT INTO table\_name (column1, …) VALUES(value1)

INSERT INTO table\_name (column1, …) VALUES(value1)

SELECT column3

FROM table2\_name;

UPDATE table\_name

SET column1 = [new value] , …

WHERE column1 = “aaaa”

DELETE FROM table\_name

WHERE …..

ALTER TABLE users

ADD u\_phone CHAR(16)

CHECK(u\_phone LIKE ‘\_(\_\_\_)\_\_ \_\_ \_\_\_’)

RESTRICT – запретить действия напр. DELETE UPDATE.

**Select**

SELECT column\_name FROM table

SELECT column\_name **FROM table AS table\_alias**

SELECT **column\_name AS column\_alias** FROM table

SELECT **DISTINCT** column\_name FROM table – без повторов в столбце

WHERE column LIKE pattern ‘%AN%’; - выведется то где есть AN в строке

‘O%’ – начинается с О.

ORDER BY – сортировка. ASC или DESC

ORDER BY 2 DESC – сортировка второго столбца (второе название в SELECT \_\_\_\_\_).

SELECT AVG(age) FROM person

LIKE ‘[r-w]’ - в диапазоне, ‘[^r-w]’ – не в диапазоне

**Множественный Select**

**Подзапросы или Вложенный select**

UNION – объединение множеств повторы не выводятся

UNION ALL – объединение множеств повторы выводятся

INTERSECT – пересечение

MINUS – убираем того что есть во второй таблице

JOIN – соединение

INNTER JOIN – явное соединение

SELECT

FROM first\_table

INNER JOIN second\_table

ON first\_table.asdf\_ID = second\_table.asdf\_ID

SELECT

FROM first\_table

LEFT JOIN second\_table

ON first\_table.asdf\_ID = second\_table.asdf\_ID

Если в первой таблице нет соответствия из второго то значение все равно выводится но во второй столбец закидывается пустая ячейка

SELECT

FROM first\_table

RIGHT JOIN second\_table

ON first\_table.asdf\_ID = second\_table.asdf\_ID

Если в первой таблице нет соответствия из второго то значение все равно выводится но во второй столбец закидывается пустая ячейка

Группирование GROUP BY

GROUP BY (столбец) HAVING – работает почти как WHERE

SELECT column\_name

INTO (внешний ресурс) IN “backup.mdb” (это файл)

FROM table\_name

Представления CREATE VIEW

CREATE VIEW view\_name AS

SELECT column\_name(s) FROM table\_name

WHERE condition

(создается виртуальная таблица которая удаляется после закрытия все это время можно с ней работать. Проще один раз открыть и работать с таблицей чем каждый раз открывать через select. При изменении в бд изменяется и view)

**DKL**

GRANT {ПРИВИЛЕГИИ}

ON {ОБЪЕКТЫ} TO {ПОЛЬЗЬВАТЕЛИ} [WITN ADMIN OPTION] Дать привилегии

REVOKE {ПРИВИЛЕГИИ}

ON {ОБЪЕКТЫ} FROM {ПОЛЬЗЬВАТЕЛИ} Снять привилегии

**TCL**

START TRANSACTION;

…

IF (ВСЕ ЗАВЕРШИЛОСЬ ХОРОШО) THEN COMMIT; --“утвердить” изменения

ELSE ROLLBACK; -- восстановить состояние до начала транзакции (до старта)

END IF;

*Это неделимый набор действий*

LOCK TABLES tbl\_name [AS alias]

{READ [LOCAL] | [LOW\_PRIORITY] WRITE}

[, tbl\_name [as alial] {READ [LOCAL] | [LOW\_PRIORITY] WRITE} …]

…

UNLOCK TABLES

Это закрывает доступ к данным во время выполнения операций.

**Сущность**

Сущность – любой различимый объект (объект, который мы можем отличить от другого)

Тип сущности – Автомобиль(Таблица)

Экземпляр сущности – BMW X3 a777aa59rus (Строка таблицы)

Основные классы сущностей:

- Стержневая сущность (человек) – независимая сущность

- Характеристическая сущность (Паспорт (данные от туда) связь один к одному или один ко многим от стержневой сущности)

- Обозначающая сущность (Кодовая сущность (Enum)) (Пол – не может быть без стержневой сущности(один ко многим, один к одному) не зависит от стержневой сущности)

- Ассоциативная сущность ((Брак)Для связи таблиц. Для связи многое ко многим)

|  |  |
| --- | --- |
| 1 | 1 |
| 1 | 2 |
| 2 | 1 |

Атрибут – поименованная характеристика сущности (цвет, марка авто)

Ключ – минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр. Пример: регистрационный номер. Суррогатный ключ – ключ который может повторяться.

Отношения по ключам:

Один к одному

Один ко многим

Многие к одной

Сущность – связь

1. Прямая связь
2. Косвенная связь (через таблицу)
3. Связи по месту и времени(по условию а не по ключу)

**Индексация СУБД**

Основная проблема СУБД – это поиск нужных данных за минимальное время

Быстрый поиск может быть выполнен в случае если данные отсортированы.

Сортировка данных в таблице невозможна, т к критериев поиска может быть несколько.

Поэтому для таблицы с данными создаются специальные таблицы для каждого критерия поиска, которые называются индексы.

Индексы – структура данных для быстрого поиска записей в таблице ключей.

Методы организации индекса:

1. По первичному ключу
   1. Плотный индекс (Удаление и добавление записи чуть медленнее из-за добавления записи в индексную таблицу. Но поиск гораздо быстрее)
      1. Основная таблица
         1. Содержит последовательность записей одинаковой длины в произвольном порядке
      2. Индексная таблица
         1. Индексные записи в отсортированном порядке содержит
            1. Индексный ключ
            2. Номер записи в основной таблице
   2. Неплотный индекс (Если данные хранятся уже в отсортированном виде)
      1. Основная таблица
         1. Содержит последовательность записей одинаковой длины в отсортированном виде
      2. Индексная таблица
         1. Индексные записи в отсортированном порядке содержит
            1. Индексный ключ
            2. Номер записи в основной таблице
   3. Б-деревья
      1. Индекс Б-дерево. Существует несколько уровней. Оно спускается из родительских уровней к дочерним. Нужно если содержится очень много записей. К примеру
         1. 1 уровень 1-10000 100001-20000. Т
         2. 2 уровень 1- 100 101-200
2. По вторичному ключу
   1. Инвертируемые списки
   2. Приведенный к индексу РК

**Нормализация баз данных**

**Схемой** базы данных называется структура связей между полями и таблицами.

Нормализацией схемы базы данных называется процедура, производимая над базой данных с целью удаления в ней избыточности.

Первая нормальная форма (ПНФ) – каждый атрибут отношения должен хранить атомарное значение, каждое отношение (строка в таблице) должно содержать одинаковое количество атрибутов (столбцов) т. Е.:

1. Запрещает повторяющиеся столбцы (содержащие одинаковую по смыслу информацию)
2. Запрещает множественные столбцы

Вторая нормальная форма (2НФ)- отношение находится во второй нормальной форме, если оно находится в 1НФ, и при этом все исключенные атрибуты зависят только от первичного ключа.

Третья нормальная форма (3НФ) – отношение находится в третьей нормальной форме если каждый столбец, не являющийся ключом, зависит **только** от столбца, который является ключом, то есть должна отсутствовать **транзитивная функциональная зависимость** (transitive functional dependency)

Транзитивная функциональная зависимость выражается следующим образом: А → В и В → С. То есть атрибут С транзитивно зависит от атрибута А, если атрибут С зависит от атрибута В, а атрибут В зависит от атрибута А (при условии, что атрибут А функционально не зависит ни от атрибута В, ни от атрибута С).

Аномалии – это проблемы возникающие в данных из-за дефектов проектирования БД:

1. Аномалия вставки – проявляются при вводе данных в дефектную таблицу
2. Аномалия удаления – возникают при удалении данных из дефектной схемы
3. Аномалия модификации возникают при изменении данных дефектной схемы

Нормализация отношений – это формальный аппарат ограничений на формирование отношений, который позволяет устранить дублирование и потенциальную противоречивость.

Основные виды зависимости:

Функциональная зависимость – атрибут В функционально зависит от атрибута А, если каждому значению А соответствует в точности одно значение В.

Если A=> B и B=> C то А=>С

Многозначная зависимость – для каждого атрибута соответствует множество других атрибутов.

Нормальная форма Бойса-Кодда (БКНФ) – развитие 3НМ и требует чтобы в отношении были только функциональные зависимости, левая часть которых является потенциальным ключом отношения.

Детерминантным называется левая часть функциональной зависимости.

**Ненормальная схема**

Расщепление схемы по строкам

**Первая нормальная схема**

Расщепление схемы по столбцам

**Вторая нормальная схема**

Расщепление схемы по схемам

**Третья нормальная схема** практически всегда достаточна

Полной декомпозицией отношений называют такую совокупность произвольного числа его проекций, соединение которых позволяет получить исходное отношение

Отношение находится в 5НФ когда в каждой его полной, декомпозиции все проекции содержат возможный ключ. Отношение, не имеющее ни одной полной декомпозиции, также находится в 5НФ.

Ассоциативная таблица содержит первичный ключ как два внешних ключа

Четвертая нормальная форма (4НФ) – является частным случаем 5НФ.

Ассоциативная таблица содержит отдельный столбец с первичным ключом.

Полная функциональная зависимость – если атрибут Y не зависит функционально от множества X