Progresif: Jurnal Ilmiah Komputer
https://ojs.stmik-banjarbaru.ac.id/index.php/progresif/index
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0877

Analisis *Trade-Off* Arsitektur *Multi-Tenant* Sebagai Dasar Justifikasi Strategi Basis Data SaaS

DOI: http://dx.doi.org/10.35889/progresif.v22i2.3590

Siti Tiara Nur Aziza[1*], Yopi Nugraha[2]
Sistem Informasi, Institut Pendidikan Indonesia Garut, Garut, Indonesia
**e-mail Corresponding Author:* tiaranuraziza517@gmail.com

Abstrac

Designing a database architecture within a multi-tenant software as a service environment brings forth strategic challenges related to balancing cost efficiency, data separation, and scalability. The aim of this study is to provide a foundational framework for such systems by analyzing the trade-offs among three distinct database architecture models. By employing a case study that involves the deployment of a web-based system, the descriptive qualitative methodology utilizes data collection techniques such as document reviews, system observations, and artifact analyses. The evaluation results indicate that no single model stands out as the best, as each one presents its own set of competing advantages and disadvantages. The innovation in this research lies in the creation of a conceptual compromise model intended to enhance decision-making in database design. The choice of approach should be tailored to meet the specific requirements of the system and its operational context.

Keywords: *SaaS; Multi-tenant; Trade-off analysis; Database architecture; Shared schema.*

Abstrak

Merancang arsitektur basis data dalam lingkungan perangkat lunak sebagai layanan multi-tenant menghadirkan tantangan strategis terkait dengan keseimbangan antara efisiensi biaya, pemisahan data, dan skalabilitas. Tujuan dari penelitian ini adalah untuk menyediakan suatu rencana dasar untuk sistem tersebut dengan membandingkan pertukaran di antara tiga jenis arsitektur basis data yang berbeda. Dengan menggunakan studi kasus yang melibatkan penyebaran sistem berbasis web, metodologi kualitatif deskriptif menerapkan teknik pengumpulan data yang mencakup tinjauan dokumen, pengamatan sistem, dan analisis artefak. Hasil evaluasi menunjukkan bahwa tidak ada satu model yang benar-benar menonjol karena masing-masing memiliki kelebihan dan kekurangan yang saling bertentangan. Inovasi dalam penelitian ini terletak pada pengembangan model kompromi konseptual yang bertujuan untuk meningkatkan pengambilan keputusan dalam desain basis data. Pemilihan pendekatan harus disesuaikan dengan kebutuhan spesifik dari sistem dan lingkungan kerjanya.

Kata kunci: *Perangkat lunak sebagai layanan; Multi-tenant; Analisis trade-off; Arsitektur basis data; Skema basis data*

1. Pendahuluan

Perkembangan dalam teknologi *Software as a Service* (SaaS) meningkatkan permintaan akan desain sistem yang efektif, khususnya dalam pengelolaan database *multi-tenant.* Desain *multi-tenant* memungkinkan satu aplikasi untuk mendukung banyak pengguna (*tenant*) dalam satu struktur, sehingga menghasilkan efisiensi biaya dan kemudahan dalam manajemen. Namun, penerapan desain ini disertai dengan berbagai *trade-off*, seperti isu keamanan data, performa sistem, kemampuan skalabilitas, dan kompleksitas dalam pengelolaan. Dengan demikian, analisis *trade-off* pada desain *multi-tenant* sangat penting untuk menentukan strategi database yang sesuai dalam pengembangan SaaS, sehingga sistem yang dikembangkan dapat memenuhi kebutuhan pengguna dengan baik sambil tetap menjaga efisiensi dan keandalan layanan [1].

Dalam penerapannya, sistem yang dibangun dengan model SaaS dan menggunakan arsitektur *multi-tenant* sering kali menemui berbagai hambatan, terutama dalam menentukan metode pengelolaan basis data yang paling tepat [2]. Setiap strategi, seperti penggunaan basis data bersama dengan skema yang sama (*Shared Database Shared Schema*), skema yang terpisah (*Separate Schema*), atau basis data yang berbeda (*Separate Database*), memiliki kelebihan dan kekurangan masing-masing [3]. Dalam konteks penelitian ini, yaitu pengembangan sistem manajemen berbasis SaaS, muncul tantangan dalam menemukan keseimbangan antara kebutuhan efisiensi sumber daya dan tuntutan akan keamanan serta pemisahan data antar *tenant*. Di samping itu, bertambahnya jumlah pengguna juga memengaruhi kinerja sistem dan kerumitan dalam pengelolaan data. Situasi ini menyoroti perlunya analisis yang lebih mendalam mengenai *trade-off* dari setiap arsitektur yang diterapkan, untuk menghindari pemilihan strategi yang kurang tepat yang dapat menyebabkan penurunan kualitas layanan.

Dalam lapisan penyimpanan data, terdapat tiga jenis implementasi utama yang sering diterapkan dalam sistem SaaS, yaitu basis data yang digunakan bersama dengan skema yang sama, basis data yang digunakan bersama dengan skema yang berbeda, dan basis data yang terpisah untuk setiap pengguna [4][5]. Setiap jenis model ini memiliki sifat yang berbeda, di mana skema yang digunakan bersama memiliki keunggulan dalam efisiensi penggunaan sumber daya, sementara basis data yang terpisah memberikan tingkat keamanan dan isolasi yang lebih baik. Pemilihan model ini secara langsung memengaruhi kualitas sistem, termasuk aspek seperti kemampuan untuk berkembang, kinerja, dan kemudahan dalam perawatan.

Namun, dalam implementasinya, menentukan arsitektur basis data *multi-tenant* bukanlah hal yang mudah. Model skema bersama dapat menimbulkan masalah terkait isolasi data, sedangkan model basis data terpisah dapat membuat biaya dan kerumitan operasional meningkat. Sementara itu, model skema terpisah berada di antara kedua pilihan tersebut dengan beberapa kompromi. Situasi ini menandakan adanya pertukaran yang kompleks antara efisiensi sumber daya, keamanan data, kinerja sistem, dan kesulitan dalam pengelolaan.

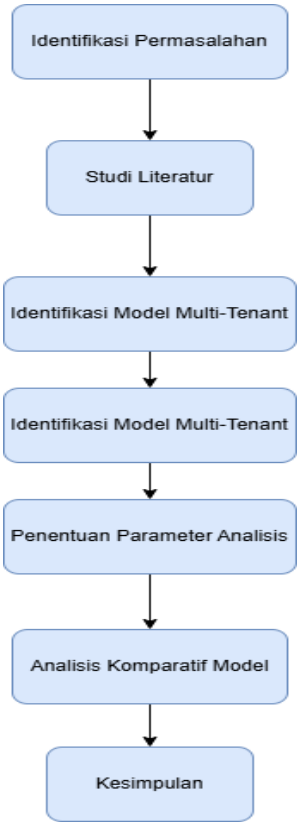
Beragam studi sebelumnya telah menelaah berbagai sisi arsitektur *multi-tenant*, seperti keamanan dalam SaaS [6][7], model penerapan basis data, pengoptimalan struktur *multi-tenant*, dan penilaian arsitektur berdasarkan atribut kualitas dengan metode seperti ATAM [8]. Namun, kebanyakan penelitian tetap berfokus pada satu atau beberapa atribut kualitas secara terpisah dan belum menggabungkannya ke dalam suatu kerangka evaluasi yang sistematis dan komparatif[9]. Di samping itu, penelitian yang khusus mendukung pengambilan keputusan strategis dalam memilih arsitektur basis data *multi-tenant* dalam konteks SaaS masih sangat terbatas.

Oleh karena itu, diperlukan metode analisis perbandingan yang dapat mengidentifikasi *trade-off* antara model arsitektur secara komprehensif, yang mencakup efisiensi, keamanan[10], skalabilitas, dan kompleksitas. Berbeda dengan penelitian sebelumnya yang lebih banyak menitikberatkan pada peningkatan teknis dari arsitektur *multi-tenant* untuk bisnis besar, studi ini memberikan sumbangan yang lebih spesifik terhadap strategis basis data dalam konteks sistem manajemen masjid berbasis SaaS. Inovasi dari penelitian ini terletak pada pembuatan kerangkat analisis *trade-off* yang disesuaikan dengan sifat organisasi keagamaan, yang memiliki struktur operasional yang seragam namun melibatkan banyak entitas tenant. Dengan menggabungkan analisis karakteristik arsitektur dan evaluasi parameter strategis, penelitian ini memberikan penjelasan mengenai sejauh mana efektivitas model *Shared Database Shared Schema* sebagai Solusi yang skalabel dan efisien. Temuan dari analisis ini diharapkan menjadi pedoman strategis dalam memilih desain arsitektur basis data yang paling tepat untuk pengembangan platform SaaS pada organisasi nirlaba yang memiliki keterbatasan dalam sumber daya [11][12][13][14][15].

3. Metodologi Penelitian

Penelitian ini menggunakan pendekatan analisis perbandingan untuk menilai berbagai model arsitektur basis data *multi-tenant* yang digunakan dalam sistem *Software as a Service* (SaaS). Penilaian dilakukan dengan membandingkan karakteristik dari setiap model berdasarkan kriteria penilaian tertentu untuk mengidentifikasi kompromi yang muncul akibat penerapan strategi basis data dalam lingkungan *multi-tenant*.

Tahapan penelitian yang dilakukan dalam studi ini ditunjukkan dalam diagram metodologi penelitian yang terdapat pada Gambar 1. Diagram ini menggambarkan alur proses penelitian, dimulai dari pengidentifikasian masalah hingga tahap analisis dan kesimpulan.



Gambar 1. Diagram Metodologi

Berdasarkan Gambar 1, penelitian ini dilaksanakan melalui beberapa tahap yang mencakup pengidentifikasian masalah, tinjauan pustaka, pengenalan model arsitektur basis data *multi-tenant*, penentuan parameter analisis, dan analisis perbandingan terhadap model yang sedang diteliti. Tahapan ini digunakan untuk memahami kelebihan dan kekurangan masing-masing model arsitektur dalam mendukung strategi basis data di sistem *Software as a Service* (SaaS).

3.1. Objek Penelitian

Fokus dari penelitian ini adalah model arsitektur basis data *multi-tenant* yang sering dijumpai di sistem SaaS. Model-model yang sedang diteliti, meliputi *Shared Database Shared Schema*, *Shared Database Separate Schema* dan *Separate Database*. Ketiga model ini dipilih karena sering digunakan dalam penerapan arsitektur SaaS serta memiliki karakteristik yang berbeda mengenai isolasi data, kemampuan untuk meningkat, dan efisiensi pemanfaatan sumber daya.

3.2. Metode Pengumpulan Data

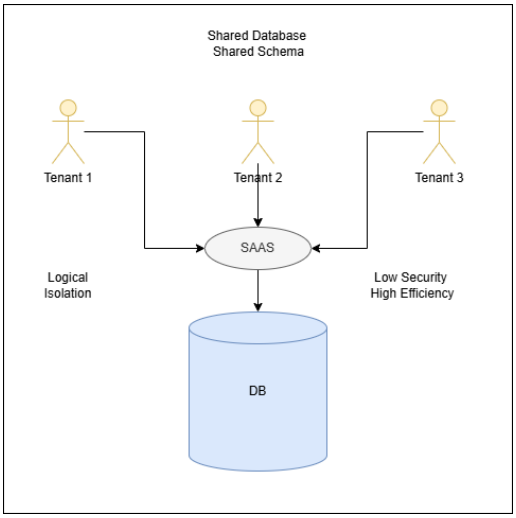
Data penelitian dikumpulkan melalui tinjauan literatur dari berbagai publikasi ilmiah yang membahas arsitektur basis data *multi-tenant* dalam sistem SaaS. Sumber-sumber yang digunakan mencakup artikel jurnal, prosiding konferensi, dan referensi akademis lain yang relevan dengan topik penelitian.

3.3. Model Arsitektur yang Dianalisis

Penelitian ini menganalisis beberapa model arsitektur basis data *multi-tenant* yang umum digunakan dalam sistem SaaS. Setiap model memiliki pendekatan berbeda dalam pengelolaan data *tenant*.

Model yang dianalisis meliputi, *Shared Database Shared Schema*, dimana seluruh tenant menggunakan database dan schema yang sama. *Shared Database Separate Schema*, dimana tenant menggunakan database yang sama tetapi memiliki schema yang berbeda. *Separate Database*, dimana setiap tenant memiliki database yang terpisah. Ketiga model ini dibandingkan berdasarkan kriteria evaluasi tertentu, yaitu *scalability, security, cost efficiency*, *performance*, dan *Maintainability*, untuk menyoroti pertukaran yang terkait dengan setiap pendekatan.

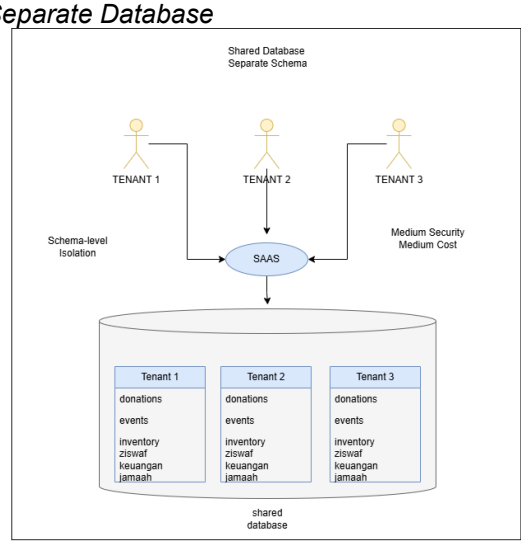
1) Shared Database – Shared Schema



Gambar 2. *Shared Database – Shared Schema*

Seperti yang terlihat pada Gambar 3. 1, model skema basis data bersama mengintegrasikan semua data tenant ke dalam satu basis data fisik dengan struktur tabel yang konsisten. Penyewaan diidentifikasi dengan atribut seperti *tenant_id*, sehingga pemisahan data dilakukan secara logis dalam tabel yang sama.
Dari segi efisiensi biaya dan kemampuan bertumbuh, model ini sangat efektif karena penggunaan sumber daya yang terpusat serta mengurangi duplikasi data. Namun, dari sudut pandang keamanan, model ini membawa risiko yang lebih besar, terutama jika terjadi kesalahan dalam pengelolaan akses atau *query*. Selain itu, dari aspek pemeliharaan, kompleksitas meningkat sejalan dengan bertambahnya jumlah *tenant* karena seluruh data disimpan dalam satu skema.

2) Shared Database – Separate Database

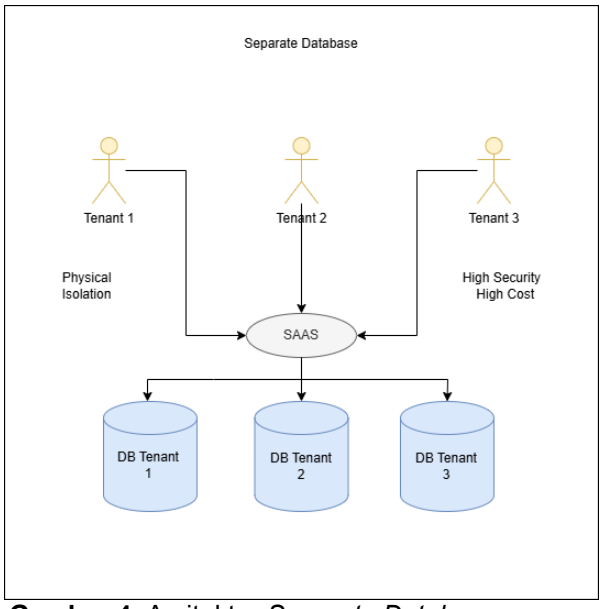


Gambar 3. Arsitektur *Shared Database - Separate Database*

Pada model ini, semua *tenant* menggunakan satu basis data yang sama, namun masing-masing memiliki skema yang berbeda. Setiap *tenant* memiliki serangkaian tabel sendiri dalam satu database, sehingga pemisahan data dilakukan di tingkat skema.

Dari perspektif keamanan, model ini lebih unggul dibandingkan skema bersama karena ada pemisahan data yang lebih jelas. Dalam hal kinerja, model ini masih cukup efisien, walaupun dapat mengalami penurunan jika jumlah *tenant* sangat banyak. Dari sisi kemudahan pemeliharaan, model ini lebih fleksibel karena perubahan pada satu *tenant* tidak langsung berpengaruh kepada *tenant* lainnya, tetapi pengelolaan skema menjadi lebih rumit. Dari segi efisiensi biaya, model ini berada pada tingkat menengah.

3) Separate Database



Gambar 4. Arsitektur *Separate Database*

Model *Separate Database* menyediakan satu database yang berbeda untuk setiap *tenant*. Dengan cara ini, setiap *tenant* memiliki kendali penuh atas data mereka masing-masing.

Di sisi keamanan, model ini memberikan tingkat pemisahan tertinggi karena tidak ada data yang saling dibagikan di antara *tenant*. Dari segi kinerja, model ini juga cenderung lebih stabil karena beban tidak dibagi dengan *tenant* lainnya. Namun, dari perspektif efisiensi biaya, model ini memerlukan lebih banyak sumber daya karena setiap *tenant* memiliki database pribadi. Dalam hal pemeliharaan, pengelolaan menjadi lebih rumit terutama saat jumlah *tenant* bertambah.

3.4. Parameter Analisis

Penilaian untuk setiap model arsitektur dilakukan dengan memperhatikan sejumlah parameter evaluasi yang ditampilkan dalam tabel berikut:

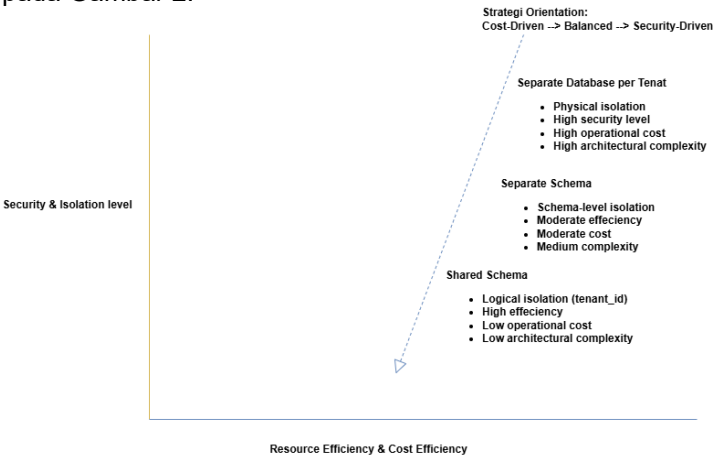
Tabel 1. Evaluasi Arsitektur Parameter Basis Data *Multi-Tenant*

Parameter	Deskripsi
Scalability	Kemampuan sistem menangani peningkatan jumlah tenant
Security	Tingkat isolasi dan keamanan data antar tenant
Cost Efficiency	Efisiensi penggunaan sumber daya infrastruktur
Performance	Performa sistem dalam mengakses dan memproses data
Maintainability	Kemudahan pengelolaan dan pemiliharaan sistem

Parameter-perparameter ini dimanfaatkan untuk menilai keunggulan dan keterbatasan dari setiap model struktur basis data *multi-tenant*.

3.5. Metode Analisis

Pendekatan analitis yang digunakan dalam penelitian ini adalah analisis komparatif, yang meliputi penilaian karakteristik setiap model arsitektur berdasarkan kriteria evaluasi yang sudah ditetapkan. Hasil dari analisis ini digunakan untuk mengidentifikasi kompromi masing-masing model dalam memfasilitasi strategi manajemen basis data dalam sistem SaaS.
Berdasarkan kriteria evaluasi yang telah ditetapkan, penelitian ini menyusun suatu model konseptual untuk menggambarkan hubungan timbal balik antara tingkat keamanan dan efisiensi sumber daya dalam arsitektur basis data *multi-tenant*. Model konseptual tersebut ditampilkan pada Gambar 2.



Gambar 2. Model Konseptual *Trade-off* Strategi Basis Data *Multi-Tenant* dalam lingkup SaaS

Gambar 2 menunjukkan posisi perbandingan dari tiga model arsitektur basis data multi-tenant, yaitu *Shared Schema*, Separate Schema, dan *Separate Database*, berdasarkan tingkat isolasi keamanan dan efisiensi sumber daya mereka. Model *Shared Schema* berada pada sisi efisiensi tinggi dengan biaya operasional yang rendah. Sebaliknya, model *Separate Database* berada pada sisi Tingkat isolasi keamanan yang tinggi, namun disertai dengan biaya operasional yang lebih besar.

4. Hasil dan Pembahasan

4.1. Analisis Struktur dan Mekanisme Isolasi Data

Analisis dalam penelitian ini dilakukan terhadap tiga model arsitektur basis data *multi-tenant*, yaitu *Shared Database – Shared Schema*, *Shared Database – Separate Schema*, dan *Separate Database*. Ketiga model ini dipilih karena umum digunakan dalam sistem *Software as a Service* (SaaS) dalam mengelola data dari banyak tenant.

Evaluasi dilakukan dengan menggunakan lima indikator yang sudah ditetapkan dalam metode penelitian, yaitu *scalabilitas*, *security*, *cost efficiency*, *performance*, dan *maintainability*. Indikator-indikator ini dipakai untuk menganalisis keunggulan serta kelemahan dari setiap model arsitektur.

Dengan analisis ini, diharapkan akan didapatkan pemahaman perbandingan yang tegas sehingga dapat mendukung dalam memilih model arsitektur yang paling tepat untuk kebutuhan sistem.

Tabel 1. Analisi Perbandingan Model Arsiektur *Multi-Tenant*

Kriteria Evaluasi	***Shared Database – Shared Schema* (Implementasi Sistem Masjid)**	***Shared Database – Separate Schema***	***Separate Database* Per-Masjid**
Struktur Basis Data	Satu database (masjid) dengan table bersama seperti donations, events, keuangan, inventory, jamaah	Satu database dengan *schema* terpisah setiap masjid	Satu database terpisah untuk setiap masjid
Manajemen *Tenant*	Table tenants dan domains sebagai identitas masjid	Identitas masjid melalui *schema*	Identitas masjid melalui database

Kriteria Evaluasi	***Shared Database – Shared Schema* (Implementasi Sistem Masjid)**	***Shared Database – Separate Schema***	***Separate Database* Per-Masjid**
Pemisahan Data Masjid	Logis (filter *tenant_id* pada *query*)	Semi-fisik (*schema* terpisah)	Fisik penuh (database terpisah)
Skalabilitas	Sangat tinggi, penambahan masjid tidak memerlukan perubahan struktur	Tinggi, namun penambahan schema meningkatkan kompleksitas	Rendah-sedang, tergantung kapasitas server
Efesiensi Sumber Daya	Sangat efesien, table digunakan bersama seluruh masjid	Cukup efesien	Kurang efesien
Performa Sistem	Baik untuk jumlah masjid kecil-menengah	Stabil untuk beban menengah	Sangat baik karena beban terisolasi
Keamanan & Isolasi Data	Bergantung pada *middleware tenancy* dan kontrol *query*	Lebih aman karena pemisahan schema	Sangat aman karena pemisahan database
Kompleksitas Implementasi	Rendah, mudah dikembangkan dengan *laravel tenancy*	Sedang	Tinggi
Biaya Operasional	Rendah	Sedang	Tinggi
Kemudahan untuk Sistem Masjid SaaS	Tinggi, satu struktur tabel	Sedang	Rendah
Kesesuaian untuk Sistem Masjid SaaS	Sangat sesuai untuk banyak masjid dengan fitur seragam	Cocok jika kebutuhan isolasi meningkat	Cocok untuk masjid besar dengan regulasi ketat

Berdasarkan Tabel 1, model *Shared Database – Shared Schema* menggunakan satu basis data dan satu skema yang sama untuk seluruh *tenant*. Data dibedakan menggunakan atribut seperti *tenant_id*. Model ini unggul dalam *cost efficiency* dan *scalability*, namun memiliki tingkat isolasi data yang rendah karena pemisahan hanya dilakukan secara logis.

Model *Shared Database – Separate schema* menggunakan satu basis data dengan skema terpisah untuk setiap *tenant*. Pendekatan ini memberikan isolasi data yang lebih baik dibandingkan model sebelumnya, tetapi meningkatkan kompleksitas dalam pengelolaan skema. Sementar itu, model *Separate Database* menyediakan basis data yang terpisah untuk setiap *tenant*. Model ini menawarkan tingkat isolasi dan keamanan tertinggi, namun membutuhkan sumber daya yang lebih besar serta pengelolaan yang lebih kompleks.

Secara umum, semakin tinggi tingkat isolasi data, semakin besar pula kebutuhan sumber daya dan kompleksitas sistem. Oleh karena itu, diperlukan keseimbangan antara isolasi data, efisiensi, dan skalabilitas dalam memilih arsitektur yang tepat.

Berdasarkan kebutuhan sistem manajemen masjid berbasis SaaS yang memiliki banyak tenant dengan struktur data seragam, model *Shared Database – Shared Schema* dipilih karena lebih efisien dan mudah dikembangkan, meskipun tetap memerlukan kontrol akses yang baik dalam menjaga keamanan data.

4.2. Analisis Karakteristik Setiap Model

Setelah menganalisis strukturr dan mekanisme isolasi data, tahap berikutnya adalah mengkaji karakteristik utama dari setiap model arsitektur basis data *multi-tenant*. Analisis ini bertujuan untuk memahami pengaruh masing-masing model terhadap pengelolaan data, *scalability* dan *cost efficiency* dalam lingkungan *Software as a Service* (SaaS).

Model *Shared Database – Shared Schema* menggunakan satu basis data dan satu skema untuk seluruh *tenant*. Data dibedakan menggunakan atribut seperti *tenant_id*. Model ini unggul dalam *cost efficiency* dan *scalability*, serta lebih mudah dalam pengelolaan, namun memiliki tingkat isolasi data yang rendah.

Model *Shared Database – Separate Schema* menggunakan satu basis data dengan skema terpisah untuk setiap *tenant*. Pendekatan ini memberikan isolasi data yang lebih baik, tetapi meningkatkan kompleksitas dalam pengelolaan, terutama ketika jumlah *tenant* bertambah.

Model *Separate Database* menyediakan basis data terpisah untuk setiap *tenant*. Model ini menawarkan tingkat isolasi dan keamanan yang paling tinggi serta fleksibilitas dalam

pengelolaan data. Namun, pendekatan ini membutuhkan *cost efficiency* yang lebih besar dan pengelolaan yang lebih kompleks.

Secara umum, setiap model memiliki kelebihan dan keterbatasan masing-masing. Perbedaan ini menjadi dasar untuk analisis komparatif lebih lanjut berdasarkan parameter *scalability, security, performance, dan maintainability* pada sub-bagian berikutnya.

4.3. Analisis Trade-Off Arsitektur *Multi-Tenant*

Dalam perancangan sistem Software as a Service (SaaS), pemilihan arsitektur basis data *multi-tenant* tidak dapat ditentukan berdasarkan satu parameter. Oleh karena itu, dilakukan analisis *trade-off* untuk mengevaluasi keseimbangan antara *scalability, security, cost efficiency, performance,* dan *maintainability*. Setiap model memiliki keunggulan pada aspek tertentu, namun juga memiliki keterbatasan pada aspek lainnya.

Tabel 2. Analisis *Trade-Off* Arsitektur Basis Data *Multi-Tenant*

Aspek Analisis	***Shared Database – Shared Schema***	***Shared Database – Separate Schema***	***Separate Database***
Isolasi Data	Rendah (pemisahan logis melalui *tenant_id*)	Sedang (pemisahan melalui skema)	Sangat tinggi (pemisahan basis data)
Efesiensi Sumber Daya	Sangat Tinggi	Cukup Tinggi	Rendah
Skalabilitas	Sangat Tinggi	Tinggi	Sedang
Kompleksitas Manajemen	Rendah	Sedang	Tinggi
Biaya Infrastruktur	Rendah	Sedang	Tinggi
Fleksibilitas Pengelolaan Tenant	Terbatas	Cukup Fleksibel	Sangat Fleksibel
Risiko Kesalahan Akses Data	Lebih tinggi jika kontrol query tidak tepat	Lebih rendah	Sangat rendah

Berdasarkan tabel tersebut, terlihat adanya hubungan trade-off antara tingkat isolasi data (*security*) dan efesiensi biaya (*cost efficiency*). Model Shared Database – Shared Schema menunjukan keunggulan pada *scalability* dan *cost efficiency* karena penggunaan sumber daya yang terpusat.

4.4. Justifikasi Strategi Pemilihan Basis Data SaaS

Berdasarkan hasil analisis struktur, karakteristik, dan *trade-off* dari setiap model arsitektur, pemilihan strategi basis data dalam sistem *Software as a Service* (SaaS) perlu mempertimbangkan keseimbangan antara *scalability, cost efficiency*, dan *security*.

Dalam penelitian ini, sistem manajemen masjid dirancang untuk melayani banyak *tenant* dengan struktur data yang relatif seragam. Oleh karena itu, dibutuhkan model arsitektur yang mampu mendukung pertumbuhan *tenant* secara efisien tanpa meningkatkan kompleksitas sistem secara signifikan.

Dari ketiga model yang dianalisis, yaitu *Shared Database – Shared Schema*, *Shared Database – Separate Schema*, dan *Separate Database*, masing-masing memiliki kelebihan dan keterbatasan. Model *shared Schema* unggul dalam efisiensi sumber daya dan skalabilitas, model *separate schema* memberikan isolasi yang lebih baik, sedangkan model *separate database* menawarkan tingkat keamanan tertinggi namun dengan biaya dan kompleksitas yang lebih besar.

Dengan mempertimbangkan kebutuhan sistem, model *Shared Database – Shared Schema* dipilih sebagai strategi yang paling sesuai. Model ini mampu mendukung skalabilitas tinggi dan efisiensi sumber daya dalam mengelola banyak *tenant*.

Meskipun memiliki keterbatasan dalam isolasi data, hal ini dapat diatasi melalui penerapan kontrol akses dan pengelolaan *tenant* di tingkat aplikasi, seperti penggunaan atribut *tenant_id*. Dengan demikian, strategi ini tetap dapat menjaga *integritas* data sekaligus memberikan keseimbangan antara efisiensi dan kinerja sistem.

4.5 Pembahasan dan Rekomendasi

Setelah menganalisis parameter *Security*, *Cost Efficiency*, *Scalability*, *Performance*, dan *Maintainability*, penelitian ini menyimpulkan bahwa tidak ada satu pun model arsitektur basis data *multi-tenant* yang secara universal unggul di semua aspek tersebut. Temuan ini selaras dengan penelitian sebelumnya, yang menyatakan bahwa desain arsitektur multi-tenant selalu melibatkan trade-off antara efisiensi sumber daya dan tingkat isolasi data[4][12].

Jika dibandingkan dengan studi oleh Pippal et al. (2024) yang menekankan pada optimlisasi performa dan efisiensi pada arsitektur *multi-tenant*, temuan dari penelitian ini menegaskan bahwa *Shared Database Shared Schema* benar-benar lebih unggul dalam hal efisiensi sumber daya dan skalabilitas. Namun, penelitian ini memberikan bukti tambahan dengan memperlihatkan bahwa keunggulan tersebut sangat penting dalam konteks sistem yang memiliki banyak *tenant* dan data yang seragam, seperti dalam sistem manajemen masjid berbasis SaaS.

Di sisi lain, temuan dari studi ini sejalan dengan hasil yang diperoleh oleh Nugraha dan Chandra (2023) [5], yang menekankan signifikansi pemisahan data dengan metode *Separate Database* untuk memperkuat keamanan. Namun, riset ini memperluas sudut pandang dengan mengungkapkan bahwa peningkatan keamanan isolasi fisik berimbas langsung pada kenaikan biaya operasional dan kerumitan dalam pengelolaan sistem. Hal ini menegaskan bahwa terdapat hubungan *trade-off* yang tidak bisa dihindari dalam memilih arsitektur basis data.

Selain itu, penelitian yang dilakukan oleh Rapaka (2025) mengenai pola skalabilitas dallam sistem *cloud-native* juga menunjukkan bahwa pendekatan yang berbasis sumber daya bersama lebih mampu beradaptasi dengan peningkatan jumlah pengguna. Namun, penelitian ini mengaitkan aspek tersebut dengan parameter lain seperti pemeliharaan dan keamanan, sehingga menghasilkan analisis yang lebih menyeluruh dibandingkan dengan riset sebelumnya yang cenderung terfokus pada satu aspek.

Untuk memperkaya hubungan dengan studi sebelumnya, penelitian ini juga sejalan dengan penelitian terbaru Zhang et al. (2023) dan Kumar dan Singh (2022), yang menyatakan bahwa pendekatan *hibrida* semakin sering diajukan sebagai solusi tengah antara efisiensi dan keamanan dalam arsitektur *multi-tenant.* Dalam hal ini, temuan penelitian ini menegaskan pentingnya menjelajahi model *hibrida* sebagai arah pengembangan di masa depan.

Dari perspektif kontribusi akademis, studi ini menghadirkan sejumlah sumbangan signifikan di bidang sistem informasi serta arsitektur *Software as a Service* (SaaS). Pertama, penelitian ini menyajikan sebuah kerangka analisis komparatif yang menggabungkan lima elemen penting (*scalability, security, cost efficiency, performance,* dan *maintainability*) untuk menilai arsitektur basis data *multi-tenant*. Pendekatan ini memberikan tambahan nilai dibandingkan studi sebelumnya yang umumnya hanya menekankan satu atau dua elemen secara terpisah.

Kedua, studi ini menghasilkan model konseptual *trade-off* yang mengilustrasikan koneksi antara efisiensi sumber daya dan tingkat pemisahan data. Model ini bisa dijadikan dasar dalam pengambilan keputusan strategis oleh pengembangan sistem untuk menentukan arsitektur yang paling sesuai dengan kebutuhan aplikasi mereka.

Ketiga, penelitian ini berkontribusi dalam konteks dengan meneliti penerapan arsitektur *multi-tenant* pada sistem manajemen masjid berbasis SaaS, topik yang masih jarang dibahas dalam literatur yang ada. Hal ini memperluas jangkauan penerapan konsep *multi-tenant* ke dalam area sistem informasi yang berkaitan dengan agama.

Berdasarkan hasil yang ditemukan, studi ini merekomendasikan penerapan model *Shared Database Shared Schema* untuk sistem dengan karakterirtik *tenant* yang serupa dan jumlah yang besar. Namun, untuk mengatasi keterbatasan dalam aspek keamanan, diperlukan penerapan mekanisme tambahan seperti kontrol akses berbasis peran, validasi *query* yang ketat, serta pemisahan konteks *tenant* di tingkat aplikasi.

Untuk pengembangan yang akan datang, disarankan agar penelitian lanjutan dapat mengeksplorasi pendekatan arsitektur *hibidra*, melakukan pengujian kinerja sistem secara empiris dalam skala besar, serta mengintegrasikan teknologi keamanan seperti enkripsi data dan pemisahan berbasis kontainer untuk meningkatkan perlindungan data antar *tenant*.

5. Simpulan

Berdasarkan analisis arsitektur basis data multi-tenant dalam sistem *Software as a Service* (SaaS) untuk pengelolaan masjid, penelitian ini menunjukkan bahwa pemilihan

arsitektur dipengaruhi secara signifikan oleh kebutuhan akan efisiensi, tingkat isolasi data, dan kompleksitas pengelolaan sistem. Masalah utama yang diidentifikasi di awal penelitian adalah bagaimana memilih arsitektur yang dapat mendukung banyak *tenant* secara efektif tanpa mengorbankan keamanan data, yang menjadi dasar bagi proses analisis selanjutnya.

Temuan menunjukkan bahwa model *Shared Database–Shared Schema* dapat mencapai efisiensi pemanfaatan sumber daya yang tinggi dan mendukung skala sistem, menjadikannya lebih efektif di lingkungan dengan banyak *tenant* dan kebutuhan yang relatif seragam. Namun, model ini memiliki kelemahan dalam hal keamanan karena hanya mengandalkan isolasi logis, sehingga diperlukan peningkatan di tingkat aplikasi untuk mencegah kemungkinan kesalahan akses data.

Di sisi lain, model *Shared Database–Separate Schema* meningkatkan isolasi data namun meningkatkan kompleksitas dalam pengelolaan skema. Sementara itu, model *Separate Database* memberikan tingkat *security* dan isolasi data tertinggi, tetapi membutuhkan lebih banyak sumber daya infrastruktur dan melibatkan pengelolaan yang lebih rumit, yang membuatnya kurang efisien untuk sistem berskala besar dengan kebutuhan yang homogen.

Dengan mempertimbangkan temuan ini, penelitian ini menyimpulkan bahwa model *Shared Database–Shared Schema* adalah strategi yang paling tepat untuk menerapkan sistem pengelolaan masjid berbasis SaaS. Model ini dianggap mampu menangani masalah penelitian awal dengan menyeimbangkan efisiensi, kemampuan skala, dan kemudahan penerapan, asalkan didukung oleh mekanisme perlindungan data yang memadai di tingkat aplikasi.

Daftar Referensi

[1] N. T. Muhammed, “Growth Performance of Eucalyptus Microtheca Seedlings in Response to Different Levels of Organic and Inorganic Fertilizer,” *Qalaai Zanist Sci. J.*, vol. 7, no. 2, pp. 1135–1153, 2022, doi: 10.25212/lfu.qzj.7.2.44.

[2] R. K. R. Kumar, “Multi-Tenant SaaS Architectures: Design Principles and Security Considerations,” *J. Softw. Eng. Simul.*, vol. 6, no. 5, pp. 28–41, 2020, doi: 10.35629/3795-06052841.

[3] O.C. Adeusi, Y.O. Adebayo, P.A. Ayodele, T.T. Onikoyi, K.B. Adebayo, and I.O. Adenekan, “IT standardization in cloud computing: Security challenges, benefits, and future directions,” *World J. Adv. Res. Rev.*, vol. 22, no. 3, pp. 2050–2057, 2024, doi: 10.30574/wjarr.2024.22.3.1982.

[4] S. K. Pippal, S. Kumar, and R. Rani, “Optimizing multi-tenant database architecture for efficient software as a service delivery,” *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 22, no. 5, pp. 1128–1137, 2024, doi: 10.12928/TELKOMNIKA.v22i5.26385.

[5] S. Nugraha and D. W. Chandra, “Peningkatan Keamanan Database Pada Layanan Azure Melalui Metode Multi-Tenant Dengan Pendekatan Separate Database,” *J. Pendidik. dan Teknol. Indones.*, vol. 3, no. 6, pp. 233–240, 2023, doi: 10.52436/1.jpti.293.

[6] S. S. Boda, “A Structured Review of SaaS Security Architecture: Challenges, Models, and Research Gaps,” *IJIRCT2506031 Int. J. Innov. Res. Creat. Technol.*, vol. 11, no. 3, p. 1, 2025, [Online]. Available: https://www.researchgate.net/publication/395767792

[7] A. Krishna, “Systematic Literature Review of Software as a Service [SaaS] in view of Security and Multitenancy,” *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. 9, pp. 1742–1751, 2021, doi: 10.22214/ijraset.2021.38205.

[8] A. D. Alexander, “Studi Perbandingan Model Keamanan Data pada Cloud Computing,” vol. 6, no. 2, pp. 105–114, 2026.

[9] V. Singhania, “A Review of Cloud-Based Data Security Protocols,” vol. 2, no. 6, pp. 1–5, 2024.

[10] S. Chippagiri, “A Study of Cloud Security Frameworks for Safeguarding Multi-Tenant Cloud Architectures,” *Int. J. Comput. Appl.*, vol. 186, no. 60, pp. 50–57, 2025, doi: 10.5120/ijca2025924369.

[11] R.K. Sharma, “Multi-Tenant Architectures in Modern Cloud Computing: A Technical Deep Dive,” *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 11, no. 1, pp. 307–317, 2025, doi: 10.32628/cseit25111236.

[12] A. Rapaka, “Multi-Tenant System Design for Platform Scalability: Architectural Patterns

and Implementation Strategies for Modern Cloud-Native Applications," *J. Inf. Syst. Eng. Manag.*, vol. 10, no. 63s, pp. 314–324, 2025, doi: 10.52783/jisem.v10i63s.13863.

[13] V. Muniyandi, "Multi-Tenant SaaS Performance Isolation Using Container-Based Resource Sandboxing," *SSRN Electron. J.*, vol. 30, no. 2, pp. 677–695, 2025, doi: 10.2139/ssrn.5363393.

[14] M. Pande, S. Patil, and P. Student, "Designing a Framework for Detection and Capturing of Network attacks in Cloud Computing: a Literature Review," *Int. J. Innov. Res. Sci. Eng. Technol. (An ISO*, vol. 3297, pp. 4172–4178, 2007, doi: 10.15680/IJIRSET.2015.0406072.

[15] A. Shah and N. Bhatt, "A systematic review of in-memory database over multi-tenancy," *Int. J. Electr. Comput. Eng.*, vol. 14, no. 2, pp. 1720–1729, 2024, doi: 10.11591/ijece.v14i2.pp1720-1729.